# LoadRunner

Software Version: 12.60

## User Guide

Go to **HELP CENTER ONLINE**
http://admhelp.microfocus.com/lr/

Document Release Date: August 19, 2018 | Software Release Date: August 2018

# Legal Notices

## Disclaimer

Certain versions of software and/or documents ("Material") accessible here may contain branding from Hewlett-Packard Company (now HP Inc.) and Hewlett Packard Enterprise Company. As of September 1, 2017, the Material is now offered by Micro Focus, a separately owned and operated company. Any reference to the HP and Hewlett Packard Enterprise/HPE marks is historical in nature, and the HP and Hewlett Packard Enterprise/HPE marks are the property of their respective owners.

## Warranty

The only warranties for products and services of Micro Focus and its affiliates and licensors ("Micro Focus") are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein. The information contained herein is subject to change without notice.

## Restricted Rights Legend

Contains Confidential Information. Except as specifically indicated otherwise, a valid license is required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

## Copyright Notice

© Copyright 1994-2018 Micro Focus or one of its affiliates.

## Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

Oracle and Java are registered trademarks of Oracle and/or its affiliates.

# Contents

LoadRunner ..... 1

Welcome to the LoadRunner User Guide ..... 52
What's New ..... 52
TruWeb ..... 52
VuGen ..... 52
Controller and Analysis ..... 53
Technologies and platforms ..... 54

Virtual User Generator ..... 56
Introducing Virtual User Generator ..... 56
Overview ..... 56
Use VuGen scripts to test your environment ..... 56
Want to learn more? ..... 57
VuGen updates ..... 57
Vusers ..... 57
Vuser Technology ..... 57
Vuser Types ..... 59
VuGen User Interface ..... 59
VuGen Workspace ..... 59
How to Modify the VuGen Layout ..... 63
Keyboard Shortcuts ..... 65
File Menu ..... 65
Edit Menu ..... 66
View Menu ..... 66
Search Menu ..... 67
Design Menu ..... 67
Record Menu ..... 68
Replay Menu ..... 68
ALM Menu ..... 68
Windows Menu ..... 68
Solution Explorer Pane ..... 69
Understanding the Solution Explorer ..... 70
Solution Explorer Structure and Context Menu Options ..... 72
Step Navigator Pane ..... 75
Editor Pane ..... 77
Supported Programming Languages ..... 78
Tips for Working with the Editor ..... 79
Code Completion and Tooltips ..... 79
Code-Coloring ..... 80

Script Folding ... 81
Community Search ... 81
User Interface Elements ... 81
Steps Toolbox Pane ... 83
Bookmarks Pane ... 84
Snapshot Pane ... 85
Standard Snapshot pane controls ... 86
Snapshot pane controls for Citrix, RDP, SAP, and PCoIP protocols ... 87
Snapshot pane controls for Windows Sockets protocol ... 89
Snapshot pane controls for Web - HTTP/HTML protocol ... 89
Snapshot pane controls for protocols with an XML request/response (such as Web Services) ... 91
Snapshot pane controls for Database Protocols ... 92
Copying images to the clipboard ... 93
Copying snapshot text to the clipboard ... 93
Customized Snapshot pane functionality ... 93
Activating the Snapshot on error functionality ... 93
Thumbnail Explorer ... 94
Errors Pane ... 95
Tasks Pane ... 97
Task Editor ... 98
Output Pane ... 100
Breakpoints Pane ... 101
Breakpoints Pane ... 102
Breakpoint Condition Dialog Box ... 103
Call Stack Pane ... 104
Watch Pane ... 104
Runtime Data Pane ... 105
.NET Recording Filter Pane ... 106
Options Dialog Box ... 109
General Options Tab ... 109
Editor Options Tab ... 113
Scripting Options Tab ... 119
Search and Replace Dialog Boxes ... 128
Search Dialog Box ... 128
Replace Dialog Box ... 129
Incremental Search ... 130
Business Process Report Dialog Box ... 130
Replay Summary Pane ... 131
VuGen Workflow ... 136
Creating or Opening Vuser Scripts ... 136
Creating Vuser Scripts - Overview ... 136
Create and Open Vuser Scripts ... 137
Compare Scripts Side-by-Side ... 138

Working with Application Lifecycle Management 138
Managing Scripts Using ALM - Overview 138
Connect to ALM 138
ALM Version Control - Overview 139
Work with Scripts in ALM Projects 139
Work with Version-Controlled Scripts in ALM Projects 140
Save VuGen Vuser Scripts to ALM Projects 140
Compare Previous Versions of a Script 141
ALM Connection Dialog Box [VuGen] 141
Working with Git 143
Managing Scripts with Git - Overview 143
Work with Scripts in Git 143
Troubleshooting 144
Git Operations 144
Configure Ignore List Dialog Box 147
Resolve Conflicts Dialog Box 148
Import from Remote Repository Dialog Box 149
Multiple Protocol Scripts 149
Script Directory Files 150
Create a New Script Dialog Box 150
How to Work with .zip Files 152
Vuser Script Templates 153
Template Guidelines 153
Create and Open Vuser Script Templates 153
Create a Vuser Script Template 153
Create a Vuser Script From a Template 153
Rename a Vuser Script Template 153
Template Guidelines 154
Recording 154
Recording - Overview 154
Vuser Script Sections 155
VuGen Script Editor 155
Java Classes 155
Script Section Structure Example 156
Header Files 157
Record a Vuser Script 158
Recording Summary Report 159
Enable the Recording Summary report 159
Access the Recording Summary report 160
Recording Summary report overview 160
Filter recorded script using the Recording Summary report 162
Regenerate the script 163
Create a Business Process Report 163

Types of information included in business process reports 163
Create a business process report 164
Configure additional report options 164
Recording Options 164
Citrix > Configuration Recording Options 164
Citrix > Code Generation Recording Options 165
Citrix > Login Recording Options 165
ICA File Structure 167
Citrix > Recorder - Recording Options 168
Correlations > Configuration Recording Options 169
Correlations > Rules Recording Options 171
Advanced Correlation Properties Dialog Box 174
Token Substitution Testpad Dialog Box 175
Database > Database Recording Options 176
Database > Advanced Recording Options Dialog Box 176
Data Format Extension > Chain Configuration Recording Options 177
Add Prefix/Postfix to Chain Dialog Box 178
Add Data Format Extension 179
Data Format Extension > Code Generation Recording Options 180
Flex > RTMP Recording Options 182
Flex > Configuration Recording Options 182
Flex > Externalizable Objects Recording Options 183
FTP > Configuration Recording Options 184
General > Code Generation Recording Options 185
General > Protocol Recording Options 185
General > Recording - Recording Options 186
Advanced URL Dialog Box 186
Advanced HTML Dialog Box 187
General > Script Recording Options 188
GUI Properties > Web Event Configuration Recording Options 191
Custom Web Event Recording Configuration Dialog Box 192
GUI Properties > Advanced Recording Options 193
Recording Settings Properties 194
Code Generation Settings Properties 194
HTTP Properties > Advanced Recording Options 195
Headers Dialog Box 198
Content Type Filters Dialog Box 199
Non-Resources Dialog Box 199
Java > VM Recording Options 200
Java > Classpath Recording Options 201
Microsoft .NET > Recording - Recording Options 201
Remote Objects Property 203
Network > Mapping and Filtering Recording Options 204

Server Entry - Port Mapping Dialog Box ... 206
Advanced Port Mapping Settings Dialog Box ... 207
Server Entry - Traffic Filtering Dialog Box ... 208
Oracle NCA > Code Generation Recording Options ... 209
RDP > Code Generation > Advanced Recording Options ... 210
RDP > Code Generation > Agent Recording Options ... 211
RDP > Code Generation > Basic Recording Options ... 212
RDP > Client Startup Recording Options ... 213
Recording Properties > Correlation Options - Recording Options ... 214
Recording Properties > Log Options Recording Options ... 214
Recording Properties > Recorder Options - Recording Options ... 215
Recording Properties > Serialization Options - Recording Options ... 217
RTE > Configuration Recording Options ... 218
RTE > RTE Recording Options ... 218
SAPGUI > Auto Logon Recording Options ... 219
SAPGUI > Code Generation Recording Options ... 219
SAPGUI > General Recording Options ... 220
Traffic Analysis > Traffic Filters Recording Options ... 220
WinSock Recording Options ... 221
Recording Options - Miscellaneous Topics ... 223
Protocol Compatibility Table ... 223
Port Mapping and Traffic Filtering Overview ... 226
Port Mapping ... 226
Traffic Filtering ... 227
Port Mapping Auto Detection ... 227
EUC-Encoding (Japanese Windows only) ... 228
Script Generation Preference Overview ... 229
Script Language Options ... 229
Recording Levels - Overview ... 229
Serialization Overview ... 231
Tips for Working with Event Listening and Recording ... 231
Providing Authentication Information for Multi-Protocol Scripts ... 232
How Internet Explorer authenticates users ... 232
Generating a web_set_user function ... 233
Recording via a Proxy - Overview ... 233
Record a Script via a Proxy ... 235
Use Case 1 ... 235
Use Case 2 ... 236
Use Case 3 ... 237
Use Case 4 ... 238
After recording ... 238
Import Actions to a Script ... 239
Regenerate a Vuser Script ... 239

Start Recording Dialog Box ... 240
Floating Recording Toolbar ... 244
Files Generated During Recording ... 246
Troubleshooting and Limitations for Recording with VuGen ... 248
Proxy recording ... 248
Security Levels ... 248
Troubleshooting missing steps ... 248
Recording on Internet Explorer 10 ... 249
Recording on Microsoft Edge ... 249
Certificate warning message ... 249
Multi-Protocol recording ... 250
Overwriting of data ... 250
Firefox as default browser ... 250
FTP and Active SSL ... 250
Microsoft User Experience Virtualization Issue ... 250
HSTS Web Recording ... 250
FTP Recording ... 250
64-bit Recording ... 251
Correlating ... 251
Correlation Overview ... 251
Correlating Values ... 254
Overview ... 254
Correlation Types ... 255
Correlation Rules ... 256
Snapshot Details and Occurrences ... 256
Automatic Correlation ... 256
Rule-based correlation ... 256
Recording-based correlations ... 257
Replay-based correlations ... 257
Automatic Correlation Configuration ... 257
Exclude Strings or Content Types from the Correlation Scan ... 258
Design Studio ... 259
Manually Correlate Scripts ... 260
Design Studio - Correlation Tab ... 260
Correlation Tab - Main Area ... 261
Correlations grid ... 261
Details area ... 262
Modify Correlation Definitions ... 264
Modifying Boundary Based Correlation Definitions ... 264
Modifying Regular Expression Correlation Definitions ... 264
Modifying XPath Correlation Definitions ... 265
Modifying Winsocket Correlation Definitions ... 265
Correlate Scripts Using Design Studio ... 268

Advanced Correlation Techniques ... 269
Search for Values that Need Correlation ... 269
Search by Comparing Scripts ... 269
Replay Log Search ... 270
Wdiff Correlation Utility ... 270
Modifying Saved Parameters ... 271
Correlation Specifics for Protocols ... 271
COM Correlation ... 271
C Vuser Scripts Correlation Functions ... 271
lr_eval_string ... 272
lr_save_string ... 272
lr_save_var ... 272
Database, RTMP and COM Correlation from a Snapshot ... 272
Flex (RTMP/AMF) Correlation ... 273
Flex XPath Correlation ... 276
Java Custom Correlation ... 276
Java Scripts Correlation - Serialization ... 278
Microsoft .NET Correlation ... 281
Oracle NCA Correlation ... 282
Correlate Statements for Load Balancing ... 282
Correlate the JServSessionIdroot Values ... 283
Siebel Correlation ... 284
Correlation Library ... 284
Correlation Rules ... 285
Correlate SWECount Parameters ... 288
Correlate ROWID Parameters ... 288
Correlate SWET (timestamp) Parameters ... 289
Web Manual Correlations ... 289
Web Manual Correlation in Code ... 290
Web-based Correlation Using Design Studio ... 291
Winsock Correlation ... 293
Correlating a Winsock script ... 293
Parameterization versus Correlation ... 294
Winsock Manual Correlation ... 294
Replaying ... 295
Replay Overview ... 295
Replay a Vuser Script ... 296
Check Linux Compatibility ... 297
Work with Snapshots ... 297
Show the Snapshot pane ... 297
Copy a snapshot to the clipboard ... 298
Copy snapshot text to the clipboard ... 298
Activate the snapshot-on-error functionality ... 298

Set the snapshot options ... 298
Troubleshooting Snapshots ... 298
Snapshots that Have an XML View ... 299
Add a Text Check From the XML View in the Snapshot Pane ... 300
Running a Vuser as a Process or Thread ... 300
Runtime Settings ... 301
Configuring Runtime Settings ... 301
Access runtime settings ... 301
Search the runtime settings ... 302
Runtime setting descriptions ... 303
Runtime setting value validation ... 308
Revert runtime settings to the default settings ... 309
Preferences View - Internet Protocol ... 309
Checks ... 309
Web Performance Graph Generation ... 310
Advanced ... 310
HTTP ... 311
General ... 313
Authentication ... 316
Logging ... 317
JavaScript ... 317
Defining Non-Critical Resources ... 318
Prevent unnecessary failures due to errors with non-critical resources ... 318
Resource classification ... 318
Example ... 319
Import and Export Runtime Settings ... 319
Export runtime settings ... 319
Import runtime settings ... 319
Configure Runtime Settings Manually ... 320
General Options ... 320
Think Time Options ... 320
Log Options ... 321
Iterations and Run Logic ... 322
Bookmarks Overview ... 323
Run a Vuser Script from a Command Prompt ... 323
Use Bookmarks ... 324
Create a Bookmark ... 324
Remove a Bookmark ... 325
Navigate between Bookmarks ... 325
Navigate to a Specific Bookmark in a Vuser Script ... 325
Files Generated During Replay ... 325
NV Insights Report ... 327
Opening the NV Insights Report ... 327

Limitations 328
Debugging 328
Debugging Overview 328
Running a Vuser script 328
The Run Step by Step Command 329
Breakpoints 329
Bookmarks 329
Watching Variables 329
Go To Commands 330
Output Pane 330
Correlation 330
Error Handling 330
Additional Debugging Information 331
General Debugging Tip 331
Using C Functions for Tracing 331
Additional C Language Keywords 331
Add Additional Keywords 332
Examining Replay Output 332
Working with Breakpoints 332
Breakpoints Pane 332
Watching Expressions and Variables 333
Adding a New Watch to the Watch List 334
Editing a Watch Expression 334
Deleting a Watch Expression 334
Deleting All Watch Expressions From the Watch List 334
Debugging Web Vuser Scripts 335
Debug Scripts with Breakpoints 335
Enhancing 336
Enhancing a Script for Load Testing - Overview 336
Add Parameterization 336
Insert Transactions 337
Insert Rendezvous Points 337
Insert VuGen Functions 337
Insert Steps 337
Insert Comments 337
Insert Log Messages 338
Insert Synchronization Points (RTE Vusers only) 338
Transaction Overview 338
Insert Transactions 339
Insert a transaction while recording 339
Insert a transaction after recording 339
Transaction guidelines 340
Display Transactions 341

Display Hidden Transactions ... 341
Display Transactions With Errors ... 341
Display Transactions for Non-primary Steps ... 341
Cross-Vuser Transaction Overview ... 341
Create a Cross-Vuser Transaction ... 342
Rendezvous Points ... 343
Adding VuGen Functions Overview ... 344
Obtain Vuser Information ... 344
Send Messages to Output ... 344
General Vuser Functions ... 345
Protocol-Specific Vuser Functions ... 346
Encoding Passwords and Text ... 347
Encode a password ... 347
Mask a string ... 347
Database Integration Overview ... 348
Connecting to a Database ... 348
Using Data Retrieved from SQL Queries ... 349
Validating Database Values ... 351
Checking Returned Values Through a Database ... 352
Performing Actions on Datasets ... 353
Create a Controller Scenario from VuGen ... 354
Insert Steps into a Script ... 354
Insert Think Time Steps ... 354
Insert Debug Messages ... 355
Insert Error and Output Messages ... 355
Create Controller Scenario Dialog Box ... 355
Parameterization ... 357
Parameterization Overview ... 357
Delimiters ... 357
Input/Output parameters ... 357
Correlation ... 358
VTS and Parameterization ... 359
Parameter Types ... 359
File Type Parameters ... 359
Table Type Parameters ... 360
XML Type Parameters ... 360
Internal Data Type Parameters ... 360
User-Defined Function Parameters ... 361
Create Parameters ... 361
Import Parameter Values from a File ... 362
Work with Existing Script Parameters ... 363
Replace a value with a parameter ... 363
Replace multiple occurrences of a value with a parameter ... 363

Restore the original value ... 363
Export and Import VuGen script parameters ... 364
Data Assignment Methods for File-Type Parameters ... 365
Data Assignment and Update Methods for File Parameters ... 366
Vuser Behavior in the LoadRunner Controller ... 368
XML Parameters ... 368
Create an XML Parameter from a Web Service Call ... 369
Create XML Parameters - Standard Method ... 369
Define XML Value Sets ... 370
Set an Assignment Method ... 374
Modify XML Parameter Properties ... 374
Set AUT Environment Parameters ... 375
Select or Create Parameter Dialog Box ... 375
Parameter Properties Dialog Box ... 376
Date/Time, Group Name, Iteration Number, Load Generation Name, and Vuser ID Parameters ... 376
File Parameters ... 379
Table Parameters ... 380
Random Number Parameters ... 383
Unique Number Parameters ... 383
User Defined Function Parameters ... 384
XML Parameters ... 385
Parameter Simulation Dialog Box ... 385
Parameter List Dialog Box ... 389
Create Parameter Dialog Box ... 389
Parameter Original Value Dialog Box ... 390
Parameter Delimiters Configuration Dialog Box ... 390
Troubleshooting and Limitations for Parameterization ... 391
Asynchronous Communication ... 393
Synchronous and Asynchronous Concepts ... 393
Asynchronous Communication Types - Request and Response Sequences ... 394
Polling Asynchronous Communication ... 394
Long-Polling Asynchronous Communication ... 395
Push Asynchronous Communication ... 395
VuGen Support for Asynchronous Communication ... 396
Create an Asynchronous Vuser Script ... 397
Create a new Vuser script ... 398
Enable Async Scan ... 398
Record the business processes using the typical VuGen recording process ... 398
Generate, scan, and modify the Vuser script ... 398
Review the modifications that VuGen made to the script ... 398
Asynchronous Communication API ... 399
How Asynchronous Functions Differ from Synchronous Functions ... 400
How VuGen Modifies a Vuser Script for Asynchronous Communication ... 401

How VuGen modifies Vuser scripts ... 401
How VuGen modifies flex_amf_call steps ... 403
Defining the Start of an Asynchronous Conversation ... 404
Defining the End of an Asynchronous Conversation ... 405
Using Asynchronous Request Thresholds ... 407
Fine-Tuning the End of an Asynchronous Conversation ... 408
Correlating Asynchronous Vuser Scripts ... 408
Implementing Callbacks ... 409
Modifying Callbacks ... 412
Parsing URLs ... 416
Async Rules Overview ... 419
Adding Async Rules ... 419
Adding a positive Async rule ... 420
Adding a negative Async rule ... 421
Async Tab [Design Studio] ... 421
Asynchronous Options Dialog Box ... 423
Asynchronous Example - Poll ... 424
Asynchronous Example - Push ... 428
Asynchronous Example - Long-Poll ... 430
Viewing Replay Results ... 432
Virtual Table Server (VTS) ... 433
VTS Parameterization ... 434
Standard Parameterization vs. VTS Parameterization ... 434
Vuser protocols that support VTS functionality ... 434
Install VTS ... 434
Manage VTS ... 435
Accessing the VTS Server ... 435
Using Authentication ... 437
Configure authentication ... 437
SSL communication ... 437
Logging ... 438
Configuring VTS ... 438
Accessing the VTS configuration tool ... 438
VTS configuration settings ... 439
Creating and Managing Multiple VTS Instances ... 445
Create a named instance ... 445
Stop a named instance ... 446
Using the command line mode ... 446
VTS Data ... 447
The VTS Table ... 447
Populate the VTS table ... 447
VTS cell background colors ... 448
Divide the VTS table into pages ... 448

Refresh the VTS table data 449
Reconnect to the Database 449
Multiple VTS Instances 449
Manipulating the Contents of the VTS Table 449
Refreshing the data 450
Deleting the entire table 450
Changing the name of a column 450
Inserting a column 450
Deleting a column 450
Deleting the content of a column 450
Inserting a value at the top of a column 451
Appending a value to the end of a column 451
Searching Inside the VTS Table 451
Detecting Duplicate Entries 451
Check that an existing value is unique 451
Check if new or modified entries are unique 452
Importing Data into the VTS Table 452
Importing from a database 453
Importing from a CSV file 454
Using the sample files 454
Exporting Data from the VTS Table 454
Introduction 454
Exporting to a database 455
Exporting to a CSV file 456
Preparing Vuser Scripts for VTS Parameterization 456
VTS upgrade scenarios 456
Modifying old Vuser scripts 457
Upgrading Vuser Scripts for VTS 457
Using the API to Access the VTS Table 458
VuGen Protocols 460
IPv6 Support 463
IPv6 Deployment 464
Protocols Supported 464
Protocol Support Limitations 464
Troubleshooting 465
Protocol Support for 64-bit Recording, Async, and IPv6 465
Ajax - Click and Script Protocol 467
Ajax (Click and Script) Example Script 468
Ajax (Click and Script) - Replay Tips 469
Ajax (Click and Script) Miscellaneous Tips 470
Ajax (Click & Script) Enhancements 472
Ajax Click and Script Troubleshooting and Limitations 475
Citrix Protocol 478

Citrix Protocol - Overview ... 478
Set Up Your Citrix Environment ... 479
Agent for Citrix Presentation Server - Overview ... 481
Citrix Recording Tips ... 484
Citrix Synchronization ... 486
Citrix - Automatic Synchronization ... 487
Sync on Window ... 487
Sync on Obj Info ... 488
Sync on Text ... 488
Citrix - Manual Synchronization ... 489
Citrix - Additional Ways to Synchronize Your Script ... 490
Failed Bitmap Synchronization Dialog Box ... 491
Citrix Replay Tips ... 491
Wildcards ... 492
Run as a Process—not a Service ... 492
Modify the Citrix client window size for replay ... 492
Enable Think Time ... 492
Set Initialization Quota ... 492
Set a Bitmap Polling Delay ... 492
Use Exact Tolerance ... 493
Set Consistency Between Machines ... 493
Increasing the Number of Vusers per Load Generator Machine ... 493
Citrix Debugging Tips ... 493
Citrix - Troubleshooting and Limitations ... 495
General Limitations ... 495
Effects and Memory Requirements of Citrix Agent ... 497
Random Failures of Functions Accessing Citrix Agent ... 497
Citrix Agent will not start ... 497
Unexpected Disconnection ... 498
Citrix Receiver—Security Warning ... 498
Failed to get session from client ... 499
Citrix Error 13 "Unsupported Function" ... 499
Citrix Error 70, Client Error 1030 "Protocol driver error" ... 499
Capturing Empty Text ... 499
Click and Script Protocols ... 500
Click and Script Troubleshooting and Limitations ... 500
COM/DCOM Protocol ... 502
COM/DCOM Protocol Overview ... 503
COM/DCOM Technology Overview ... 503
Objects, Interfaces, and Type Libraries ... 503
COM Interfaces ... 503
COM Class Context and Location Transparency ... 504
COM Data Types ... 504

COM/DCOM Vuser Script Structure ... 504
COM Sample Vuser Scripts ... 505
Database Protocols ... 510
Database Protocols Overview ... 510
Supported protocols ... 510
Prerequisites ... 510
VuGen Database Recording Technology ... 512
Database Grids ... 513
Correlate a value in a data grid ... 513
Save the data in a data grid to a file ... 513
Copy the text from a cell in a data grid to the clipboard ... 513
Search for data inside a data grid ... 513
Handling Database Errors ... 514
Globally Modifying Error Handling ... 514
Locally Modifying Error Handling ... 515
Debugging Database Applications ... 515
Database Protocols - Troubleshooting and Limitations ... 517
Troubleshooting all database protocols ... 517
Two-tier Database Scripting Tips ... 518
Troubleshooting Oracle 2-Tier Vusers ... 522
Flex (RTMP/AMF) Protocol ... 526
Flex Overview ... 526
What is Flex? ... 526
Flex Technologies ... 526
Learn more about developing a Flex Vuser script ... 527
Recording Flex Scripts ... 528
AMF ... 528
RTMP Functions ... 530
RTMP Tunneled Functions ... 530
RTMP/RTMPT Streaming ... 531
RTMP Tunneled ... 538
Record a Flex Script ... 539
Setting the Flex Recording Mode ... 540
Example ... 540
Code Generation in the Flex Protocol ... 541
Code Generation Notification ... 541
Parsing Responses in Flex Scripts ... 541
Externalizable Objects in Flex Scripts ... 542
Flex Correlations ... 544
Flex Snapshots ... 544
Serialize Flex Scripts ... 545
External Java Serializer ... 545
Built-in Serializer ... 546

Query an XML Tree ... 546
Troubleshooting and Limitations for Flex ... 548
GraniteDS (Data Services) ... 549
Java Protocols ... 549
Set Up the Java Environment ... 550
Setting up the Java environment ... 550
JDK support matrix ... 551
Java Protocol Programming Tips ... 551
Manually Insert Java Methods ... 553
Enhance a Java Script ... 555
Insert transactions ... 555
Mark a transaction ... 555
Insert rendezvous points ... 555
Obtain Vuser information ... 556
Issue output messages ... 556
Emulate user think time ... 558
Handle command line arguments ... 558
Correlate Java Scripts ... 559
Insert Parameters into Java Scripts ... 560
Manually Configure Script Generation Settings ... 561
jquery.ini file structure ... 561
Class Name path ... 561
Automatic transactions ... 562
Default parameter values ... 562
Class pasting ... 562
Run Java Vuser Scripts ... 563
Compile and Run a Script as Part of a Package ... 564
Edit and Run Scripts in Eclipse ... 564
Debug Java Scripts ... 565
Notes and Limitations - Java Vuser protocols ... 566
Notes ... 566
Limitations ... 567
Java Record Replay Protocol ... 567
Supported Java communication protocols ... 567
Java Record Replay Protocol Recording Tips ... 568
Working with RMI ... 569
Working with CORBA ... 571
CORBA Application Vendor Classes ... 571
Editing a CORBA Vuser Script ... 572
Working with Jacada ... 573
Recording a Jacada Vuser ... 573
Editing a Jacada Vuser Script ... 574
Replaying Jacada Scripts ... 574

Working with JDBC ..... 575
Supported Databases ..... 575
Recording JDBC Scripts ..... 575
Limitations ..... 576
Sample Script ..... 576
Custom Java Filters ..... 577
Overview ..... 577
Determining which elements to include ..... 578
Create a custom Java filter ..... 579
Hook file structure ..... 580
Java Vuser (Manual) Protocol ..... 583
Manually Create a Java Script ..... 584
Java over HTTP Protocol ..... 586
Enable .jars for Recording with Java over HTTP ..... 586
Java applets ..... 587
Local Java applications ..... 587
Handle custom serialization and encryption of serialized objects ..... 588
Viewing Requests and Responses in XML Format ..... 588
Notes, Limitations, and Troubleshooting - Java over HTTP ..... 588
Notes ..... 589
Limitations ..... 589
Troubleshooting ..... 590
LDAP Protocol ..... 590
LDAP Protocol Overview ..... 590
LDAP Protocol Example Script ..... 591
Defining Distinguished Name Entries ..... 592
LDAP Connection Options ..... 593
Troubleshooting and Limitations - LDAP ..... 594
Mailing Service Protocols ..... 594
Mailing Service Protocols Overview ..... 595
IMAP Protocol Overview ..... 595
MAPI Protocol Overview ..... 595
POP3 Protocol Overview ..... 596
SMTP Protocol Overview ..... 597
Message Protocols ..... 598
MMS (Multimedia Messaging Service) Protocol Overview ..... 598
Run an MMS Scenario in the Controller ..... 598
Mobile Protocols ..... 599
Select a Recording Method for Mobile Applications ..... 599
Web HTTP/HTML ..... 599
TruClient - Native Mobile ..... 599
TruClient - Mobile Web ..... 600
SMP (SAP Mobile Platform) ..... 600

Speed Simulation for Mobile Vuser Scripts ..... 600
Web - HTTP/HTML - Recording Methods for Mobile Applications ..... 601
TruClient - Mobile Web Protocol ..... 602
TruClient - Mobile Web Protocol Overview ..... 602
How to Record a Script with TruClient - Mobile Web ..... 602
Add, Remove, and Import Mobile Device Settings for TruClient - Mobile Web ..... 603
Create a Custom Device Using the Mobile Device Manager ..... 603
Remove a Mobile Device ..... 603
How to Import a Mobile Device Settings to Your Script ..... 603
Mobile Device Dialog Box ..... 603
SMP (SAP Mobile Platform) Protocol ..... 604
MQTT Protocol ..... 605
Create an MQTT script ..... 605
Working with multi-protocol scripts ..... 606
Ready-to-use MQTT script template ..... 607
.NET Protocol ..... 609
.NET Protocol Overview ..... 609
Considerations for Working with the .NET Protocol ..... 610
Viewing Data Sets and Grids ..... 610
Recording WCF Duplex Communication ..... 611
Replacement of the Callback in the Script ..... 614
Asynchronous Calls ..... 615
Recording Dual HTTP Bindings ..... 616
Connection Pooling ..... 616
Debugging .NET Vuser Scripts ..... 618
.NET Filters Overview ..... 618
Guidelines for Setting .NET Filters ..... 619
.NET Filters - Advanced ..... 622
Reference List Dialog Box [.NET Protocol] ..... 623
Add Reference Dialog Box [.NET Protocol] ..... 624
Create a New Filter Dialog Box [.NET Protocol] ..... 625
Configure Application Security and Permissions ..... 626
Troubleshooting and Limitations - .NET ..... 626
Replay Limitations ..... 627
Recording Limitations ..... 627
Script Editing Limitations ..... 628
Oracle NCA Protocol ..... 628
Oracle NCA Protocol Overview ..... 628
Oracle NCA Protocol Example Scripts ..... 629
Oracle NCA Record and Replay Tips ..... 630
Pragma Mode ..... 631
Enable the Recording of Objects by Name ..... 632
Launch Oracle Applications via the Personal Home Page ..... 634

Oracle NCA - Troubleshooting and Limitations ... 635
General Limitations ... 635
Testing Secure Oracle NCA Applications ... 635
IE crashes when recording Oracle NCA or Oracle - Web scripts ... 636
Testing Servlets and other Oracle NCA Applications ... 636
Determining the Recording Mode ... 637
Recording Trace Information for Oracle DB ... 637
RDP Protocol ... 638
RDP Protocol - Overview ... 638
RDP Recording Tips ... 638
Single vs. Multi-Protocol Scripts ... 638
Record into Appropriate Sections ... 638
FIPS ... 638
Run a Clean Session ... 638
Explicit Mouse Clicks ... 639
Using Windows Logo key combinations ... 639
Synchronizing using Windows 8 apps ... 640
RDP Replay Tips ... 640
Modify the RDP remote desktop resolution size for replay ... 641
Working with Clipboard Data (RDP Protocol) ... 641
Correlating Clipboard Parameters ... 642
RDP Snapshots - Overview ... 643
Image Synchronization Overview (RDP) ... 643
Image Synchronization Tips (RDP Protocol) ... 644
Image Synchronization - Shifted Coordinates (RDP Protocol) ... 645
Setting Security Levels in RDP Vuser Scripts ... 646
RDP Agent (for Microsoft Terminal Server) Overview ... 649
Tips for Using the Agent for Microsoft Terminal Server ... 649
Enhancements to RDP functionality ... 650
Install/Uninstall the RDP Agent ... 651
Install the RDP Agent (Agent for Microsoft Terminal Server) ... 651
Use the RDP Agent (Agent for Microsoft Terminal Server) ... 651
Uninstall the RDP Agent (Agent for Microsoft Terminal Server) ... 652
Add Image Synchronization Points to a Script ... 652
Failed Image Synchronization Dialog Box (RDP Protocol) ... 652
Troubleshooting and Limitations for RDP ... 653
RTE Protocol ... 655
RTE Protocol Overview ... 655
Working with Ericom Terminal Emulation ... 656
SSL and SSH Support for Ericom ... 657
Typing Input into a Terminal Emulator ... 658
Setting the Timeout Value for TE_type ... 658
Allowing a Vuser to Type Ahead ... 659

Generating Unique Device Names ... 660
Setting the Field Demarcation Characters ... 660
Reading Text from the Terminal Screen ... 661
RTE Synchronization Overview ... 662
Synchronizing Block-Mode (IBM) Terminals ... 663
Synchronizing Character-Mode (VT) Terminals ... 665
Wait for the cursor to appear at a specific location ... 665
Instruct VuGen to automatically generate and insert TE_wait_cursor statements while recording ... 666
Wait for text to appear on the screen ... 666
Instruct VuGen to automatically generate and insert TE_wait_text statements while recording ... 667
Wait for the terminal to be silent ... 668
Map Terminal Keys to PC Keyboard Keys ... 668
Record RTE Vuser Scripts ... 669
Implement Continue on Error ... 671
Troubleshooting and Limitations - RTE ... 671
IP Spoofing ... 672
Disconnection Failures ... 672
Initialization Failures ... 672
SAP Protocols ... 672
Selecting an SAP Protocol Type ... 672
SAP GUI Protocol ... 673
SAP Web Protocol ... 676
Replaying SAP GUI Optional Windows ... 677
Configure the SAP Environment ... 677
Use IPV6 with a SAP installation ... 678
Check that SAPGUI Scripting API is enabled ... 678
Check the Patch Level ... 678
Check the Kernel Patch Level ... 678
Check the R/3 Support Packages ... 679
Enable scripting on the SAP Application Server ... 680
Enable scripting on SAP GUI client ... 682
Examine the hierarchy of GUI Scripting objects - Optional ... 683
Record SAP GUI Scripts ... 683
Replay SAP GUI Scripts ... 685
Run SAP GUI Scripts from the Controller ... 686
LoadRunner Controller Settings ... 686
Make Sure the Agent is Running in Process Mode ... 686
Enhance SAP GUI Scripts ... 687
Additional SAP Resources ... 690
Troubleshooting and Limitations for SAP ... 691
General SAP GUI limitations ... 691
I was able to record a script, but why does replay fail? ... 692
Why were certain SAP GUI controls not recorded? ... 692

What is the meaning of the error popup messages that are issued when I try to run the script? 692
Can I use the single sign-on mechanism when running a script on a remote machine? 692
Can VuGen record all SAP objects? 692
Are all business processes supported? 692
When I go to the Auto Logon node of the Recording Options, why is the list of server names empty? 693
Siebel Web Protocol 693
Siebel Web Protocol Overview 693
Siebel Web Recording Options and Runtime Settings 693
Record Transaction Breakdown Information 693
Siebel Web - Troubleshooting and Limitations 695
Silverlight Protocol 697
Silverlight Protocol - Overview 697
Silverlight - Troubleshooting and Limitations 697
Teradici PCoIP (PC over IP) Protocol 697
PCoIP client 698
Record a PCoIP test 698
Enhance the recorded script 698
Add a timeout for finding an area of a snapshot 698
Sample Script 699
TruClient Protocol 700
Convert TruClient scripts to TruClient - Web 700
Converting multiple scripts 700
Converting a single script 700
TruWeb Protocol (Tech Preview) 701
Web - HTTP/HTML Protocol 701
Web - HTTP/HTML Protocol - Overview 701
When should you use the Web - HTTP/HTML Vuser protocol? 701
Web - HTTP/HTML Vuser Technology 701
Learn how to develop a Web - HTTP/HTML Vuser script 702
JavaScript Web HTTP Scripts 704
Recording your Vuser Script in JavaScript 704
Converting a C Script to JavaScript 705
JavaScript Function Libraries 705
Using the VuGen JavaScript Engine 706
JavaScript Engine: XMLHTTPRequest Example 711
Convert a Web - HTTP/HTML Vuser Script into a Java Vuser Script 713
Create a Script for a REST API 713
REST API components 713
Generate web rest functions 714
Generate a web.rest function in JavaScript 715
Edit an existing web_rest function 715
Create script steps for a Web - HTTP/HTML script that calls a REST API 715

Using Emulation to Record Mobile Applications ... 716
Recording with a Google Android Emulator ... 717
Record HTTP/2 ... 721
Record and Replay on servers with SNI Enabled ... 722
Additional SNI Guidelines ... 722
Create a Vuser Script by Analyzing a Captured Traffic File ... 723
Create Web - HTTP/HTML scripts from TruClient Scripts ... 724
Record Streaming Media in Web - HTTP/HTML ... 724
Record Applications Using Smooth Streaming ... 725
Common Web Recording Problems ... 726
No events are being recorded ... 727
Very few events are being recorded ... 727
Specific events are not being recorded ... 727
Application hangs during recording ... 727
Wrong Server Certificate ... 728
Troubleshooting and Limitations - Web - HTTP/HTML Protocol ... 728
Web Protocols (Generic) ... 729
Web Protocols - Overview ... 729
Web Vuser Technology ... 730
Web Vuser Types ... 731
Text and Image Verification (Web Vuser Scripts) - Overview ... 731
Understanding Web Text Check Functions ... 732
web_reg_find ... 732
web_find ... 733
web_global_verification ... 733
Add Text Checks and Image Checks (Web Vuser Protocols) ... 733
Add a Text Check While Recording ... 733
Add a Text Check After Recording ... 733
Add Other Text Checks After Recording ... 734
Add an Image Check After Recording ... 734
Web Snapshots - Overview ... 734
Browser Emulation - Overview ... 735
What is a User-Agent String? ... 736
Emulating a specific browser ... 736
What information is included in Web Vuser's user-agent string? ... 737
Maximum number of concurrent connections ... 737
Perform Load Testing with nCipher HSM ... 738
Working with Cache Data ... 739
Create the cache file ... 739
Load the cache file into a Vuser ... 740
Insert Caching Functions ... 741
Data Format Extensions (DFEs) - Overview ... 741
When to use DFEs ... 741

DFE support 742
DFE chains 743
Replaying Vuser scripts that contain DFEs 744
Implement Data Format Extension (DFE) Support 744
Define a Chain of DFEs 745
Adding a DFE chain 745
Adding DFEs to the new DFE chain 746
Enable DFE Support 746
Configure DFE Support 747
Apply DFE Chains to Sections of the HTTP Message 748
How DFEs Modify a Vuser Script 749
Data Format Extension List 750
Applying DFEs to a String 751
Google Web Toolkit - Data Format Extension (GWT-DFE) 752
Overview 752
Example of code generated with and without GWT-DFE support 754
Auto-detection of GWT Remote Procedure Calls (RPCs) 756
How to apply a GWT Data Format Extension to your script before recording 756
Tips and troubleshooting 758
Implementing GWT-DFE Support 759
Prerequisites for implementing GWT-DFE support 759
Recording GWT-DFE Headers 759
Applying GWT-DFE chains 760
Troubleshooting - Data Format Extension (DFE) 761
Web Services Protocol 762
Web Services - Adding Script Content 762
Web Service Testing Overview 762
Adding Web Service Script Content - Overview 762
Recording a Web Services Script 763
Adding New Web Service Calls 763
Importing SOAP Requests 763
Analyzing Server Traffic 764
Script Integration 764
Web Service Call Attachments 764
Special Argument Types 765
Server Traffic Scripts Overview 769
Filtering Traffic 770
Data on Secure Servers 771
Add Content 771
Assign Values to XML Elements 773
Generate a Test Automatically 774
Create a Script by Analyzing Traffic (Web Services) 775
Specify Services Screen 775

Specify Application to Record Dialog Box ... 776
Import SOAP Dialog Box ... 777
New Web Service Call Dialog Box ... 778
Add Input Attachment Dialog Box ... 785
Add Array Elements Dialog Box ... 786
Process Base64 Data - Simple Data Dialog Box ... 787
Process Base64 Data - Complex Data Dialog Box ... 787
Aspects List ... 788
Specify Services Screen ... 789
Specify Traffic Information Screen ... 790
SSL Configuration Dialog Box ... 791
Web Services - Preparing Scripts for Replay ... 791
Preparing for Replay Overview ... 792
Testing Web Service Transport Layers Overview ... 792
Sending Messages over HTTP/HTTPS ... 792
JMS Transport Overview ... 792
JMS Script Functions ... 793
JMS Message Structure ... 794
Asynchronous Messages Overview ... 794
Sending Asynchronous Calls with HTTP/HTTPS ... 795
WS-Addressing ... 795
Customizing Overview ... 796
User Handlers ... 797
Custom Configuration Files ... 799
User Handler Examples ... 799
Prepare Scripts for Replay ... 801
Send Messages over JMS ... 803
Send Messages over HTTP/S ... 804
Define a Testing Method ... 805
Add a Database Connection ... 806
Create a User Handler ... 807
Customize Configuration Files ... 810
Web Services Snapshots - Overview ... 810
Database Connection Dialog Box ... 813
Connection String Generator Dialog Box ... 813
Web Services - Managing Services ... 814
Importing Services ... 814
Comparison Reports ... 815
Web Reference Analyzer ... 815
Add and Manage Services ... 816
Open the Manage Services Dialog Box ... 816
Import a Service ... 816
Get to Know the WSDL ... 816

Check for WSDL Updates - Optional ... 816
Override the Service Address- Optional ... 817
Set a Security Scenario - Optional ... 817
Analyze WSDL Dependencies ... 817
Manage Services Dialog Box ... 817
Connection Settings Dialog Box ... 821
Import Service Dialog Box ... 822
Search for Service in UDDI Dialog Box ... 822
XML/WSDL Comparison Dialog Box ... 823
WSDL Reference Analyzer Dialog Box ... 824
Web Services - Security ... 824
Setting Security Overview ... 824
Security Tokens and Encryption ... 825
SAML Security Options ... 827
Security Scenarios Overview ... 828
Choosing a Security Model ... 829
Private, Imported, and Shared Scenarios ... 829
Scenario Categories ... 830
WCF Scenario Settings ... 832
The WsHttpBinding Scenario ... 832
The Federation Scenario ... 833
The Custom Binding Scenarios ... 834
WCF Extensibility ... 835
Preparing Security Scenarios for Running ... 836
Parameterizing Security Elements ... 837
Protecting Custom Headers ... 837
Emulating Users with Iterations ... 838
Add Security to a Web Service Script ... 838
Customize the Security ... 839
Add SAML Security ... 841
Create and Manage Security Scenarios ... 842
Parameterize Security Elements ... 844
Set Security Properties Dialog Box ... 845
WS—Security Tab ... 845
WS Addressing ... 848
Security Scenario Editor Dialog Box ... 848
Advanced Settings Dialog Box ... 849
Select Certificate Dialog Box ... 853
Web Services Security Examples ... 854
Troubleshooting and Limitations for Web Services ... 857
Windows Sockets Protocol ... 858
Recording Windows Sockets - Overview ... 858
Translation Tables ... 858

Windows Sockets Data ... 859
Windows Sockets Snapshots - Overview ... 860
Displaying Buffer Data ... 860
Buffer Views ... 860
Status Bar ... 861
Navigating within the Buffer Data ... 862
Creating Correlations from the Buffer Data ... 862
Data Navigation Tools ... 862
Buffer Data Editing ... 862
Record a Windows Sockets Script ... 863
View and Modify Windows Sockets Buffers ... 864
Modifying buffer data ... 864
View and modify the data in the data.ws file ... 864
View the data in the Snapshot pane ... 865
Navigate within the snapshot data ... 865
Insert data into a buffer ... 865
Copy and paste blocks of data ... 866
Data Buffers ... 866
Go To Offset Dialog Box ... 867
Customize Your Scripts ... 869
Create a PCAP File ... 869
Create a capture file ... 869
Troubleshooting missing packets ... 870
Manually Program a Script using the VuGen Editor ... 871
Manually Programming Scripts - Overview ... 871
Programming Vuser Actions ... 871
Create a Template ... 872
Define Transaction and Insert Rendezvous Points Manually ... 873
C Vuser Scripts ... 873
Java Vusers ... 874
.NET Vusers ... 875
Troubleshooting and Limitations - Programming ... 876
Framework 4.5 for .NET scripts ... 876
Framework 3.5 for .NET scripts ... 876
Missing references for scripts created in Visual Studio ... 876
Create Scripts in External IDEs ... 877
Creating Vuser Scripts or Unit Tests in Visual Studio or Eclipse ... 877
Create a Vuser Script in Visual Studio ... 878
Create a Vuser Script in Eclipse ... 879
Prerequisite ... 879
Method 1: Create a script in VuGen, and develop it further in Eclipse ... 879
Method 2: Create the script in Eclipse ... 879
Develop the script in Eclipse ... 880

Develop a Unit Test Using Visual Studio (NUnit test) ... 880
Develop a Unit Test Using Eclipse (JUnit or Selenium test) ... 881
Use DLLs and Customize VuGen ... 882
Calling Functions from External DLLs ... 882
Load a DLL Locally ... 882
Load a DLL Globally ... 883
VuGen File and Library Locations ... 884
Storing Runtime Settings in External Files ... 885
Command Line Parameters ... 885
Create and Run Scripts in Linux ... 886
Creating and Running Scripts in Linux - Overview ... 886
Compile Scripts Manually on Linux ... 886
Run a Vuser Script from a Linux Command Line ... 887
Program with the XML API ... 889
Programming with the XML API Overview ... 889
Using XML Functions ... 889
Reading Information from an XML Tree ... 890
Writing to an XML Tree ... 890
Specifying XML Function Parameters ... 891
Defining the XML Element ... 892
Querying an XML Tree ... 892
Multiple Query Matching ... 892
XML Attributes ... 893
Structuring XML Scripts ... 893
Enhancing a Recorded Session with XML ... 894
Use Result Parameters ... 898
Non-English Language Support ... 899
Non-English Language Support Overview ... 899
Page Request Header Language ... 900
Convert Encoding Format of a String ... 900
Convert Encoding Format of Parameter Files ... 901
Record Web Pages with Foreign Languages ... 902
Automatically Record Foreign Language Web Pages. ... 902
Manually Record Foreign Language Web Pages ... 903
Troubleshooting and Limitations for Non-English Languages ... 903
Share Software Content Resources ... 906
Additional Components ... 907
Where to install additional components ... 907
Standalone Applications ... 911
Protocol SDK ... 912
Install the Protocol Library Package ... 913
Upgrade Protocol SDK Projects from Visual Studio 2012 ... 913
Installing the Microsoft Terminal Server Agent ... 913

Install the Agent for Microsoft Terminal Server ... 913
Uninstall the Agent for Microsoft Terminal Server ... 914
Troubleshooting and Limitations for Additional Components ... 914
Troubleshooting and Limitations for VuGen ... 914
Internet Explorer Issue ... 914
Error Messages ... 914
Slow replay of JavaScript language scripts ... 915
Installing and Upgrading JVMs ... 915
McAfee Compatibility Issues ... 915

Controller ... 916
Introducing Controller ... 916
Primary Items in Scenario ... 916
Scripts and Test Types ... 916
Licenses ... 917
Controller Technology ... 918
Example ... 918
LoadRunner Agents ... 919
LoadRunner Terminology ... 919
Plan the LoadRunner Implementation ... 920
Define the scope of performance measurements ... 921
Define Vuser activities ... 922
Select Vusers ... 923
Choose testing hardware/software ... 923
So how many Vusers can I run on a Load Generator? ... 924
The LoadRunner Testing Process ... 924
1. Planning the Test ... 924
2. Creating the Vuser Scripts ... 924
3. Designing the Scenario ... 925
4. Running the Scenario ... 925
5. Monitoring the Scenario ... 925
6. Analyzing Test Results ... 925
LoadRunner Licensing ... 925
About LoadRunner Licenses ... 926
License types ... 926
License bundles ... 926
VUD Licenses ... 927
License Consumption ... 927
Install LoadRunner Licenses ... 928
LoadRunner Community license (free) ... 928
Paid LoadRunner license ... 929
LoadRunner License Utility ... 930
Troubleshooting Licenses ... 932

Controller Window ... 933
Designing Load Test Scenarios ... 936
Planning Load Test Scenarios ... 936
Load Test Planning Overview ... 936
Load Testing Objectives ... 936
Measuring End-User Response Time ... 937
Defining Optimal Hardware Configuration ... 937
Checking Reliability ... 937
Checking Hardware or Software Upgrades ... 937
Evaluating New Products ... 937
Identifying Bottlenecks ... 938
Measuring System Capacity ... 938
Plan a Load Test ... 939
How to Analyze the Application ... 939
Identify system components ... 939
Describe the system configuration ... 940
Analyze the usage mode ... 941
Examine task distribution ... 941
Define the Load Test Objectives ... 941
Designing Scenarios ... 942
Manual Scenarios ... 942
Changing Scenario Modes ... 942
Goals Types for Goal-Oriented Scenarios ... 943
Virtual Users ... 943
Pages per Minute/Hits per Second/Transactions per Second ... 943
Transaction Response Time ... 944
JMeter Tests ... 945
JMeter tests overview ... 945
Using JMeter tests in a scenario ... 946
Using JMeter remote/distributed testing ... 946
Set Up the JMeter Test ... 946
Prerequisite setup for JMeter tests ... 947
Add a JMeter test to a scenario ... 947
JMeter additional properties file ... 948
JMeter Best Practices and Troubleshooting ... 948
Graceful shutdown ... 949
Best practices ... 949
Limitations ... 949
Troubleshooting JMeter tests ... 949
Noise Scenarios ... 950
How to Design a Goal-Oriented Scenario ... 951
1. Prerequisites ... 951
2. Open a new goal-oriented scenario ... 951

3. Add load generators to the scenario 951
4. Assign load generators to each script 951
5. Define a goal for the scenario 952
6. Define a virtual location for each script in the scenario - optional 952
7. Assign each script a percentage of the total scenario target 952
8. Define virtual services for the scenario - optional 952
9. Define service level agreements for the scenario - optional 952
How to Design a Manual Scenario 953
1. Prerequisites 953
2. Open a scenario, or create a new one 953
3. Add load generators to the scenario 953
4. Add Vuser groups/scripts to the scenario - Vuser group mode 953
5. Add Vuser groups/scripts to the scenario - Percentage mode 954
6. Define a virtual location for the scenario - optional 954
7. Define a schedule for the scenario 954
8. Define virtual services for the scenario - optional 954
9. Define service level agreements for the scenario - optional 954
How to Change the Scenario Mode (Manual Scenario) 954
How to View/Modify Scripts in the Scenario 955
View script details 955
Modify a script's runtime settings 955
Modify multiple scripts' runtime settings 956
View/Edit a script in VuGen 956
Specify command line options 956
View rendezvous points included in the script 956
View Vusers associated with the script 956
View files associated with the script 957
Relative Paths for Scripts 957
Vuser Statuses 958
Add Group Dialog Box 958
Add Script Dialog Box 960
Add Vusers Dialog Box 961
Design Tab 962
Edit Scenario Goal Dialog Box 965
Group Information Dialog Box 967
Multiple Runtime Settings Mode Dialog Box 968
New Scenario Dialog Box 969
Scenario Goal Pane 971
Scenario Groups/Scripts Pane - Manual Scenarios 972
Scenario Scripts Pane - Goal-Oriented Scenarios 975
Scenario Start Time Dialog Box 977
Script Information Dialog Box 978
Vuser Information Dialog Box 980

Vusers Dialog Box ... 981
Load Generators ... 983
Creating load generators ... 983
Local versus cloud-based load generators ... 983
Load generator configuration ... 984
Status of a load generator ... 984
Setting up a Load Generator Environment ... 984
Configure Dockerized Load Generators ... 985
About Dockerized load generators ... 986
Dockerized load generator image ... 986
Assign load generator containers to scenarios ... 986
Load Generators on the Cloud ... 987
Define your cloud account in LoadRunner ... 987
Set up your cloud account for provisioning load generators ... 988
Set up secure communication: ... 989
Manage Load Generators on the Cloud ... 990
Provision load generators on the cloud ... 990
Rename provisioned load generators ... 990
Delete provisioned load generators from scenario ... 990
Terminate provisioned load generators automatically after scenario run ... 991
Create Certificates for Azure Cloud ... 991
Create Custom Image for Load Generators on the Cloud ... 992
Azure – Windows ... 992
Azure - Ubuntu ... 994
Amazon EC2 – Windows ... 995
Amazon EC2 - Ubuntu ... 996
Google Cloud - Windows ... 997
Google Cloud - Ubuntu ... 999
DigitalOcean - Ubuntu ... 1001
Add Load Generators to a Scenario ... 1002
How to Modify Load Generator Settings ... 1004
Load Balancing ... 1005
How to Connect/Disconnect a Load Generator ... 1005
How to Connect to a Linux Load Generator Without Using RSH ... 1005
How to Change the Load Generator Temporary Folder [Linux] ... 1006
Linux Environment Variables ... 1006
Add New Load Generator/Load Generator Information Dialog Box ... 1007
Load Generator Configuration > Connection Tab ... 1009
Load Generator Configuration > Connection Log Tab ... 1010
Load Generator Configuration > Linux Environment Tab ... 1011
Load Generator Configuration > Network Virtualization Tab ... 1013
Load Generator Configuration > Runtime File Storage Tab ... 1014
Load Generator Configuration > Runtime Quota Tab ... 1015

Load Generator Configuration > Status Tab ... 1016
Load Generator Configuration > Terminal Services Tab ... 1016
Load Generator Configuration > Vuser Limits Tab ... 1017
Load Generator Configuration > Vuser Status Tab ... 1018
Load Generators Dialog Box ... 1019
Manage Cloud Accounts Dialog Box ... 1022
Create Cloud Load Generator Dialog Box ... 1024
Create Cloud Load Generator Sections ... 1025
Use Cloud Load Generator Dialog Box ... 1027
Network Profile Manager Dialog Box ... 1028
LoadRunner Agent Runtime Settings Dialog Box ... 1030
SSL Utility ... 1031
Converting Certificates ... 1031
Test Connection ... 1032
Remove Certificate Encryption ... 1032
Troubleshooting and Limitations - Load Generators ... 1032
Load Generator not included with "All Generators" ... 1032
Overloading of CPU ... 1033
Custom Cipher ... 1033
Virtual Machines ... 1033
VB and C# Projects ... 1033
Limited Users ... 1033
ODBC scripts on Linux machines ... 1033
GUI Vusers ... 1034
Non-English Locales ... 1034
Scheduling Manual Scenarios ... 1034
Scheduling Manual Scenarios Overview ... 1034
Scheduling by Scenario or Group ... 1034
Scheduling By Scenario ... 1034
Scheduling By Vuser Group ... 1035
Schedule Run Modes ... 1035
Schedule Actions ... 1036
Start Group ... 1036
Initialize ... 1037
Start Vusers ... 1038
Duration ... 1038
Stop Vusers ... 1039
Define a Schedule for the Scenario ... 1039
Add and Edit Schedule Actions ... 1041
Add a schedule action ... 1042
Edit a schedule action ... 1042
Add Action Dialog Box ... 1042
Edit Action Dialog Box ... 1043

Scenario Schedule Pane ... 1043
Schedule definition area ... 1044
Action grid ... 1045
Schedule graph ... 1047
Service Level Agreements ... 1048
Multiple IP Addresses ... 1048
Multiple IP Addresses Overview ... 1048
How to Add IP Addresses to a Load Generator ... 1048
IP Wizard ... 1049
LoadRunner Terminal Services Manager ... 1051
Terminal Services Overview ... 1051
Using the Terminal Services Manager ... 1052
Running Load Test Scenarios ... 1053
Online Monitor Graphs ... 1053
Online Monitor Graphs Overview ... 1054
About Online Monitor Graphs ... 1054
Viewing Monitor Data Offline ... 1054
How to Display Online Monitor Graphs ... 1054
How to Customize Online Graph and Measurement Settings ... 1055
Configure the graph settings ... 1055
Configure the measurement settings ... 1055
Manage Online Graphs ... 1057
Freeze graphs ... 1057
Overlay graphs ... 1058
Export graphs to HTML ... 1058
Export a single graph ... 1058
Graph Configuration Dialog Box ... 1058
Measurement Configuration Dialog Box ... 1060
Open a New Graph Dialog Box ... 1061
Overlay Graphs Dialog Box ... 1062
Available Graphs Tree ... 1063
Configuring Scenario Options ... 1067
Configuring Scenario Options Overview ... 1067
Expert Mode ... 1068
Runtime File Storage Locations ... 1068
Shared Network ... 1068
Path Translation ... 1070
What is path translation? ... 1070
Path Translation Table ... 1071
Examples ... 1072
How to Configure Scenario Options ... 1072
Configure timeout options ... 1072
Configure Vuser runtime settings ... 1072

Configure general scenario options for Expert mode ... 1073
Configure the default schedule run mode ... 1073
Define a command to run after scenario results are collated ... 1073
Configure global runtime file storage options ... 1073
Configure monitoring options ... 1073
Configure debug information options (Expert mode only) ... 1074
Configure output display options (Expert mode only) ... 1074
Options Dialog Box ... 1074
Options > Debug Information Tab ... 1075
Options > Execution Tab ... 1076
Options > General Tab ... 1077
Options > Monitors Tab ... 1078
Options > Output Tab ... 1080
Options > Path Translation Tab ... 1080
Options > Runtime File Storage Tab ... 1081
Options > Runtime Settings Tab ... 1082
Options > Timeout Tab ... 1083
Before Running Your Scenario ... 1085
How to Prepare a Scenario to Run ... 1085
Specify result file name and location ... 1085
Specify Scenario Runtime Settings ... 1086
Set up the monitoring environment - optional ... 1086
Enable Automatic Result Collation - optional ... 1087
Enable Auto Load Analysis- optional ... 1087
Schedule scenario - Optional ... 1087
Provide scenario summary information - optional ... 1087
Set up the scenario to run MMS (Media Player)- optional ... 1087
Set up the scenario to run GUI Vusers - optional ... 1088
Set Results Directory Dialog Box ... 1088
Summary Information Dialog Box ... 1089
Rendezvous Points ... 1090
Rendezvous Points Overview ... 1090
How to Set Up a Rendezvous in a Scenario ... 1090
Rendezvous Information Dialog Box ... 1091
Running Scenarios in Controller ... 1093
Running Scenarios Overview ... 1093
Before run ... 1093
Start of run ... 1093
During run ... 1093
End of run ... 1094
How to Run a Scenario ... 1094
Use-Case Scenario - Initialize/Run Additional Vusers or Stop Running Vusers ... 1096
Initialize/Run additional Vusers in Vuser group mode ... 1096

Initialize/Run additional Vusers in Percentage mode ... 1098
Stop running Vusers in Vuser group mode ... 1100
Stop running Vusers in Percentage mode ... 1101
Execution Notes Dialog Box ... 1102
Output Window ... 1102
Filtered Tab ... 1103
Summary Tab ... 1104
Run Tab ... 1106
Run/Stop Vusers Dialog Box ... 1108
Scenario Groups Pane ... 1110
Scenario Status Pane ... 1111
Transactions Dialog Box ... 1112
Vuser Script Log ... 1112
Running Scenarios from the Command Line ... 1114
Running Scenarios using the Command Line Interface (CLI) Tool ... 1114
Invoking the CLI Tool ... 1115
Command line format ... 1115
Examples ... 1116
XML Input File ... 1116
XML Input File Samples ... 1118
Notes about running tests using the CLI tool ... 1119
Running Scenarios Using the wlrun Command ... 1120
Invoking the Controller using the wlrun command ... 1120
wlrun runtime arguments ... 1120
Notes about running tests using the wlrun command ... 1121
Using Unified Functional Testing Tests in LoadRunner ... 1122
Using Unified Functional Testing Tests in LoadRunner - Overview ... 1122
About GUI Vuser Scripts ... 1123
Understanding GUI Vuser Technology ... 1123
Guidelines for Using QuickTest or Unified Functional Testing Tests in LoadRunner ... 1124
Limitations ... 1124
Including Transactions ... 1124
Adding Statements ... 1125
Designing Tests for LoadRunner ... 1125
How to Add a Unified Functional Testing Test to a Load Test Scenario ... 1125
Managing Scenarios Using Application Lifecycle Management ... 1126
Managing Scenarios Using Application Lifecycle Management - Overview ... 1126
How to Work with Scenarios in ALM Projects ... 1126
How to Connect to ALM ... 1126
How to Save Scenarios to ALM Projects ... 1127
Add Vuser Scripts from an Application Lifecycle Management Project ... 1128
Add a Vuser script to a manual scenario ... 1128
Add a Vuser script to a goal-oriented scenario ... 1128

ALM Connection Dialog Box [Controller] 1128
Continuous Integration with Jenkins 1130
After the Scenario Run 1130
Post Scenario Run Procedures - Overview 1130
Collating Run Data 1131
How to Collate Scenario Run Results 1131
Scenario Run Results 1132
LoadRunner Raw Results Exporter 1133
Launching the tool 1134
Notes 1134
JSON output 1134
InfluxDB output 1135
Collate Results Dialog Box 1135
Working with Firewalls in LoadRunner 1136
Set Up Your LoadRunner System Over Firewalls 1136
Set Up an Over-Firewall Deployment 1137
Configure the LoadRunner Agent 1140
Create and Verify the Connection Between Controller and Agent Machines 1142
Set Firewall Monitoring Preferences 1143
MI Listener Configuration Dialog Box 1144
Monitor Configuration Dialog Box 1145
Agent Configuration Dialog Box 1146
Agent Configuration Settings Dialog Box 1148
Secure communication over Firewalls 1149
Troubleshooting and Limitations for Firewalls 1150
Checking Connectivity 1150
Windows Firewall Considerations 1152
Linux LoadRunner Agent Menu 1153
Secure Communication with TLS (SSL) 1153
Configure Secure Communication with TLS (SSL) 1153
Manage certificates using the LoadRunner Certificate Manager 1154
Manage certificates using the command line utility 1155
Other tools for setting up communication with communication with TLS 1157
About Certificates 1157
LoadRunner Default Certificate 1157
Certificate Attributes and Requirements 1157
Two-way TLS (SSL) Authentication 1158
Configure Secure Communication with Two-way TLS (SSL) Authentication 1160
LoadRunner Certificate Manager 1161
Authentication Settings Tool 1163
Network and Security Manager - Command Line Tool 1163
Launching the Network and Security Manager tool 1164
Command line arguments for the Network and Security Manager tool 1164

Common Examples ... 1168
Monitoring Load Test Scenarios ... 1171
How to Set Up a Monitoring Environment ... 1171
Monitor Types ... 1173
Add Machine Dialog Box ... 1175
Access information ... 1175
Monitored Machine Information ... 1175
SiteScope Server Information ... 1176
Dynatrace Server Information ... 1176
<Monitor Type> Monitor Configuration Dialog Box ... 1177
<Monitor Name> Dialog Box ... 1177
Application Deployment Solution Monitoring ... 1178
Application Deployment Solution Monitoring Overview ... 1178
How to Set up the Citrix Monitoring Environment ... 1178
Citrix Server Performance Counters ... 1179
Non-Virtual Counters ... 1179
Virtual Channel Counters ... 1182
Citrix Monitor Dialog Box ... 1184
Database Server Resource Monitoring ... 1185
Database Resource Monitoring Overview ... 1185
How to Set Up the Oracle Monitoring Environment ... 1185
Oracle Performance Counters ... 1187
SQL Server Performance Counters ... 1188
Dynatrace Monitor ... 1189
Dynatrace dashboards and dashlets ... 1189
Set up the Dynatrace monitor ... 1190
Flex Monitoring Graphs ... 1190
Infrastructure Resources Monitoring ... 1191
Infrastructure Resources Monitoring Overview ... 1191
Network Client Performance Counters ... 1191
JMeter Monitoring Graphs ... 1191
Middleware Performance Monitoring ... 1192
How to Set Up the IBM WebSphere MQ Monitor ... 1193
IBM WebSphere MQ Performance Counters ... 1194
Queue Performance Counters ... 1194
Channel Performance Counters ... 1194
IBM WebSphere MQ Queue Attributes ... 1196
MQ Monitor Add Measurements Dialog Box ... 1197
MQTT Statistics Monitoring ... 1198
MQTT Statistics monitors ... 1199
MQTT Statistic Measurements ... 1199
Network Delay Monitoring ... 1201
Network Monitoring Overview ... 1201

How to Set Up the Network Monitoring Environment ... 1202
How to Configure the Linux Source Machine for Network Monitoring ... 1203
Adding Destination Machines for Network Delay Monitoring Dialog Box ... 1205
Network Delay Time Dialog Box ... 1205
Network Delay Time Graph ... 1206
Network Monitor Settings for Defined Path Dialog Box ... 1207
Troubleshooting and Limitations - Network Delay Monitor ... 1208
Network Virtualization Monitoring ... 1209
Average Latency Monitor Graph ... 1209
Packet Loss Monitor Graph ... 1210
Average Throughput Monitor Graph ... 1213
Average Bandwidth Utilization Monitor Graph ... 1214
Total Throughput Monitor Graph ... 1216
Runtime and Transaction Monitoring ... 1217
Runtime Graphs Overview ... 1217
Running Vusers Graph ... 1218
User-Defined Data Points Graph ... 1218
Error Statistics Graph ... 1219
Vusers with Errors Graph ... 1219
Transaction Monitor Graphs Overview ... 1219
SiteScope Server Monitoring ... 1220
SiteScope Resource Monitoring ... 1220
Set up the SiteScope Integration ... 1220
Set up SiteScope without authentication ... 1220
Set up SiteScope with authentication as a non-anonymous user ... 1221
Streaming State Statistics Monitoring ... 1221
Streaming State Statistics Monitors ... 1222
Streaming State Statistics Measurements ... 1222
System Resource Monitoring ... 1223
System Resource Monitors Overview ... 1223
Windows Resource Monitoring ... 1224
UNIX Resource Monitoring ... 1224
SNMP Resource Monitoring ... 1225
How to Set up the UNIX Monitoring Environment ... 1225
UNIX Resources Performance Counters ... 1226
Add Windows Resources Measurements Dialog Box ... 1227
TruClient - Native Mobile Monitors ... 1227
CPU Utilization Percentage Graph ... 1227
Total Free Memory In Device Monitor ... 1228
Total Memory Consumed by Application Monitor ... 1228
Web Application Server Monitoring ... 1228
MS Active Server Pages Performance Counters ... 1228
Microsoft Active Server Pages Dialog Box ... 1229

Web Resource Monitors ... 1230
Web Resource Monitoring Overview ... 1230
Hits per Second Graph ... 1230
Throughput Graph ... 1230
HTTP Responses per Second Graph ... 1231
WebSocket Statistics Graph ... 1231
Pages Downloaded per Second Graph ... 1231
Retries per Second Graph ... 1232
Connections Graph ... 1233
Connections per Second Graph ... 1233
SSLs per Second Graph ... 1233
WebSocket Statistics Monitor ... 1233
HTTP Status Codes ... 1234
Web Server Resource Monitoring ... 1235
Web Server Resource Monitoring Overview ... 1235
How to change the Apache default server properties ... 1236
Microsoft IIS Performance Counters ... 1236
Apache Performance Counters ... 1237
Network Virtualization Integration ... 1237
Installation ... 1238
Network Virtualization Locations ... 1238
NV Insights Report ... 1239
Considerations for Working with NV Insights ... 1239
Excluding Machines from Network Virtualization ... 1240
How to Run a Scenario with Network Virtualization ... 1240
Virtual Locations Settings Dialog Box ... 1243
User interface elements ... 1245
Per Group versus Per Load Generator ... 1246
Troubleshooting and Limitations for Network Virtualization ... 1246
Service Virtualization Integration ... 1247
Workflow ... 1248
Virtual Service Locks ... 1250
Virtual Service Modes ... 1250
Use Virtual Services in a Scenario ... 1250
Prepare the environment ... 1250
Add virtual services to a scenario ... 1251
Manage and monitor virtual services during the test ... 1252
Service Virtualization Setup Dialog Box ... 1253
Service Virtualization Runtime Dialog Box ... 1254
Service Virtualization Monitors ... 1255
J2EE/.NET Diagnostics ... 1257
How to Install the LoadRunner J2EE/.NET Diagnostics Add-in ... 1258
How to Configure a LoadRunner Scenario to use J2EE/.NET Diagnostics ... 1259

How to View J2EE/.NET Diagnostics Data in LoadRunner During a Scenario Run ....1260
How to View Offline J2EE/.NET Diagnostics Results ....1260
Diagnostics for J2EE/.NET Setup Dialog Box ....1261
Diagnostics Distribution Dialog Box ....1261
J2EE/.NET Configuration Dialog Box ....1263
Troubleshooting and Limitations for Controller ....1265
Vusers do not stop with scenario ....1265
Secure Communication with TLS (SSL) ....1265
Linux Machine Issues ....1265
Shellshock Vulnerability ....1266
Scenario run automation ....1267

Analysis ....1268
Introducing Analysis ....1269
Results Overview ....1269
Analysis Toolbars ....1270
Common Toolbar ....1270
Graph Toolbar ....1271
Analysis API ....1272
Workflow ....1272
Analysis Basics ....1273
Creating Analysis Sessions ....1273
Starting Analysis ....1273
Collating Execution Results ....1273
Session Explorer Window ....1273
Analysis Window Layouts ....1274
Open Windows ....1275
Lock/Unlock the Layout of the Screen ....1275
Restore the Window Placement to the Default Layout ....1275
Restore the Window Placement to the Classic Layout ....1275
Reposition and Dock Windows ....1275
Using Auto Hide ....1275
Printing Graphs or Reports ....1276
Configuring Analysis ....1277
Summary Data Versus Complete Data ....1277
Importing Data Directly from the Analysis Machine ....1277
Importing Data from the SQL Server ....1277
Importing Data from the Analysis Machine ....1278
How to Configure Settings for Analyzing Load Test Results ....1279
General Tab (Options Dialog Box) ....1279
Result Collection Tab (Options Dialog Box) ....1282
Data Aggregation Configuration Dialog Box (Result Collection Tab) ....1285
Database Tab (Options Dialog Box) ....1286

Main Area ... 1287
Server Details Area ... 1289
Shared Folder Details Area ... 1289
Advanced Options Dialog Box (Database Tab) ... 1290
Web Page Diagnostics Tab (Options Dialog Box) ... 1291
Session Information Dialog Box (Options Dialog Box) ... 1292
Viewing Load Test Scenario Information ... 1294
Viewing Load Test Scenario Information ... 1294
Configuring Controller Output Messages Settings ... 1295
Controller Output Messages Window ... 1296
Summary Tab ... 1296
Filtered Tab ... 1298
Scenario Runtime Settings Dialog Box ... 1300
Defining Service Level Agreements ... 1300
Service Level Agreements Overview ... 1300
Tracking Period ... 1301
Define Service Level Agreements ... 1301
Define Service Level Agreements - Use-Case Scenario ... 1302
Service Level Agreement Pane ... 1304
Advanced Options Dialog Box (Service Level Agreement Pane) ... 1305
Goal Details Dialog Box (Service Level Agreement Pane) ... 1306
Service Level Agreement Wizard ... 1307
Select a Measurement Page ... 1307
Select Transactions Page ... 1308
Set Load Criteria Page ... 1309
Set Percentile Threshold Values Page ... 1310
Set Threshold Values Page (Goal Per Time Interval) ... 1311
Set Threshold Values Page (Goal Per Whole Run) ... 1312
Set Up Graphs ... 1313
Configuring Graph Display ... 1313
Customize the Analysis Display ... 1313
Display Options Dialog Box ... 1314
Editing Main Chart Dialog Box (Display Options Dialog Box) ... 1315
Series Tab (Editing MainChart Dialog Box) ... 1316
Chart Tab (Editing MainChart Dialog Box) ... 1316
Export tab ... 1317
Print tab ... 1317
Legend Window ... 1317
Legend Toolbar ... 1318
Legend grid shortcut menu ... 1319
Measurement Description Dialog Box ... 1320
Measurement Options Dialog Box ... 1321
Legend Columns Options Dialog Box ... 1322

Apply/Edit Template Dialog Box ... 1323
Color Palettes ... 1325
Color Palette Dialog Box ... 1325
Working with Analysis Graph Data ... 1328
Determining a Point's Coordinates ... 1328
Drilling Down in a Graph ... 1328
Changing the Granularity of the Data ... 1330
Viewing Measurement Trends ... 1331
Auto Correlating Measurements ... 1332
Viewing Raw Data ... 1333
Getting more information from Graph Data ... 1333
Determine a point's coordinates ... 1333
Drill down in a graph ... 1333
Filter the data ... 1333
Change the granularity of the data ... 1334
View measurement trends ... 1334
Auto correlate measurements ... 1334
Save graph as image ... 1335
Drill Down Options Dialog Box ... 1335
Auto Correlate Dialog Box ... 1336
Time Range Tab ... 1337
Correlation Options tab ... 1338
Graph/Raw Data View Table ... 1339
Graph Properties Pane ... 1340
Filtering and Sorting Graph Data ... 1342
Filtering Graph Data Overview ... 1342
Sorting Graph Data Overview ... 1342
Filter Conditions ... 1342
Common Filter Condition Options ... 1343
Vuser Graphs ... 1343
Error Graphs ... 1343
Transaction Graphs ... 1344
Web Resource Graphs ... 1344
Web Page Diagnostics Graphs ... 1344
User Defined Data Point Graphs ... 1345
System Resources Graphs ... 1346
Network Monitor Graphs ... 1346
Web Server Resource Graphs ... 1346
Web Application Server Resource Graphs ... 1347
Database Server Resource Graphs ... 1347
J2EE & .NET Diagnostics Graphs ... 1347
Application Component Graphs ... 1348
Application Deployment Graphs ... 1349

Middleware Performance Graphs 1349
Infrastructure Resource Graphs 1349
External Monitor Graphs 1349
Custom Filter Dialog Box 1350
Filter Dialog Boxes 1351
Filter Builder Dialog Box 1353
Hierarchical Path Dialog Box 1354
Scenario Elapsed Time Dialog Box 1355
Set Dimension Information Dialog Box 1355
Vuser ID Dialog Box 1356
Cross Result and Merged Graphs 1357
Cross Result Graphs Overview 1358
Merging Types Overview 1359
Overlay 1359
Tile 1359
Correlate 1360
Generating Cross Results Graphs 1361
Generating Merged Graphs 1361
Merge Graphs Dialog Box 1362
Working with Application Lifecycle Management 1362
Managing Results Using ALM - Overview 1362
How to Connect to ALM from Analysis 1363
How to Work with Results in ALM - Without Performance Center 1363
How to Work with Results in ALM - With Performance Center 1364
Open an existing Analysis Session file 1364
Open raw data and create a new Analysis session 1365
Save the changes to the Analysis session file 1366
Save the Analysis session file to a new ALM location 1366
Integration Methods - TestPlan or TestLab 1367
How to Upload a Report to ALM with Performance Center 1367
ALM Connection Dialog Box 1368
Upload Report to Test Lab Dialog Box 1369
Analysis Graphs 1370
Open a New Graph Dialog Box 1371
AppDynamics Graph 1372
Application Component Graphs 1373
COM+ Average Response Time Graph 1373
COM+ Breakdown Graph 1375
COM+ Call Count Distribution Graph 1376
COM+ Call Count Graph 1378
COM+ Call Count Per Second Graph 1379
COM+ Total Operation Time Distribution Graph 1380
COM+ Total Operation Time Graph 1381

Microsoft COM+ Graph ... 1383
Authentication Metrics ... 1384
Application Event ... 1384
Thread Event ... 1384
Transaction Events ... 1385
Object Events ... 1385
Method Events ... 1386
.NET Average Response Time Graph ... 1386
.NET Breakdown Graph ... 1387
.NET Call Count Distribution Graph ... 1389
.NET Call Count Graph ... 1389
.NET Call Count per Second Graph ... 1390
.NET Resources Graph ... 1391
Application Level ... 1392
Assembly Level ... 1394
Class Level ... 1394
Method Level ... 1394
.NET Total Operation Time Distribution Graph ... 1394
.NET Total Operation Time Graph ... 1395
Application Deployment Solutions Graphs ... 1396
Citrix Measurements ... 1396
Non-Virtual Counters ... 1397
Virtual Channel Counters ... 1399
Citrix Server Graph ... 1401
Database Server Resource Graphs ... 1402
Oracle Server Monitoring Measurements ... 1402
SQL Server Default Counters ... 1404
Oracle Graph ... 1405
SQL Server Graph ... 1405
Dynatrace Graph ... 1406
Error Graphs ... 1406
Errors per Second (by Description) Graph ... 1407
Errors per Second Graph ... 1407
Error Statistics (by Description) Graph ... 1408
Error Statistics Graph ... 1409
Total Errors Graph ... 1410
Total Errors per Second Graph ... 1411
Flex Graphs ... 1412
Flex RTMP Throughput Graph ... 1412
Flex RTMP Other Statistics Graph ... 1413
Flex RTMP Connections Graph ... 1413
Flex Streaming Delivery Graph ... 1414
Flex Average Buffering Time Graph ... 1415

Infrastructure Resources Graphs ... 1416
Network Client Measurements ... 1416
Network Client Graph ... 1417
J2EE and .NET Diagnostics Graphs ... 1417
J2EE and .NET Diagnostics Graphs Overview ... 1417
How to Enable Diagnostics for J2EE and .NET ... 1418
Viewing J2EE to SAP R3 Remote Calls ... 1418
View RFC function information in a graph display ... 1418
View RFC function information in the Chain Of Calls window ... 1419
J2EE and .NET Diagnostics Data ... 1420
Example Transaction Breakdown ... 1420
Transaction Level ... 1420
Layer Level ... 1421
Class Level ... 1422
Method/Query Level ... 1423
Cross VM Analysis ... 1424
Descriptions of the measurements ... 1425
Using the J2EE and .NET Breakdown Options ... 1425
Viewing Chain of Calls and Call Stack Statistics ... 1427
The Chain of Calls window ... 1427
Icons in the Chain of Calls window ... 1428
Columns in the Chain of Calls window ... 1429
Columns in the Call Stack Statistics window ... 1430
Graph Filter Properties ... 1431
J2EE and .NET Diagnostics Graphs Summary Report ... 1432
J2EE/.NET - Average Method Response Time in Transactions Graph ... 1433
J2EE/.NET - Average Number of Exceptions in Transactions Graph ... 1434
J2EE/.NET - Average Number of Exceptions on Server Graph ... 1435
J2EE/.NET - Average Number of Timeouts in Transactions Graph ... 1435
J2EE/.NET - Average Number of Timeouts on Server Graph ... 1436
J2EE/.NET - Average Server Method Response Time Graph ... 1437
J2EE/.NET - Method Calls per Second in Transactions Graph ... 1437
J2EE/.NET - Probes Metrics Graph ... 1438
J2EE/.NET - Server Methods Calls per Second Graph ... 1439
J2EE/.NET - Server Requests per Second Graph ... 1440
J2EE/.NET - Server Request Response Time Graph ... 1441
J2EE/.NET - Server Request Time Spent in Element Graph ... 1442
J2EE/.NET - Transactions per Second Graph ... 1443
J2EE/.NET - Transaction Response Time Server Side Graph ... 1444
J2EE/.NET - Transaction Time Spent in Element Graph ... 1445
Graph UI Elements ... 1445
Graph data display filters ... 1446
JMeter Analysis Graphs ... 1446

Set up JMeter Analysis graphs ... 1447
Available JMeter graphs ... 1447
Middleware Performance Graphs ... 1447
IBM WebSphere MQ Counters ... 1448
Queue Performance Counters ... 1448
Channel Performance Counters ... 1448
IBM WebSphere MQ Graph ... 1450
MQTT Statistics Graphs ... 1450
Publish/Received Message Rate graphs ... 1451
Publish/Received Throughput graphs ... 1451
In-flight Messages graph ... 1451
Average Publish Duration graph ... 1451
Active Connection graph ... 1451
Average Connect Duration graph ... 1452
Connect Statistics graph ... 1452
Network Monitor Graphs ... 1452
Network Monitor Graphs Overview ... 1452
Network Delay Time Graph ... 1453
Network Segment Delay Graph ... 1454
Network Sub-Path Time Graph ... 1455
Network Virtualization Graphs ... 1456
Packet Loss Graph ... 1456
Average Latency Graph ... 1458
Average Bandwidth Utilization Graph ... 1460
Average Throughput Graph ... 1461
Total Throughput Graph ... 1462
New Relic Graph ... 1464
Service Virtualization Graphs ... 1465
Service Virtualization Operations Graph ... 1465
Service Virtualization Services Graph ... 1466
Streaming State Statistics Graphs ... 1467
System Resource Graphs ... 1468
Server Resources Performance Counters ... 1468
UNIX Resources Default Measurements ... 1468
Windows Resources Default Measurements ... 1469
Server Resources Graph ... 1471
Host Resources Graph ... 1472
SNMP Resources Graph ... 1473
UNIX Resources Graph ... 1474
Windows Resources Graph ... 1475
Transaction Graphs ... 1476
Transaction Breakdown Tree ... 1476
Average Transaction Response Time Graph ... 1477

Transactions per Second Graph ... 1479
Total Transactions per Second Graph ... 1479
Total Passed Transactions Graph ... 1480
Transaction Performance Summary Graph ... 1481
Transaction Response Time (Distribution) Graph ... 1482
Transaction Response Time (Percentile) Graph ... 1483
Transaction Response Time (Under Load) Graph ... 1484
Transaction Response Time by Location Graph ... 1485
Transaction Summary Graph ... 1486
TruClient - Native Mobile Graphs ... 1487
TruClient CPU Utilization Percentage Graph ... 1487
TruClient Free Memory In Device Graph ... 1487
TruClient Memory Consumed by Application Graph ... 1488
User-Defined Data Point Graphs ... 1489
User-Defined Data Point Graphs Overview ... 1489
Data Points (Average) Graph ... 1489
Data Points (Sum) Graph ... 1490
Vuser Graphs ... 1491
Rendezvous Graph (Vuser Graphs) ... 1491
Running Vusers Graph ... 1492
Vuser Summary Graph ... 1493
Web Page Diagnostics Graphs ... 1494
Web Page Diagnostics Tree View Overview ... 1494
Web Page Diagnostics Graphs Overview ... 1494
How to View the Breakdown of a Transaction ... 1496
Web Page Diagnostics Content Icons ... 1497
Web Page Diagnostics Graph ... 1498
Examples ... 1498
Page Component Breakdown Graph ... 1500
Page Component Breakdown (Over Time) Graph ... 1501
Page Download Time Breakdown Graph ... 1502
Page Download Time Breakdown (Over Time) Graph ... 1504
Page Download Time Breakdown Graph Breakdown Options ... 1506
Time to First Buffer Breakdown Graph ... 1507
Time to First Buffer Breakdown (Over Time) Graph ... 1508
Client Side Breakdown (Over Time) Graph ... 1511
Client Side Java Script Breakdown (Over Time) Graph ... 1512
Downloaded Component Size Graph ... 1513
Uploaded Component Size Graph ... 1514
Web Resources Graphs ... 1515
Web Resources Graphs Overview ... 1515
Hits per Second Graph ... 1516
Throughput Graph ... 1517

HTTP Status Code Summary Graph ... 1518
HTTP Status Codes ... 1519
HTTP Responses per Second Graph ... 1521
Pages Downloaded per Second Graph ... 1522
Retries per Second Graph ... 1523
Retries Summary Graph ... 1524
Connections Graph ... 1525
Connections per Second Graph ... 1526
SSLs per Second Graph ... 1527
Web Server Resource Graphs ... 1528
Web Server Resource Graphs Overview ... 1528
Apache Server Measurements ... 1528
IIS Server Measurements ... 1529
Apache Server Graph ... 1530
Microsoft Information Internet Server (IIS) Graph ... 1530
Web Application Server Resource Graphs ... 1531
Microsoft Active Server Pages (ASP) Graph ... 1532
Microsoft Active Server Pages (ASP) Measurements ... 1532
WebSocket Statistics Graphs ... 1533
Analysis Reports ... 1533
Understanding Analysis Reports ... 1533
Analysis Reports Overview ... 1534
Analyze Transaction Settings Dialog Box ... 1534
Analyze Transactions Dialog Box ... 1536
New Report Dialog Box ... 1538
Analysis Report Templates ... 1539
Report Templates Overview ... 1539
Report Templates Dialog Box ... 1539
Report Templates - General Tab ... 1541
Report Templates - Format Tab ... 1542
Report Templates - Content Tab ... 1544
Analysis Report Types ... 1546
Summary Report Overview ... 1546
Analysis Summary Report ... 1547
Summary Report with No SLA ... 1547
Summary Report with SLA ... 1548
Summary reports for Cross Result Graphs ... 1550
HTML Reports ... 1551
SLA Reports ... 1552
Transaction Analysis Report ... 1553
Importing Data ... 1554
Import Data Tool Overview ... 1554
Import data into your Analysis session ... 1555

Define Custom File Formats ... 1556
Supported File Types ... 1556
NT Performance Monitor (.csv) ... 1556
Windows Performance Monitor (.csv) ... 1557
Standard Comma Separated File (.csv) ... 1557
Master-Detail Comma Separated File (.csv) ... 1557
Microsoft Excel File (.xls) ... 1558
Master-Detail Microsoft Excel file (.xls) ... 1558
Advanced Settings Dialog Box (Import Data Dialog Box) ... 1558
Define External Format Dialog Box ... 1559
Mandatory tab ... 1560
Optional tab ... 1560
Import Data Dialog Box ... 1561
Troubleshooting and Limitations for Analysis ... 1562
General ... 1563
Graphs ... 1563
ALM Integration ... 1564
Microsoft SQL Server ... 1564

Additional Components ... 1566
Where to install additional components ... 1566
Standalone Applications ... 1570

Function Reference ... 1572

Send Us Feedback ... 1573

# Welcome to the LoadRunner User Guide

Welcome to LoadRunner, the Micro Focus solution for application performance testing. LoadRunner stresses your entire application to isolate and identify potential client, network, and server bottlenecks.

LoadRunner includes:

- **VuGen.** Micro Focus's tool for creating Vuser scripts. You use VuGen to develop a Vuser script by recording a user performing typical business processes. The scripts let you emulate real-life situations.
- **Controller.** Allows you to easily and effectively control all the Vusers from a single point of control and monitor the scenario performance during test execution.
- **Analysis.** You use Analysis after running a load test scenario in the LoadRunner Controller or Performance Center. The Analysis graphs help you determine system performance and provide information about transactions and Vusers. You can compare multiple graphs by combining results from several load test scenarios or merging several graphs into one.

You can access various additional documentation for LoadRunner from the Windows Start menu: **Micro Focus > LoadRunner > Documentation**.

# What's New

Check out below the new features and improvements in LoadRunner 12.60.

## TruWeb

TruWeb is a new protocol offering, focusing on the HTTP (transport) level. It provides a lightweight, scalable, cross-platform solution using a brand new JavaScript SDK and engine. See TruWeb Protocol.

> **Note:** The TruWeb protocol is provided as tech preview version.

## VuGen

### Protocol enhancements

| | |
|---|---|
| **Web - HTTP/HTML** | The Web - HTTP/HTML protocol now includes:<br>• A runtime setting to enable/disable HTTP/2 features. See the "HTTP" on page 311 section for runtime settings.<br>• An argument for **web_set_certificate_ex** that looks up a certificate in the Windows Certificate Store, using a more reliable thumbprint instead of the index. See web_set_certificate_ex in the Function Reference. |

| | |
|---|---|
| **Oracle NCA protocol** | The Oracle NCA protocol has a new runtime setting that enables connecting to one or multiple IP addresses. See "Oracle NCA Protocol" on page 628. |
| **PCoIP protocol** | We have enhanced desktop pool support with the ability to sync on bitmap timeout per step. See pcoip_sync_on_bitmap_ex in the Function Reference. |

### TruClient

All information is available in the TruClient Help Center.

| | |
|---|---|
| **TruClient Browser/Firefox browser** | TruClient Browser, first made available in version 12.56, is now fully supported and equivalent to Firefox version 58.0.2. TruClient browser has replaced the Firefox browser. See Differences Between Supported Browsers. |
| **Debug code** | When converting actions to code, you can now benefit from expanded debugging functionality using an Inspector Panel, enabling you to do a detailed review of variable values in the code. See Replay and enhance the converted script.<br><br>**Note:** The converting actions to code feature is provided as tech preview version. |
| **Windows 10** | This release includes improved support for using the TruClient IE browser on Windows 10. See TruClient for IE Browser. |
| **Interactive guidance** | TruClient now includes interactive walkthroughs and guidance, providing help within the TruClient application with onboarding, product tours, and more. See Tutorials and Help. |

## Controller and Analysis

| | |
|---|---|
| **Network Virtualization** | Network Virtualization usage is now included with your LoadRunner protocol licenses; network emulation and NV Insights do not require any additional license purchase. See "License bundles" on page 926. |
| **Analysis graphs** | You can now save a graph as an image file directly from the graph's right-click menu. See "Save graph as image" on page 1335. |

| | |
|---|---|
| **Command Line Interface tool** | You can now automate scenario runs using the Command Line Interface tool by invoking an XML input file or a LoadRunner scenario (.lrs) file.<br>The XML input can be constructed prior to running scenarios, allowing for fully DevOps-oriented flows including:<br>• Pulling scripts from any source control<br>• Dynamically allocating Dockerized load generators<br>• Running scenarios from any CI tool<br>See "Running Scenarios using the Command Line Interface (CLI) Tool" on page 1114 |
| **LoadRunner Raw Results Exporter** | Use the new **LoadRunner Raw Results Exporter** tool to export scenario results to JSON or to InfluxDB. See "LoadRunner Raw Results Exporter" on page 1133. |
| **Windows Dockerized Load Generator** | A Dockerized load generator image (for some protocols) is now available for Windows platforms, in addition to Linux. See "Configure Dockerized Load Generators" on page 985.<br>**Note:** The Windows Dockerized load generator image is provided as tech preview version. |

# Technologies and platforms

| | |
|---|---|
| **Prerequisites** | The following are no longer prerequisites for LoadRunner:<br>• Microsoft Visual C++ Redistributable 2010<br>• Microsoft Visual C++ Redistributable 2012<br>• Microsoft Visual C++ Redistributable 2013 |
| **Java** | Support added for Java 9 and 10<br>**Default:** Java 10 |
| **OpenSSL** | Support added for OpenSSL 1.1.0g |
| **SAP GUI protocol** | Support added for the following:<br>• SAP GUI 7.5<br>• Ability to record and replay SAP GUI scripts on Windows 10/Windows Server 2016 |
| **MQTT protocol** | Support added for UTF-8 |

### See also:

- What's New in LoadRunner 12.56 and 12.57
- What's New in LoadRunner 12.55
- What's New in LoadRunner 12.53

# Virtual User Generator

The Virtual User Generator (VuGen) enables you to record and develop scripts for load testing with Micro Focus products.

To learn more about VuGen, see "Introducing Virtual User Generator" below.

# Introducing Virtual User Generator

Virtual User Generator (VuGen) is a primary tool for creating testing scripts that emulate the behavior of real users on your system.

## Overview

When testing or monitoring an environment, you need to emulate the true behavior of users on your system. Micro Focus testing tools emulate an environment in which users concurrently work on, or access your system.

To perform this emulation, the human is replaced with a virtual user, or a *Vuser*. The actions that a Vuser performs are typically recorded in a *Vuser script*.

You use VuGen to develop a Vuser script by recording a user performing typical business processes. The Vuser scripts let you emulate real-life situations.

## Use VuGen scripts to test your environment

You use the scripts created with VuGen in conjunction with other Micro Focus products, as follows:

- **LoadRunner**, a tool for performance testing, stresses your entire application to isolate and identify potential client, network, and server bottlenecks. Learn more and get a free trial here: LoadRunner Home.
- **StormRunner Load (SRL)** is a cloud-based load testing solution that allows Agile development teams to create cost effective, extremely scalable tests. If your application under test is accessible from the internet, you can use SRL to store, manage, and execute your scripts in the cloud, enabling you to rapidly scale to 1 million Vusers or more. Learn more and get a free trial here: StormRunner Load Home.
- **Performance Center** implements the capabilities of LoadRunner on an enterprise level. Learn more and get a free trial here: Performance Center Home.
- **Application Performance Management (APM)** helps you optimize the management and availability of business applications and systems in production. VuGen is used in conjunction with the following APM components:
  - **Business Process Monitor (BPM)** software is a synthetic monitoring solution that simulates business transactions—whether or not real users are active. You use VuGen to create scripts for BPM, in order to reuse assets in testing and production environments. Learn more here: Business Process Monitor Home.

- **Real User Monitor (RUM)** software monitors application performance and availability on business critical application services, for all users. You use VuGen to capture and replay user sessions, and to create test scripts that reflect real user behavior. Learn more and get a free trial here: Real User Monitor Home.
- **AppPulse Active** enables you to monitor applications across traditional, mobile, virtualized, and cloud environments. You can use VuGen to record scripts for AppPulse Active, across a range of protocols. The scripts are imported into AppPulse Active and used for availability and performance monitoring, by automatically reproducing the activity of the user. Learn more and get a free trial here: AppPulse Active Home.

## Want to learn more?

The LoadRunner Tutorial is a self-paced online and printable guide, designed to lead you through the process of developing Vuser scripts with VuGen, as well as familiarizing you with the LoadRunner testing environment.

## VuGen updates

To check for VuGen version updates, use the **Check for Updates** feature available from VuGen's **Help** menu.

Alternatively, you can configure VuGen to notify you about updates automatically on startup. For details, see Update Messages.

In addition, the VuGen start page displays periodic live updates about features, blog posts, webinars, and other information.

### See also:

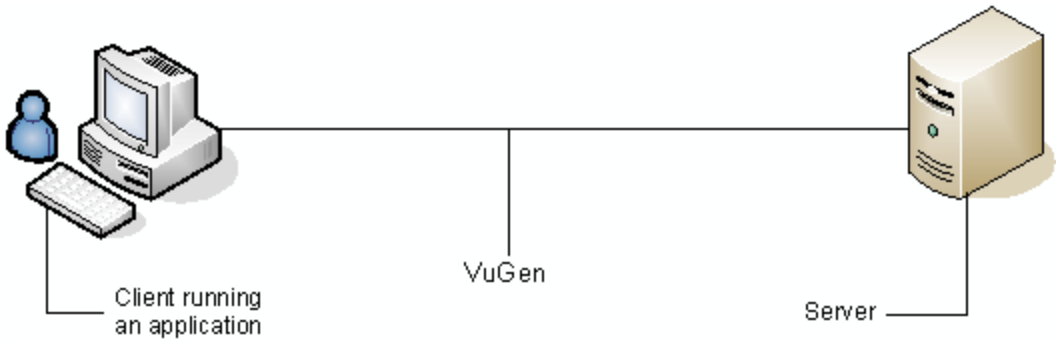
- "What's New" on page 52
- "VuGen Workflow" on page 136
- "Vusers" below

# Vusers

## Vuser Technology

You use VuGen to develop a Vuser script by recording a user performing typical business processes on a client application. VuGen records the actions that you perform during the recording session, recording only the activity between the client and the server.

During recording, VuGen monitors the client and traces all the requests sent to and received from the server.




After the recording, VuGen generates various functions that define the actions performed during the recording session. VuGen inserts these functions into the VuGen editor to create a basic Vuser script.

Instead of having to manually program the application's API function calls to the server, VuGen automatically generates functions that model and emulate real world situations.

VuGen not only records Vuser scripts, but also replays them. Replaying scripts from VuGen is useful for debugging. It enables you to determine how a Vuser script will run when it is executed as part of a larger test.

During playback, Vuser scripts communicate directly with the server by executing calls to the server API functions. When a Vuser communicates directly with a server, system resources are not required for the client interface. This lets you run a large number of Vusers simultaneously on a single workstation, and enables you to use only a few testing machines to emulate large server loads.

In addition, since Vuser scripts do not rely on client software, you can use Vusers to check server performance even before the user interface of the client software has been fully developed.

You can use the Vuser scripts in several Micro Focus products that incorporate scripts.

One option is to add them to a LoadRunner scenario using the Controller. While running the Vusers from the Controller, you gather information about the system's response. After the test run, you can view this information with the Analysis tool. For example, you can observe how a server behaves when one hundred Vusers simultaneously withdraw cash from a bank's ATM. For details, see "Introducing Controller" on page 916.

VuGen only records scripts on Windows platforms. However, a recorded Vuser script can be replayed on both Windows and Linux platforms.

You can also program Vuser scripts in your native programming application, such as MS Visual Studio. To access the API, install the appropriate IDE add-in provided in your LoadRunner installation package.

## Vuser Types

| Vuser Type | Description |
|---|---|
| Protocol Based Vusers | Vusers that run scripts developed in VuGen. VuGen supports most of the common communication protocols. For a complete list of the available Vuser protocols, see "VuGen Protocols" on page 460.<br><br>You can create a Vuser script using a single protocol or multiple ones. |
| Unit Test Based Vusers | Vusers running unit tests in the form of **.dll** or **.jar/.class** files, created in supported versions of Microsoft Visual Studio or Eclipse.<br><br>To create these tests, install the IDE for Developer add-in, available in the **Additional Components** folder of your LoadRunner installation package. |
| GUI Vusers | Vusers that perform functional testing on a user interface level. You create these tests using Micro Focus's Functional Testing software, such as UFT (Unified Functional Testing).<br><br>You can only run a single GUI Vuser on a Windows-based load generator. To run multiple GUI Vusers you can use Citrix sessions. For additional information on Windows-based GUI Vusers, see "Using Unified Functional Testing Tests in LoadRunner" on page 1122. |

### See also:

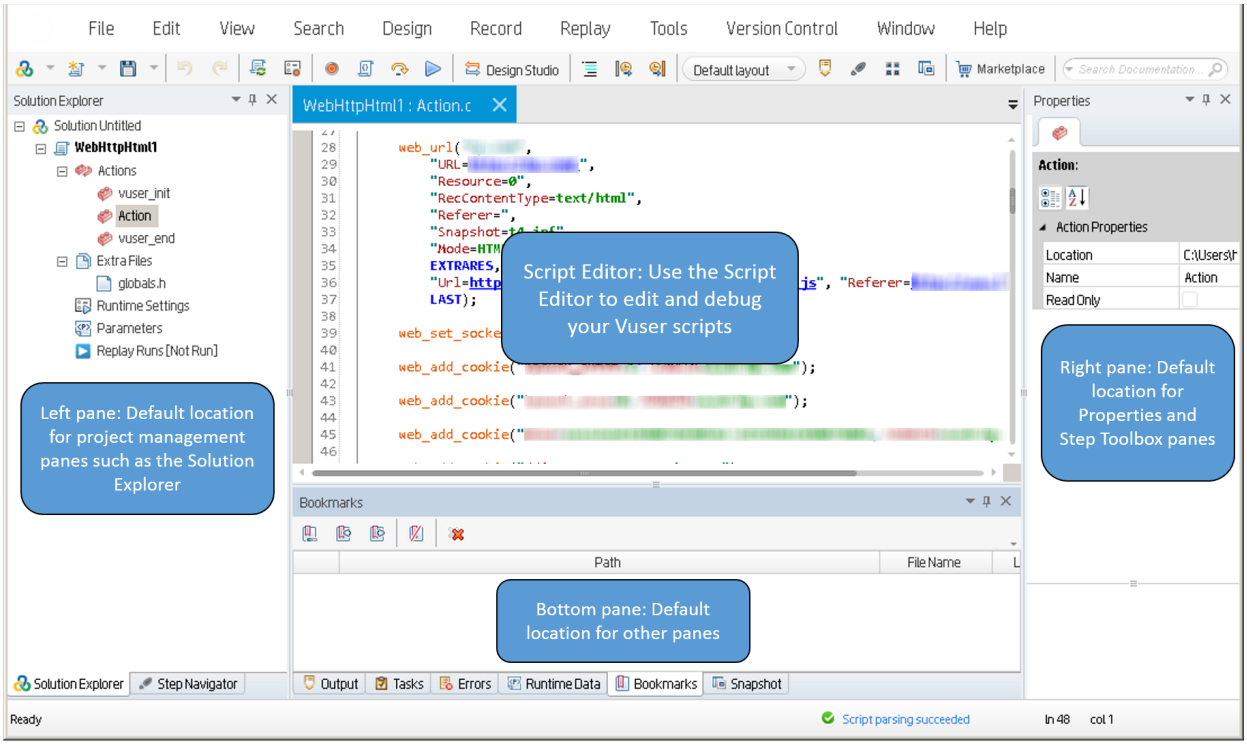
- "VuGen Workflow" on page 136

# VuGen User Interface

The VuGen editor and panes are the environment you will be working in while you record, replay, and debug a Vuser script.

## VuGen Workspace

The VuGen workspace enables you to record, edit and debug your Vuser script. The VuGen workspace is divided up as follows:




### Script Editor

VuGen's Editor enables you to edit recorded scripts and other supplementary files such as header files. You can open multiple files simultaneously, navigating tab by tab. The editor also supports multiple programming languages, code coloring, code folding (enables you to selectively hide and display sections of your code), code completion and tooltips for C scripts. For details, see "Editor Pane" on page 77.

### Project Management Panes

The project management panes include the Solution Explorer, the Step Navigator and the Outline pane. By default they all appear on the left side of the workspace. The Solution Explorer enables you to easily organize and navigate through script entities, enhancing the recording, replay and debug process. You can create a solution containing multiple scripts of different protocols related to a full-cycle business process. Each script entity includes extra files (such as header files), runtime settings, parameters, and replay runs. For details, see "Solution Explorer Pane" on page 69.

The Step Navigator enables you to navigate to a selected step in your script. If your script contains many steps, you can use the search box to search for matching text in the different parts of the steps. For details, see "Step Navigator Pane" on page 75.

### Window Panes

VuGen has a number of window panes which by default are displayed at the bottom of the workspace. Each window pane deals with one specific aspect of working with the script. For example, the Errors pane displays all errors in the script.

The following table describes each pane and provides a short use case scenario.

| Pane | Used For: | For Details: |
|---|---|---|
| **Bookmarks** | Specifying a location in a script so that you can easily find it later on for editing. | See "Bookmarks Pane" on page 84 |
| **Errors** | Displaying script errors, warnings and messages generated from script replay.<br>Creating custom filters for error messages. | See "Errors Pane" on page 95 |
| **Snapshot** | Displays server and client data associated with a specific step in a script. The format of the data is dependent on the protocol used for creating the script. | See "Snapshot Pane" on page 85 |
| **Data Grid** | Simplifying views of all recordsets associated with the script. Valid for specific protocols such as ODBC. Contains either sent or received data.<br>Parameterizing and manipulating data displayed in the data grid. | |
| **Tasks** | Adding, editing or searching for tasks related to a script or solution. | See "Tasks Pane" on page 97 |
| **Thumbnail** | Following the business process that the script has recorded. | See "Thumbnail Explorer" on page 94 |
| **Output** | Event log from different operations in VuGen such as code generation, replay, and recording. | See "Output Pane" on page 100 |
| **Breakpoints** | Managing breakpoints in Vuser scripts to help debug the scripts. | See "Breakpoints Pane" on page 101 |
| **Watch** | Monitoring variables and expressions while a script runs, and is in the **Paused** state. | See "Watch Pane" on page 104. |

| Pane | Used For: | For Details: |
|---|---|---|
| **Call Stack** | Viewing information about the methods and functions that are currently on the call stack of your script, or the context in which the run session was paused. | See "Call Stack Pane" on page 104 |

### Properties Pane and Steps Toolbox

The Properties pane and Steps Toolbox are displayed on the right side of the workspace.

The Properties pane displays the selected object's properties, such as the object's location. Each object has its own specific properties list. You can sort the property list according to category or by alphabetical order.

The Steps Toolbox displays a list of API functions which you can drag and drop into your script. The API functions are divided into categories. For details on each API function see the *VuGen Function Reference*. For more details on the Steps Toolbox, see "Steps Toolbox Pane" on page 83.

### Standard Layouts

VuGen has many different window panes which you may want to display or hide based on what you are currently doing. You can also move the panes around the workspace, in order to customize the workspace layout. VuGen comes with a set of standard layouts:

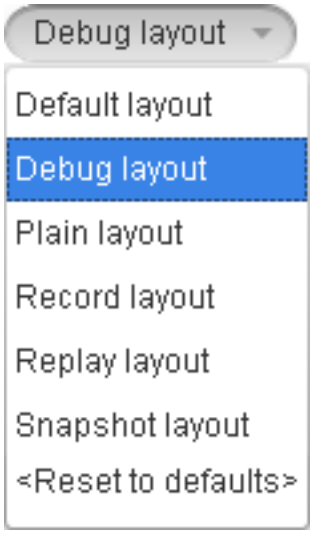
- Default
- Debug
- Plain
- Record
- Replay
- Snapshot

Each layout is designed to enhance a specific phase of the Vuser script development process. For example, the Replay layout includes the panes that are most useful when you run a Vuser script: **Errors**, **Call Stack**, **Watch**, **Breakpoints**, **Output**, and **Runtime Data**.

VuGen automatically uses specific layouts during specific phases of the script development process. For example, the Record layout is used while you record a script, and the Replay layout is used when you replay a script.

The VuGen toolbar displays the layout that is currently used:

. To change the layout, click the **Layout** drop-down and select the required layout from the list of layouts, as shown below.




For details on how to customize VuGen layouts, see "How to Modify the VuGen Layout" below.

**Customizing your Workspace**

You cannot add or delete a standard layout. However, you can modify most of VuGen's standard layouts to meet your specific requirements. When you modify a layout, you can add, move and resize zones, select which panes to include in each zone, and specify which of these panes is displayed by default. For task related details, see "How to Modify the VuGen Layout" below. After you modify a standard layout, VuGen maintains that layout until you change the layout again or reset the default layouts.

> **Note:** VuGen does not save any changes that you make to the Plain layout.

### Restoring the layout defaults

On the VuGen toolbar, click the **Layout** drop-down, and select **Reset to Defaults**. VuGen resets all standard layouts to their default settings.

# How to Modify the VuGen Layout

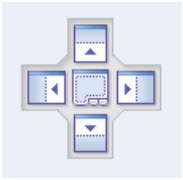
The VuGen window is composed of a number of zones. Each zone can contain a variety of panes, such as the Errors pane and the Snapshot pane. When more than one pane is included in a zone, the panes appear as tabs within the zone. This section describes how to customize and modify the zones and panes that appear in the VuGen window.

**Move a pane to a new zone**

You can move any VuGen pane to a new zone. The new zone can be either a portion of an existing zone, or it can occupy the entire left, right, top, or bottom of the VuGen window.

In the VuGen window, drag the title bar or tab of the pane that you want to move. (If the required pane is not displayed in the VuGen window, you can select it from the **View** menu.) As you drag the pane over the zones in the VuGen window, a complex marker is displayed in the center of the active zone and a simple marker appears on each edge of the VuGen window.

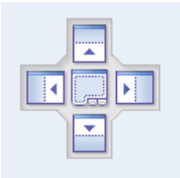
> **Note:** If you drag the title bar of a zone that contains multiple tabbed panes, all of the panes in the zone move to the new zone.

| Marker Type | Marker | Description |
|---|---|---|
| **Complex marker - Current zone** |  | Positions the selected pane in a new zone. The new zone is created in the top, bottom, left, or right of the **active zone**, according to the arrow marker selected when you release the mouse button. |
| **Simple marker - VuGen window** | | Positions the selected pane in a new zone. The new zone is created in the top, bottom, left, or right of the **VuGen window**, according to the arrow marker selected when you release the mouse button. |

### Move a pane to an existing zone

You can move any VuGen pane from one zone to another. When more than one pane is included in a zone, the panes appear as tabs within the zone.

1. In the VuGen window, drag the title bar or tab of the pane you want to move. (If the required pane is not displayed in the VuGen window, you can select it from the View menu). As you drag the pane over the zones in the VuGen window, a complex marker is displayed in the center of the active zone.

2. Locate the cursor over the center button of the complex marker. When you release the mouse button, the selected pane is added as a tabbed pane to the selected zone.
3. Repeat this procedure for each pane you want to move.

> **Note:** If you drag the title bar of a zone that contains multiple tabbed panes, then all the panes in the zone are moved to the selected zone.

### Float and dock panes

Docked panes are fixed in a set position within the VuGen window. For example, when you move a pane to a position indicated by a marker, the pane is docked in that position.

Floating panes are displayed on top of all other windows. Floating panes can be dragged to any position on your screen, even outside the VuGen window. Floating panes have their own title bars.

- To float a pane, right-click the title bar, and click **Float**. The pane opens on top of all the other windows and panes, with its own title bar.
- To dock a pane, double-click the title bar, or right-click the title bar and select **Dock as tabbed document**. The pane returns to its previous position in the VuGen window.

# Keyboard Shortcuts

This section lists the keyboard shortcuts available for the VuGen menus.

## File Menu

| | |
|---|---|
| New Script and Solution | Ctrl+N |
| Open > Script/Solution | Ctrl+O |
| Add > New Script | Ctrl+Shift+A |
| Add > Existing Script | Alt+Shift+A |
| Close > Document | Ctrl+F4 |
| Close > Solution | Ctrl+Shift+F4 |
| Save Script | Ctrl+S |
| Save All Scripts | Ctrl+Shift+S |
| Reload File | Ctrl+Shift+U |
| Print | Ctrl+P |
| Exit | Alt+F4 |

## Edit Menu

| | |
|---|---|
| Undo | Ctrl+Z |
| Redo | Ctrl+Y |
| Cut | Ctrl+X |
| Copy | Ctrl+C |
| Paste | Ctrl+V |
| Delete | Del |
| Select All | Ctrl+A |
| Format > Surround with | Ctrl+J |
| Format > Increase Indent | Tab |
| Format > Decrease Indent | Shift+Tab |
| Folding > Toggle fold | Ctrl+Shift+M |
| Folding > Toggle all folds | Ctrl+Shift+L |
| Folding > Show definitions only | Ctrl+Shift+P |
| Show Function Syntax | Ctrl+Shift+Space |
| Complete Word | Ctrl+Space |

## View Menu

| | |
|---|---|
| Solution Explorer | Ctrl+Alt+L |
| Search Results | Ctrl+Alt+R |
| Bookmarks | Ctrl+Alt+K |
| Steps Toolbox | Ctrl+Alt+B |
| Snapshot | Ctrl+Alt+P |
| Steps Navigator | Ctrl+Alt+S |
| Thumbnail Explorer | Ctrl+Alt+T |
| Properties | Ctrl+Alt+F4 |

| | |
|---|---|
| Output | Ctrl+Alt+O |
| Full Screen | Alt+Shift+Return |

## Search Menu

| | |
|---|---|
| Quick Find | Ctrl+F |
| Find Next | F3 |
| Find Next Selected | Ctrl+F3 |
| Find in Files | Ctrl+Shift+F |
| Quick Replace | Ctrl+H |
| Incremental Search<br>See "Incremental Search" on page 130 | Ctrl+E |
| Reverse Incremental Search<br>See "Incremental Search" on page 130 | Ctrl+Shift+E |
| Bookmarks > Toggle Bookmark | Ctrl+F2 |
| Bookmarks > Prev Bookmark | Shift+F2 |
| Bookmarks > Next Bookmark | F2 |
| Go To | Ctrl+G |

## Design Menu

| | |
|---|---|
| Action > Delete Action | Delete |
| Action > Rename Action | F2 |
| Insert in Script > New Step | Alt+Insert |
| Insert in Script > Start Transaction | Ctrl+T |
| Insert in Script > End Transaction | Ctrl+Shift+T |
| Insert in Script > Surround with Transaction | Ctrl+Shift+I |
| Insert in Script > REST API | Ctrl+Shift+W |
| Insert in Script > Comment | Ctrl+Alt+C |

| | |
|---|---|
| Parameters > Parameters List | Ctrl+L |
| Parameters > Create New Parameter | Ctrl+K |
| Parameters > Configure Parameter Delimiters | Ctrl+B |
| Design Studio | Ctrl+U |

## Record Menu

| | |
|---|---|
| Record | Ctrl+R |
| Regenerate Script | Ctrl+Shift+R |
| Recording Options | Ctrl+F7 |

## Replay Menu

| | |
|---|---|
| Run | F5 |
| Stop | Ctrl+F5 |
| Compile | Shift+F5 |
| Toggle Breakpoint | F9 |
| Continue Debugging | F5 |
| Run Step by Step | F10 |
| Runtime Settings | F4 |

## ALM Menu

| | |
|---|---|
| ALM Connection | Ctrl+Q |

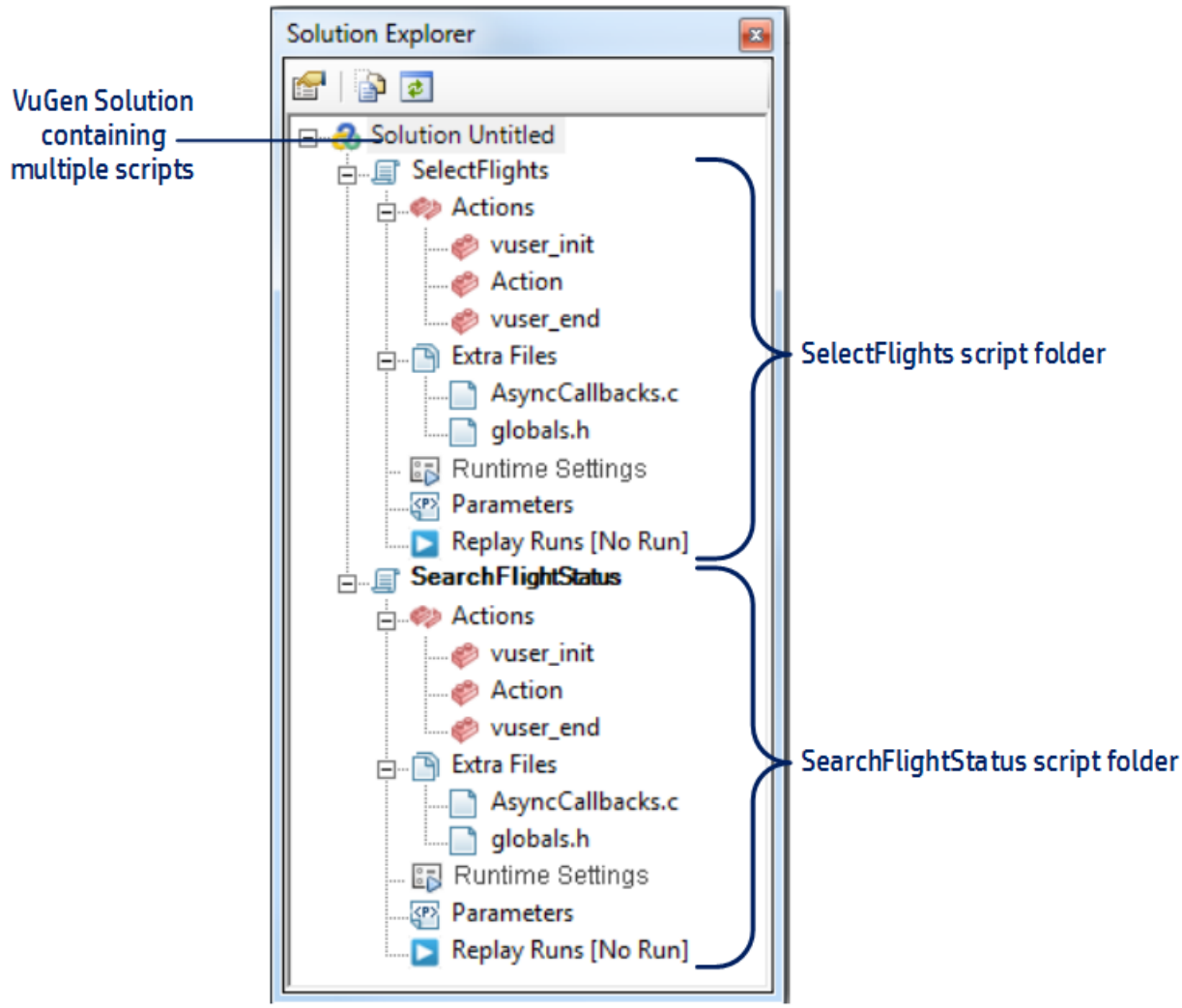
## Windows Menu

| | |
|---|---|
| Next Window | Ctrl+Tab |
| Prev Window | Ctrl+Shift+Tab |

# Solution Explorer Pane

The Solution Explorer enables you to manage your Vuser scripts. A solution contains Vuser scripts. Vuser scripts consist of script files, extra files (such as header files), runtime settings, parameters and replay runs. A solution can contain multiple scripts of different protocols.

The image below shows a Solution Explorer with two scripts.




| | |
|---|---|
| **To access** | Do one of the following:<br>• **View > Solution Explorer**<br>• Press **Ctrl + Alt + L** |

| | |
|---|---|
| **Important information** | • Solution Explorer is automatically displayed as part of the default layout.<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63.<br>• Other main interface panes such as Output, Error and Snapshot synchronize their displays based on your location in the Solution Explorer.<br>• You can double-click an asset to activate it in the editor area or right-click to examine quick operations available for that asset.<br>• You can bundle scripts in a solution. For example you can bundle scripts related to one business process.<br><br>**Note:** The solution explorer cannot be imported into any of the existing management tools such as ALM or Controller. |
| **See also** | • "VuGen User Interface" on page 59<br>• "Import Actions to a Script" on page 239 |

## Understanding the Solution Explorer

| Entity | Used for | Comments |
|---|---|---|
| Solution | Container for all script objects. | Give your solutions meaningful names, such as the name of the business process. The default solution name is "Untitled" |
| Scripts | Creating, editing and debugging scripts. | Click once on a script, or one of its assets, to change the focus to that script. VuGen applies any actions, such as clicking replay, to the script in focus.<br><br>When any part of a script is selected, the menu options, toolbar and window panes display functionality relevant to the script's protocol. For example, if the script in focus is recorded in Web HTTP/HTML , the **Recording Options** button is displayed on the toolbar. However, if the script in focus is recorded in TruClient, the **Develop Script** button is displayed on the toolbar.<br><br>Double-click the script's action to open it in the editor.<br><br>You can drag and drop scripts (`<scriptName>.usr`) from the file directory to the **Solution Explorer**. |

| Entity | Used for | Comments |
|---|---|---|
| Extra Files | Storing extra files that are used by the script. | The data contained in extra files can include:<br>• Common utility functions used by the script (for example, code)<br>You can add some types of extra utility function files to the parsing list to extend their functionality. For example, adding files enables "Go to Definition" functionality. See the "Options Dialog Box" on page 109.<br>• Definition of constants and variables used by the script (for example, code)<br>• Special assets used during script execution (such as .jpeg files)<br>• Data files manipulated by script code during script execution<br>• Additional files to be parsed. For details, see "Create and Open Vuser Scripts" on page 137.<br>The following are examples of valid file types that can be added as extra files:<br>.ws, .h,.c., dat, .ini, .vbs, .java, .js, .txt, .tux, .rec, .msc, .vdf, .xml, .xsl, .dtd, .html, .htm<br>You can drag and drop header files (`<headerFileName.h>` from your file directory. When you include files in the **Extra Files** node, these files are automatically included in a LoadRunner scenario.<br>You can edit extra files in the editor if the file type is included in **Tools > Options > Scripting Tab > Script Management**. Double click the extra file to open it in the Editor. For details on how to modify the list, see "Scripting Options Tab" on page 119. |
| Runtime Settings | Defining the way a Vuser script runs. | You can access runtime settings for a specific script from the Runtime Settings node in the **Solution Explorer > <Script> > Runtime Settings**. For details, see "Runtime Settings" on page 301. |
| Parameters | Creating and managing parameters. | You can access parameters for a specific script from the Parameters node in the **Solution Explorer > <Script> > Parameters**. For details, see "Parameterization Overview" on page 357. |

| Entity | Used for | Comments |
|---|---|---|
| Replay Runs | Enables you to access the Replay Summary Reports for each iteration in the replay. | |

## Solution Explorer Structure and Context Menu Options

The following table lists the context menu options when working in the Solution Explorer.

| UI Element and Description | Context Menu Options | Description |
|---|---|---|
| **<Solution>**<br>Container for scripts. | **Add New Script** | Creates a new script and adds it to the solution. For details, see "Create a New Script Dialog Box" on page 150. |
| | **Add Existing Script** | Adds an existing script to the solution. |
| | **Save All Scripts** | Saves the changes of all open scripts. |
| | **Close Solution** | Closes the current solution, along with its scripts, without saving the changes. |
| | **Save Solution As...** | Saves the solution with a new name or to a new location. |

| UI Element and Description | Context Menu Options | Description |
|---|---|---|
| **<Script>**<br>Container for script assets including scripts actions, extra files, runtime settings and parameters. | **Save Script** | Saves the current script. |
| | **Save Script As...** | Saves the script with a new name or new location, such as ALM. |
| | **Export to Template...** | Saves the script as a template. For details, see "Create and Open Vuser Script Templates" on page 153. |
| | **Remove Script** | Deletes a script from the solution or from the file directory, if you enable the check box. |
| | **Select file to compare** | Compares two files using the highlighted asset as the primary file. |
| | **Compare to <filename>** | Compares the primary file with the specified file. |
| | **Compare to external file...** | Compares a file to a file outside of the solution. This option sets the highlighted file as the primary file and opens Windows Explorer to enable you to select the secondary file. |
| | **Select Folder to Compare** | Compares one folder to another. The highlighted folder is the primary folder. |
| | **Compare to External Folder...** | Compares the selected folder and the primary folder. |
| | **Open Script Folder** | When you record, VuGen creates a series of configuration, data, and source code files. These files contain Vuser runtime and setup information. VuGen saves these files together with the script.<br>You can open the folder on the local disk where the script is saved. For scripts that are saved on a different storage location (such as ALM), this option opens the temporary folder on the local disk. |

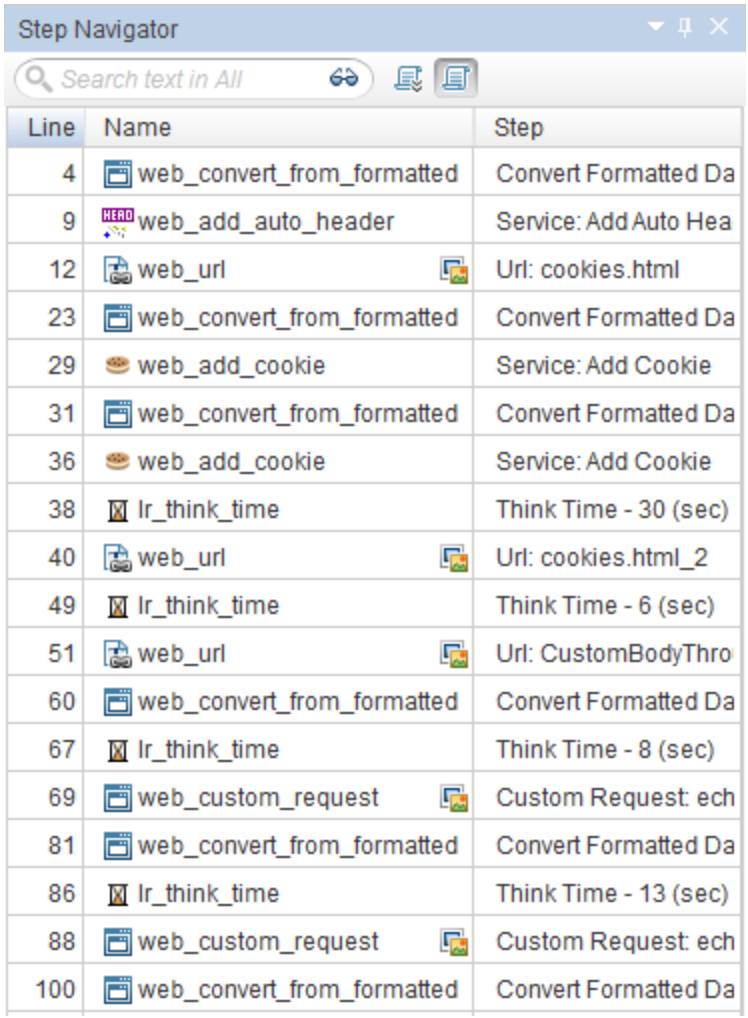
| UI Element and Description | Context Menu Options | Description |
|---|---|---|
| **<Actions>**<br>Container for individual actions including the default actions:<br>• vuser_init<br>• action<br>• vuser_end | **Create New Action** | Adds a new script action block to your script. |
| | **Import Action** | Imports an action from an existing script. |
| **Extra Files**<br>Container for extra files associated with your script. You can access these extra files directly from VuGen. | **Create New File** | Creates a new file, and adds it to the **Extra Files** node of your script. |
| | **Add Files to Script** | Adds files to the **Extra Files** node of your script. The files that you add must already exist. |
| | **Add Files from Folders and Sub-folders** | Adds the contents of folders and sub-folders to the **Extra Files** node of your script. |
| **Runtime Settings** | | Opens the runtime settings dialog box.<br>For details, see "Runtime Settings" on page 301. |
| **Parameters**<br>Enables you to create, edit, and list parameters associated with your script. | **Parameters list** | Opens the Parameter List dialog box. For details, see "Parameterization" on page 357. |
| | **Create new parameter** | Opens the Select or Create Parameter dialog box. |
| | **Edit parameter** | Opens the Parameter Properties dialog box. |
| | **Configure parameter delimiter** | Opens the Parameter Delimiters Configuration dialog box. |
| **Replay Runs**<br>Enables you to display the Replay Summary report. | **Open Replay Summary** | Opens the Replay Summary in the Editor for selected iteration. |

**See also:**

- "VuGen Workflow" on page 136

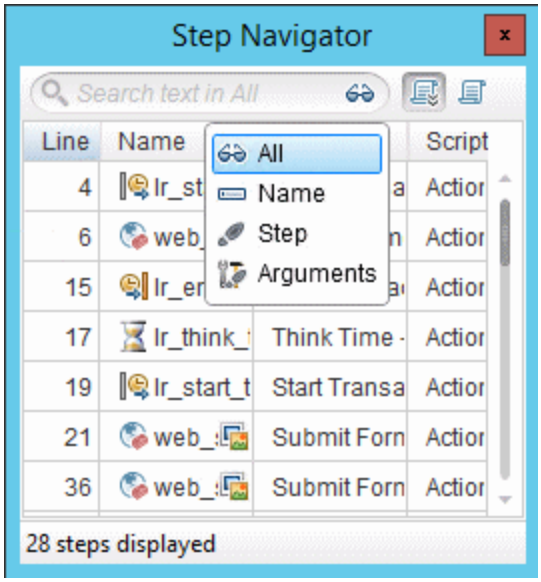
# Step Navigator Pane

The Step Navigator is a table view of the API function calls in the script. In addition, the Step Navigator pane enables you to navigate to a selected step in your script.




| Line | Name | Step |
|---|---|---|
| 4 | web_convert_from_formatted | Convert Formatted Da |
| 9 | web_add_auto_header | Service: Add Auto Hea |
| 12 | web_url | Url: cookies.html |
| 23 | web_convert_from_formatted | Convert Formatted Da |
| 29 | web_add_cookie | Service: Add Cookie |
| 31 | web_convert_from_formatted | Convert Formatted Da |
| 36 | web_add_cookie | Service: Add Cookie |
| 38 | lr_think_time | Think Time - 30 (sec) |
| 40 | web_url | Url: cookies.html_2 |
| 49 | lr_think_time | Think Time - 6 (sec) |
| 51 | web_url | Url: CustomBodyThro |
| 60 | web_convert_from_formatted | Convert Formatted Da |
| 67 | lr_think_time | Think Time - 8 (sec) |
| 69 | web_custom_request | Custom Request: ech |
| 81 | web_convert_from_formatted | Convert Formatted Da |
| 86 | lr_think_time | Think Time - 13 (sec) |
| 88 | web_custom_request | Custom Request: ech |
| 100 | web_convert_from_formatted | Convert Formatted Da |

If your script contains many steps, you can use the search box to search for matching text in the different parts of the steps.




| To access | View > Steps |
|---|---|
| **Important information** | • You can search within an action or script but not across multiple scripts in a solution.<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63.<br>• You can view the script in either action or script scope.<br>• Every step that has a snapshot is marked with an icon. When hovering over a step that has an associated thumbnail it is presented as a tooltip.<br>• Double clicking a step, takes you to the corresponding location in the script and synchronizes all other panes.<br>• Step Navigation is synchronized based on the validity of the script. You can check the status of the pane during script editing. |

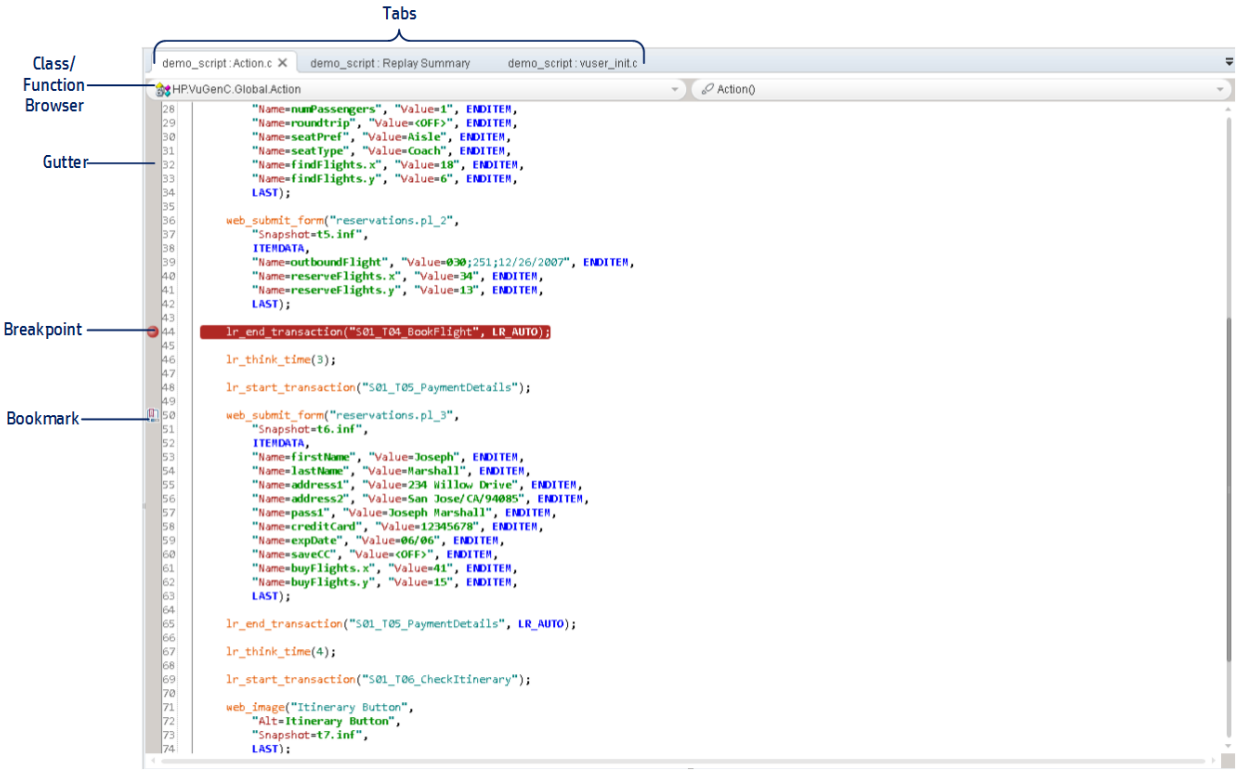
User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Search box** | You can search different parts of the steps for matching text. The following parts of the steps can be searched:<br>• **All**. (Default) All parts of the step (name, step, arguments).<br>• **Name**. The name of the step.<br>• **Step**. The description of the step (for example, web_url, web_custom_request)<br>• **Arguments**. The arguments for the step.<br>Enter the text you want to search for in your steps, in the search edit box and select the part of the steps you want to search. The Steps pane displays only those steps that match your search criteria. |
| **Line** | The number of the step in the script. |
| **Name** | A step name. |
| **Step** | The step type. |
| | **File parser indicator.** A green symbol indicates that parsing succeeded and a red symbol indicates that parsing failed. |
| **Action** | The action into which the step was created. |
| **# steps displayed** | Displays the total number of steps in the script or in the action. |

# Editor Pane

The Editor enables you to edit scripts and other related script files. In addition, you can open a browser session to search sites, such as the LoadRunner Forum.




| | |
|---|---|
| **To access** | The editor is opened when VuGen is loaded. |
| **Important information** | • The editor is automatically displayed as part of the default layout.<br>• Other interface panes, such as Output, Error, and Snapshot views, synchronize their displays based on your location in the editor.<br>• Script modifications are displayed as highlighted text in the editor. For example:<br>  • Script has been modified but not saved.<br>  • Script has been modified and saved.<br>  • Breakpoint has been inserted. |

## Supported Programming Languages

Recorded scripts are generated in C, which has full language and parsing support in VuGen. You can enhance your Vuser scripts by adding standard ANSI C functions. ANSI C functions allow you to add comments, control flow statements, conditional statements to your Vuser scripts. You can add standard ANSI C functions to any type of Vuser script. For details, see " C Vuser Scripts" on page 873.

In addition, the VuGen editor enables you to write manual scripts with the following programing languages:

- Java: For details, see "Java Vusers" on page 874.
- C#: For details, see ".NET Vusers" on page 875.
- VB.NET: For details, see "Create a Vuser Script in Visual Studio" on page 878.

## Tips for Working with the Editor

- Press **Ctrl +Tab** to display a list of tabs and panes. Highlight and click to switch tabs in the editor.
- Press **Ctrl +Tab** to display a list of tabs and panes. Highlight and click to switch tabs in the editor.
- Press **Ctrl + G** or select **Search > Go to...** to go to a specific line in the script.
- Double-click an asset in the Solution Explorer to open it in the editor.
- Drag any type of file into the editor and modify it. To save your changes, select **File > Save File As**.

## Code Completion and Tooltips

Code completion (autocomplete) enables you to quickly and accurately write code by providing a list of code items from which you can select options. Press **CTRL + SPACE** to activate statement completion when your cursor is in the editor. Tooltips appear when you hover over a code element with your mouse.

The following table describes available code completion items, scope, identifying icon, and tooltip context:

| Code Completion Item | Scope | Icon | The Tooltip Displays |
|---|---|---|---|
| **ANSI C Keywords and Types** | All possible standard C keywords. | | Function type, name, and parameters |
| **LR API Functions** | All steps in the script. | | LR API Step |
| **LR API Constants** | Used to delimit groups of parameters in steps. For example, ENDITEM | | LR API Constant |
| **User Functions** | All the functions that you have defined in action files. | | • Function type, name and parameters.<br>• When Using Functions (Method Insight)<br>The required arguments, highlighting each argument as you define it moving to the next argument when you enter the delimiter. |

| Code Completion Item | Scope | Icon | The Tooltip Displays |
|---|---|---|---|
| **Variables** | Local variables – visible only in the function where they have been defined.<br>Global variables – defined outside of any function body. Available everywhere in the script. | | Type and name. |
| **Parameters** | Available only in the function body where they have been defined. | | • Parameter type and name<br>• When Using parameters (Method Insight)<br>The required arguments, highlighting each argument as you define it moving to the next argument when you enter the delimiter. |

By default, VuGen uses code completion globally. To disable code completion, select **Tools > Options > Editor Tab > Code Completion**. Clear the **Enable code completion features** check box.

## Code-Coloring

To facilitate script writing and debugging, code item types are colored by an identifying background and foreground. The colored text enables you to easily read scripts and scan for syntax errors.

The following table provides examples of code item types and their assigned colors.

| Code Type | Color Example |
|---|---|
| Comments | `/* comment */ // comment` |
| Keywords | `if (a) { } else { }` |
| Method Parameter Name | `foo( "parameter=value" )` |
| API function | `web_url` |
| Method Call | `foo()` |
| String | `char * text = "Hello, World!"` |

In addition, you can customize code item types to suit your needs by selecting **Tools > Options > Editor Tab > Code Color**.

## Script Folding

Script folding enables you to selectively hide and display sections of a script, making it easier to manage large scripts by viewing only those sections that you are currently editing. For details, see "Editor Options Tab" on page 113.

## Community Search

You can perform Web searches from the VuGen toolbar, which opens a browser tab in the editor. The default Web site is the LoadRunner Forum which enables you to search topics, post questions, or blog about your expertise. You can add additional search sites by selecting **Tools > Options > General > Community**. For details on adding additional sites, see "General Options Tab" on page 109.

## User Interface Elements

The following table describes the editor user interface and the right-click (context) menu options (unlabeled elements are shown in angle brackets).

| UI Element | Description |
|---|---|
| **<Gutter>** | The editor gutter enables you to add toggle functionality including:<br>• Breakpoints<br>For details, see "Working with Breakpoints" on page 332 or "Breakpoints Pane" on page 101.<br>• Bookmarks<br>For details, see "Use Bookmarks" on page 324. |
| **Class/Function Browser** | When enabled, you can navigate quickly to a specific class or function in your script by selecting it from the drop-down list in the browser. |
| **Line numbers** | Display of line numbers in the script. |
| **Context Menu** | |
| **Comment or Uncomment** | Enables you to comment or uncomment highlighted script lines. |
| **Decode with DFE** | Applies to Web - HTTP/HTML including Web - HTTP/HTML steps in Flex Vuser scripts. For details see, "Data Format Extensions (DFEs) - Overview" on page 741. |
| **Show Snapshot** | Shows the snapshot associated with the highlighted script step. |
| **Show Arguments** | When the cursor is placed on a function, the Step Properties dialog box appears with the details of the function's arguments. |

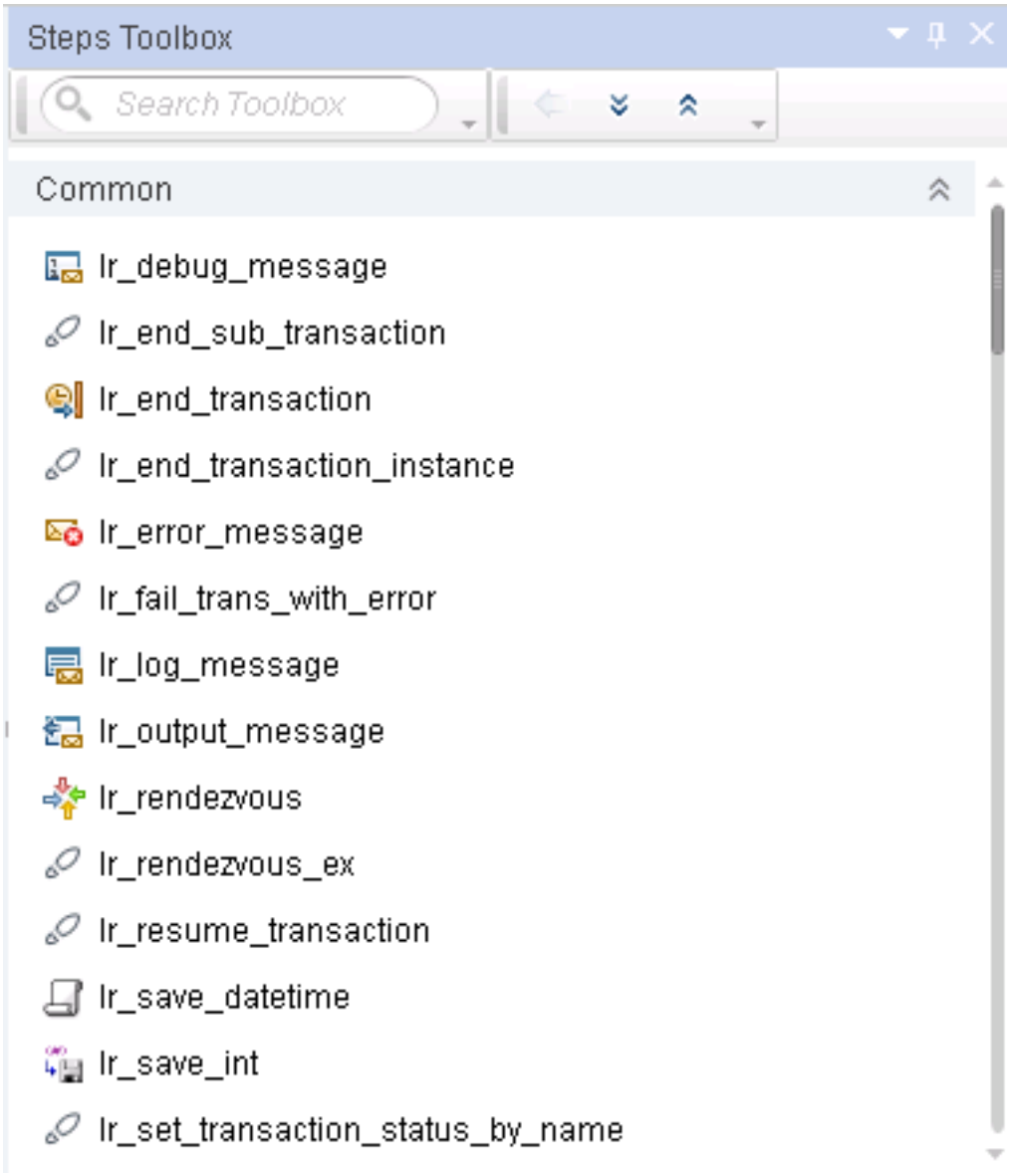
| UI Element | Description |
|---|---|
| **Correlate Selection** | Opens Design Studio and scans for dynamic values to be correlated in the selected script section. |
| **Restore Masked String (*value*)** | Restores the original value for a masked string. |
| **Mask String** | When the cursor is placed in a password, for example, inserts an `lr_unmask` function to encode the string. |
| **Surround with Transaction** | Inserts an `lr_start_transaction` function immediately above the highlighted script and a `lr_end_transaction` function immediately below the highlighted script. |
| **Go to Step in Replay Log** | Navigates to the location in the Output pane that correlates to the function in the editor. |
| **Insert** | Inserts a **New Step** into your script.<br> Inserts a **Start Transaction** step into your script. For details, see "Transaction Overview" on page 338.<br> Inserts an **End Transaction** step into your script. For details, see "Transaction Overview" on page 338.<br> Inserts a **Rendezvous point** step into your script. For details, see "Rendezvous Points" on page 343.<br>**Comment.** Inserts comments into your script.<br>**Log Message.** Inserts an **lr_log_message** step into your script. |
| **Toggle Breakpoints** | Adds or removes a breakpoint. |
| **Search Community** | Opens the default Community Search browser. |

## See also:

- "Editor Options Tab" on page 113

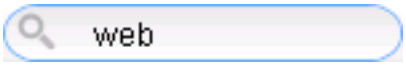
# Steps Toolbox Pane

This pane enables you to drag and drop API functions into your script.

| | |
|---|---|
| **UI example** |  |
| **To access** | Use one of the following:<br>• **Design > Insert in Script > New Step**<br>• Click within a script and select **Insert > New Step** from the right-click menu<br>• Press Ctrl+Alt+B |

| | |
|---|---|
| **Important information** | • A step's associated parameter dialog box opens when you add the step to the script.<br>• You can drag and drop steps into your script.<br>• You cannot drag and drop a step into a step from the **Steps Toolbox** but you can manually add a step parameter within a step.<br>• If you insert a step into the incorrect location in your script, the script may fail. |
| **Relevant tasks** | "Insert Steps into a Script" on page 354. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Function List>** | Displays a list of available functions divided into the following categories:<br>• Common<br>• Web checks<br>• Services<br>• XML<br>• Web JS<br>• Async |
|  | **Search.** Enables you to perform an incremental search in the function list by entering text. For example, if you type "web" into the search box, the function list will display only those function that include the letters "web". |
| | **Add.** Add the highlighted step to the current location in your script. |
| | **Expand/Collapse.** Expand or collapse the step categories. |

# Bookmarks Pane

The Bookmarks pane displays a list of the bookmarks in your Vuser script. You can navigate between the bookmarks to help analyze and debug your code.

| | |
|---|---|
| **To access** | **View > Bookmarks** |

| | |
|---|---|
| **Important information** | • All bookmarks added to a Vuser script are maintained after you close and reopen the Bookmarks pane.<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63. |
| **Relevant tasks** | "Use Bookmarks" on page 324 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Bookmarks list>** | Displays a list of all bookmarks that are defined in the Vuser script. You can double-click any bookmark line to navigate directly to the relevant line in the Vuser script. |
| | Toggles the status of the selected bookmark. |
| | Navigates to previous bookmark in the pane. |
| | Navigates to next bookmark in the pane. |
| | Deletes the selected bookmark. |
| | Deletes all bookmarks. |

### See also:

- "VuGen User Interface" on page 59

# Snapshot Pane

Snapshots contain the data generated by the traffic between the client and the server and are captured when a script is recorded and when the script is replayed.

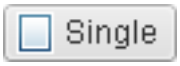
Snapshots are displayed in various formats and provide different functionality depending on the Vuser protocol.

**Note:** The Snapshot pane is not available for all Vuser protocols.

| | |
|---|---|
| **To access** | Select **View > Snapshot**, or click the **Show Snapshot Pane** button on the VuGen toolbar. |
| **Important information** | • A snapshot displays all data associated with a specific step in a script.<br>• Snapshots enable you to search for correlations, compare record versus replay snapshots and search for the specific values using the standard search operation.<br>• Press **Ctrl+F** to search for text within a snapshot view. (Search is not supported for the JSON view.)<br>• The appearance and functionality of the Snapshot pane varies depending on the protocol of the current Vuser script. In addition to the standard controls, the Snapshot pane may display controls that are specific to the current Vuser protocol.<br>• The new Web snapshot model is backward compatible with previous versions of LoadRunner. However some snapshot data may be missing. If this occurs, regenerate the script.<br>• When using Windows 2008 R2 and opening a snapshot from the step navigator in SAP GUI and Web protocols, the snapshots might not open automatically. **Workaround:** Internet Explorer Enhanced Security Configuration must be disabled to view help content. It is enabled by default. (**Control Panel > Administrative tools > Server manager > Configure IE ESC**).<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63. |

## Standard Snapshot pane controls

The Snapshot pane displays the following standard controls:

| UI Element | Description |
|---|---|
|  | Shows a single snapshot in the Snapshot pane. |
| | Splits the Snapshot pane to show two snapshots. |
| | Displays two snapshots in the Snapshot pane - one to the side of the other. |
| | Displays two snapshots in the Snapshot pane - one above the other. |
| | Shows the record snapshot in the Snapshot pane. |

| | |
|---|---|
| | Shows the replay snapshot in the Snapshot pane. |
| | Select the iteration number of the replay snapshot to display. |
| | Synchronizes the display of the two snapshots if the Snapshot pane is split.<br><br>**Note:** Snapshot synchronization is available for only specific Vuser protocols, and for only specific views within the protocols. |
| | Compares the two snapshots that are currently displayed in the Snapshot pane. To enable the **Compare** functionality, you must first split the Snapshot pane to show two snapshots. By default, VuGen uses the *WDiff* utility to compare snapshots. You can specify an alternative comparison tool as described in "Scripting Options Tab" on page 119.<br><br>**Note:** The snapshot comparison functionality is available for only the Web HTTP/HTML and Web Services protocols. |

## Snapshot pane controls for Citrix, RDP, SAP, and PCoIP protocols

User interface elements are described below:

| UI Element | Description |
|---|---|
| Image | Displays a graphical representation of the snapshot. You can synchronize the display of two snapshots in the Snapshot pane. Snapshots display faster when the Image view is used than when the Full view is used. |
| Full | Displays a graphical representation of the snapshot. You cannot synchronize the display of two snapshots in the Snapshot pane. Snapshots display slower when the Full view is used than when the Image view is used. |
| **Context Menu** | |
| Copy Image to the Clipboard | Copies the image to the clipboard. |
| Insert Mouse Click | Inserts a mouse click function, for example, ctrx_mouse_click, at the point of the cursor in the script. |

| | |
|---|---|
| Insert Mouse Double Click | Inserts a double mouse click function, for example, ctrx_mouse_double_click, at the point of the cursor in the script. |
| Insert Sync on Bitmap | Inserts a sync on image function, for example, ctrx_sync_on_bitmap, at the point of the cursor in the script. Select the bitmap area with the cursor to get the bitmap values (x,y coordinates, width, height) for the function.<br><br>**Note:** For the PCoIP protocol (pcoip_sync_on_bitmap_ex), you can define the timeout period. |
| Insert Sync on Bitmap (by coordinates) | Inserts a sync on image function with synchronization area coordinates as function parameters, for example, ctrx_sync_on_bitmap. The parameters are entered in a dialog box displayed after selecting this option. The function is inserted at the point of the cursor in the script.<br><br>**Note:** For the PCoIP protocol (pcoip_sync_on_bitmap_ex), you can define the timeout period. |
| Insert Get Bitmap Value | Inserts a get bitmap value function, for example, ctrx_get_bitmap_value, at the point of the cursor in the script. This function retrieves the hashed string value of a bitmap for use in custom synchronization functions. Select the bitmap area with the cursor to get the bitmap values (x,y coordinates, width, height) for the function. |
| Insert Get Text | Inserts a get text function, for example, ctrx_get_text function, at the point of the cursor in the script. Select the text area with the cursor to get the values (x,y coordinates, width, height) for the function.<br><br>**Note:**<br>• Available only when the Vuser script is recorded on a Citrix server, with the LoadRunner Citrix agent installed on it.<br>• The Citrix agent does not support this function on Windows 10, Windows Server 2016, or later versions of Windows. |
| Insert Sync on Text | Inserts a sync on text function, for example, ctrx_sync_on_text_ex function, at the point of the cursor in the script. Select the text area with the cursor to get the values (x,y coordinates, width, height) for the function. This function waits until the specified text is displayed (agent installations only).<br><br>**Note:**<br>• Available only when the Vuser script is recorded on a Citrix server, with the LoadRunner Citrix agent installed on it.<br>• The Citrix agent does not support this function on Windows 10, Windows Server 2016, or later versions of Windows. |

| | |
|---|---|
| Insert Obj Get Info | Inserts an object get info function, for example, ctrx_get_obj_info, at the point of the cursor in the script. This function retrieves the current state of the requested object property.<br><br>**Note:** This menu item only appears when the Vuser script is recorded on a Citrix server, with the LoadRunner Citrix agent installed on it. |
| Insert Sync on Obj Info | Inserts a sync on obj info function, for example, ctrx_sync_on_obj_info, at the point of the cursor in the script. This function causes VuGen to wait for a certain state before continuing.<br><br>**Note:** This menu item only appears when the Vuser script is recorded on a Citrix server, with the LoadRunner Citrix agent installed on it. |

## Snapshot pane controls for Windows Sockets protocol

The snapshot for Windows Sockets protocol are displayed as textual and hexadecimal representations of data buffers sent and received.

User interface elements are described below:

| UI Element | Description |
|---|---|
| Hex | Displays the buffer data in hexadecimal. |
| Text | Displays the buffer data as text. |
| Go To | Opens the Go To Offset dialog box that enables you to navigate within the data buffer. |
| **Context Menu** | |
| Create Correlation (Select the text for correlation, from the Response Body tab) | Opens the Correlation dialog box. For details see "Correlating" on page 251. |

## Snapshot pane controls for Web - HTTP/HTML protocol

The snapshot for Web - HTTP/HTML protocol are displayed as textual and hexadecimal representations of the request and response.

User interface elements are described below. These are available for both the Recording and Replay snapshots.

| UI Element | Description |
|---|---|

| | |
|---|---|
| Http Data | Displays step data in HTTP format. This enables you to view in-depth information about the step, including request data, response data, cookies, and headers. For more information, see "Web Snapshots - Overview" on page 734. |
| Page View | Displays step data in HTML format. |
| Tree | (**Http Data** view only) Displays the HTTP flow data in a tree structure.<br>The data is separated into separate tabs: **Raw Data**, **Request Body**, **Response Body**, **Headers**, **Cookies**, and **Query String**.<br>For some of the above tabs, you can display the data in text, hexadecimal, or json views, using the buttons on the right: **Text View**, **Hex View**, and **JSON View**.<br>**Note:** Depending on your application, some tabs may not have any data. |
| Grid | (**Http Data** view only) Displays the HTTP flow data in a list format. |
| Iteration | The iteration number to display in the pane (only for Replay snapshots) |
| **Context Menu** | |
| Create Correlation | Opens the Correlation dialog box. For details see "Correlating" on page 251.<br>**Note:** This option is available in the Response view only. This option is available for attribute and value nodes only. |
| Create Correlation Rule | Opens the Add as Rule dialog box. For details see "Manually Correlate Scripts" on page 260.<br>**Note:** This option is available in the Response view only. This option is available for attribute and value nodes only. |
| Create Parameter | Opens the Create Parameter dialog box. For details see "Create Parameters" on page 361.<br>**Note:** This option is available in the Response view only. This option is available for attribute and value nodes only. |
| Search Community | Opens the LoadRunner Support Forum. |

| | |
|---|---|
| Add Text Check Step | Opens the Find Text dialog box. For details see "Add a Text Check From the XML View in the Snapshot Pane" on page 300.<br><br>**Note:** This option is available in the Response view only. This option is available for attribute and value nodes only. |

## Snapshot pane controls for protocols with an XML request/response (such as Web Services)

The snapshots for XML protocols are displayed as XML and textual and hexadecimal representation of the request and response bodies.

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Displays an XML view of the server response. |
| | Displays an XML view of the request. |
| | Opens the XPath search dialog box, allowing you to search the XML code using a standard XPath expression. |
| **Context Menu** | |
| Copy Selection | Copies the text that is selected in the text view to the clipboard |
| Search Community | Performs a community search using the text that is selected in the text view as the search string. For details about performing a community search, see "Editor Pane" on page 77. |
| Copy XPath | In the tree view, copies the XPath of the selected node to the clipboard. In the text view, copies the XPath of the XML element in which the cursor is located to the clipboard. |
| Copy full value | In the tree view, copies the full XML code of the selected node to the clipboard. In the text view, copies the full XML code of the XML element in which the cursor is located. |
| Node Properties | Opens the XML Node Properties dialog box. |

| | |
|---|---|
| Insert XML Check | Opens the Insert XML Check dialog box that enables you to insert an **XML Find** step into the Vuser script.<br><br>**Note:** This option is available in the Response view only.<br>This option is available for **attribute** and **value** nodes only. |
| Save value in parameter | Opens the Save Value as Parameter dialog box that enables you to save the selected value to a simple parameter.<br><br>**Note:** This option is available in the Response view only.<br>This option is available for attribute and value nodes only. |
| Save XML in parameter | Opens the Save Value as Parameter dialog box that enables you to save the selected value to an XML parameter.<br><br>This option is available in the Response view only. |
| Create Correlation | Opens the Correlation tab in the Design Studio. The text selected in the Snapshot pane appears as a manual correlation entry in the Design Studio. For details, see "Manually Correlate Scripts" on page 260.<br><br>**Note:** This option is available in the Response view only.<br>This option is available for attribute and value nodes in the tree view, and when text is selected in the text view. |
| Create Correlation Rule | Opens the Add as Rule dialog box that enables you to add the selected text as part of a correlation rule. For details, see "Correlating Values" on page 254.<br><br>**Note:** This option is available in the Response view only.<br>This option is available for attribute and value nodes in the tree view, and when text is selected in the text view. |
| Copy XML | |

## Snapshot pane controls for Database Protocols

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Shows the record snapshot in the Snapshot pane. |

| Data Grid | Displays data in a data grid. |
|---|---|
| **Context Menu** | |
| Create Correlation | Opens the Correlation dialog box. For details see "Correlating" on page 251.<br><br>**Note:** This option is available in the Response view only. This option is available for attribute and value nodes only. |
| Save Grid to File | Saves the grid's contents to a CSV file. |

## Copying images to the clipboard

You can copy an image-based snapshot to the clipboard. This enables you to import the image into a graphics application, where you can analyze and modify the graphic.

For details on how to copy a snapshot to the clipboard, see "Work with Snapshots" on page 297.

**Note:** The "copy snapshot to the clipboard" functionality is available for only RDP, Citrix, and SAP GUI protocols.

## Copying snapshot text to the clipboard

You can copy text from a snapshot to the clipboard. You can then paste the text from the clipboard into another application.

For details on how to copy snapshot text to the clipboard, see "Work with Snapshots" on page 297.

## Customized Snapshot pane functionality

In addition to the basic Snapshot pane functionality, the Snapshot panes for some Vuser protocols include customized functionality. For example, the Snapshot pane for RDP Vuser scripts lets you display snapshots in either **Full** or **Image** modes; the Snapshot pane for Winsock Vuser scripts lets you display snapshots in either **Text** or **Hex** modes. The controls for the customized functionality can be found in the Snapshot pane toolbars.

## Activating the Snapshot on error functionality

In addition to showing record and replay snapshots, the Snapshot pane can display snapshots of errors that occurred during the replay of a script. The "snapshot on error" functionality is available for only specific Vuser protocols.

You can generate and display snapshots of errors only if the "snapshot on error" functionality is activated.

To activate the snapshot-on-error functionality:

1. Click **Replay > Runtime Settings**. The runtime settings dialog box opens.
2. Under **General**, click **Miscellaneous**.
3. Under **Error Handling**, select the **Generate snapshot on error** check box.

### See also:

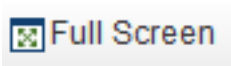
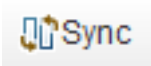
- "Work with Snapshots" on page 297

# Thumbnail Explorer

This pane enables you to flip through thumbnail images of your business process, enhancing your ability to navigate to specific locations in the **Editor** based on a visual representation of a step. Conversely, you can scroll through the **Editor** and see the visual context of your script in the **Thumbnail Explorer**.

| | |
|---|---|
| **To access** | Use one of the following:<br>• **View >Thumbnail Explorer**<br>• Click the button on the VuGen toolbar. |
| **Important information** | • You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63.<br>• You can configure the **Thumbnail Explorer** in **Options > Scripting Options**. For details, see "Scripting Options Tab" on page 119.<br>• Thumbnails are created in the same order as the actions in the Solution Explorer and not controlled by the settings in **runtime settings > General > Run Logic**.<br>• Enabling automatic creation of thumbnails in **Options > Scripting > Thumbnails** enables VuGen to create thumbnails during the application's idle time.<br>• If the Windows Aero theme is enabled, thumbnails capture more realistic images of the application. |
| **Relevant tasks** | "How to Modify the VuGen Layout" on page 63 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | Moves the cursor to the step in the **Editor** associated with the highlighted thumbnail in the **Thumbnail Explorer**. |
|  | Enables a full screen view of the thumbnail. |
|  | Synchronizes the scrolling in the **Editor** with the associated thumbnail in the **Thumbnail Explorer** and step in the **Step Navigator**. |
| | Filters out minor thumbnails that are not directly related to the recorded business process. |
| | Refreshes the generated thumbnails. |
| | Scroll a page left in the **Thumbnail Explorer**. |
| | Move to the previous thumbnail in the **Thumbnail Explorer**. |
| | Move to the next thumbnail in the **Thumbnail Explorer**. |
| | Scroll a page right in the **Thumbnail Explorer**. |

# Errors Pane

The Errors pane lists the replay and syntax errors found in your script, and enables you to locate each error so that you can resolve it.

| To access | View > Errors |
|---|---|

| | |
|---|---|
| **Important information** | • After every test process, such as code generation and replay, you can check the error pane for the error log.<br>• The error log includes errors, warnings and messages.<br>• Community search is available with context menu on highlighted error.<br>• Double-click message to jump to the location in the script.<br>• You can move this pane to different areas of the main user interface. For details, see "How to Modify the VuGen Layout" on page 63.<br>• When you open an existing script, the items displayed in the **Error Pane** are from the latest replay or compilation. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Define Available Categories>** dropdown | Filters the error list by the source of the error. |
| | Shows or hides syntax errors. |
| | Shows or hides warnings detected during the run. |
| | Shows or hides informational messages detected during the run. |
| **! <Exclamation Point>** | Task type:<br>• Error<br>• Warning<br>• Informational message |
| **Line** | The line containing the error. |

| | |
|---|---|
| **Description** | Description of the error, warning or message and advice on how to fix the problem. For example, a syntax error is displayed if you opened a conditional block with an If statement but did not close it with an End If statement, the description is Expected Expression.<br><br>**Note:** If the description does not fit within the Description column, a tooltip displays the full description when you hover the cursor over the column. In certain cases, VuGen is unable to identify the exact error and displays a number of possible error conditions, for example: Expected 'End Sub', or 'End Function', or 'End Property'. Check the statement at the specified line to clarify which error is relevant in your case. |
| **File** | The name of the file that contains the problematic statement. |
| **Path** | The full path of the file that generated the error. |
| **Test** | The name of the script in which the error was detected. |

See also:

- "VuGen User Interface" on page 59
- "Debugging" on page 328

# Tasks Pane

This pane enables you to add, edit and track tasks associated with an individual script or the overall goals of the project. Tasks are divided between user defined tasks and tasks that are inserted into the Vuser script as action items using keywords such as ToDo, Undone and FixMe.

User defined tasks are displayed when you select the User Tasks option in the Task pane. Action items are displayed when you select the Comments option in the Task pane.

| | |
|---|---|
| **To access** | **View > Tasks** |
| **See also** | "VuGen User Interface" on page 59 |

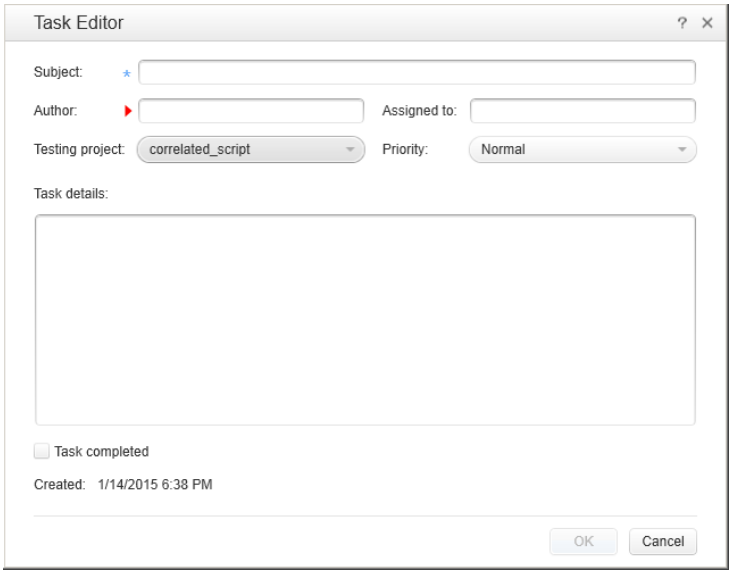
User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Comments view** | Comment tasks are added directly into your script using the comment syntax of your scripting language and include a keyword such as TODO or FIXME. For example, in C a comment script would look like this:<br>`//TODO Add Parameter`<br>The Task Pane displays the following information about each task:<br>• !: Task Type<br>• Line: What line the task is located. Double-clicking the task jumps to that location in the script.<br>• Description: The Keyword and the task contents.<br>• File: Action<br>• Path: File location of the action. |
| **User Tasks view** | You can add, edit, delete user tasks:<br>Click to add a task with the "Task Editor" below.<br>Click to edit a task with the "Task Editor" below.<br>Click to delete a task. |
| **<Task Filter>** | You can filter tasks associated with a particular script or see all the tasks associated with the solution. |

# Task Editor

This dialog box enables you to add or edit user tasks.

| | |
|---|---|
| **UI example** |  |
| **To access** | **View > Tasks > User Tasks tab**<br><br>Adds a new task<br><br>Opens the highlighted task for editing |
| **Important information** | • A blue asterisk indicates a required field.<br>• You can copy and duplicate tasks using the right-click menu. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Subject** * | Brief description of task. Required field. |
| **Author** * | User who initiates the task |
| **Assigned to** | User who is assigned to complete the task |
| **Testing Project** | Script associated with the task. |
| **Priority** | You can assign a priority to the task:<br>• High<br>• Normal<br>• Low |
| **Task details** | A textual description of the task |

| UI Element | Description |
|---|---|
| **Task completed** | A check box indicating that you have completed the task |
| **Created** | A non-editable field that displays the date and time the task was created |

# Output Pane

The Output pane displays messages that were generated during the recording, compilation, and replay of your script.

| | |
|---|---|
| **To access** | Select **View > Output**, or click the **Show Output Pane** button on the VuGen toolbar. |
| **Important information** | • You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63.<br>• When you open an existing script, the items displayed in the **Output Pane** are from the latest replay or compilation. |
| **See also** | "VuGen User Interface" on page 59 |

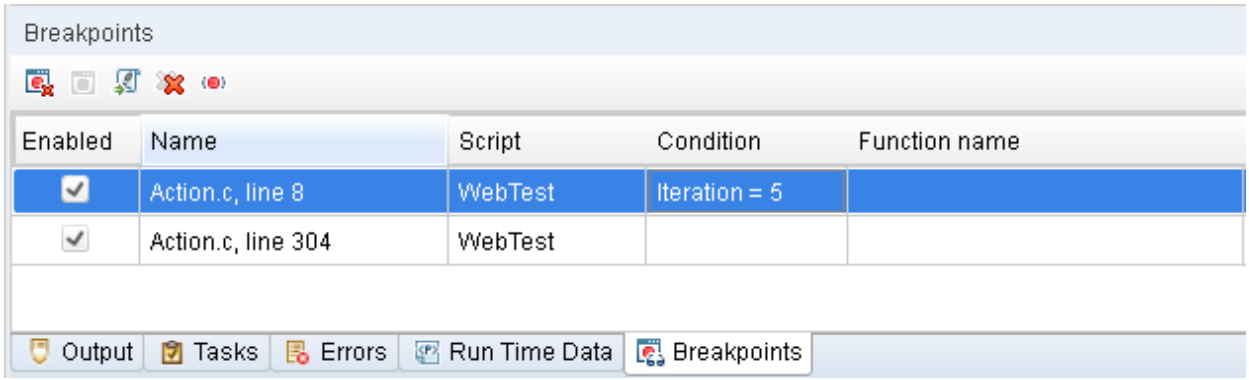
User interface elements are described below:

| UI Element | Description |
|---|---|
| | The type of output to display. The following types are available:<br>• **Replay.** Displays the messages generated by the script replay.<br>**Note:** If you double-click an entry in the Replay log, VuGen moves the cursor to the corresponding line in the Editor.<br>• **Compilation.** Displays the compilation messages.<br>• **Code Generation.** Displays the code generated during the recording.<br>• **Recording.** Displays the messages generated during the recording.<br>• **Recorded Events.** Displays events that occurred during recording. |

| UI Element | Description |
|---|---|
| | Clears all of the messages from the message list. |
| | **Toggle Line Wrap.** When selected, wraps the text of each message onto the next line - as required. |
| | Jumps to the location in the source document relevant to the selected output message. |
| | **<Find box>.** The text string that you want to find. You can refine your search by selecting one of the **Options** described below.<br>Press ENTER to begin the search. |
| | **Find Previous / Find Next.** Highlights the next or previous string that matches the text you entered in the **Find** box.<br>These buttons are available only after you enter text in the **Find** box. |
| | Enables you to refine your search with the following options:<br>• **Match Case.** Distinguishes between upper-case and lower-case characters in the search.<br>• **Match Whole Word.** Searches for occurrences that are only whole words and not part of longer words.<br>• **Use Regular Expression.** Treats the specified text string as a regular expression.<br>**Note:** Extended regular expressions and multi-line searches are not supported. |
| | Opens the Save As dialog box, enabling you to save the contents of the message list as a text file. |
| **View Summary**<br>[Available in the Replay log only] | Opens the Replay Summary tab. For details, see "Replay Summary Pane" on page 131. |

# Breakpoints Pane

The Breakpoints pane enables you to set and manage breakpoints to help analyze the effects of the script on your application at pre-determined points during script execution.




| | |
|---|---|
| **To access** | **View > Debug > Breakpoints** |
| **Important information** | You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63. |
| **Relevant tasks** | "Debug Scripts with Breakpoints" on page 335 |

## Breakpoints Pane

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | Removes the selected breakpoint. |
| | Removes all breakpoints. |
| | Locates the cursor in the Vuser script at the line that contains the selected breakpoint. |
| | Disables the selected breakpoint if it is enabled, and enables the selected breakpoint if it is disabled. |
| | Allows you to enter conditions for the selected breakpoint. See "Breakpoint Condition Dialog Box" on the next page for more details. |
| **<Breakpoints Grid>** | A list of the breakpoints and their locations in the script. To enable a breakpoint, select the **Enable** check box next to that breakpoint. To disable a breakpoint, clear the **Enable** check box. |
| **Enabled** | A check box that specifies whether the breakpoint is enabled or disabled, and enables you to enable or disable the adjacent breakpoint. |

| | |
|---|---|
| **Name** | The name of the file that contains the breakpoint, and the line number within the file that contains the breakpoint. |
| **Script** | The name of the Vuser script that contains the breakpoint. |
| **Condition** | The condition that applies to this breakpoint. If there is no condition, the replay will always stop at the breakpoint. |
| **Function Name** | The name of the function within the Vuser script that contains the breakpoint. |

## Breakpoint Condition Dialog Box

The Breakpoint Condition dialog box enables you to condition breakpoints by the iteration number, the value of one or more parameters, or a combination of both parameter values and iteration number.

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Break when iteration number is** | A check box that enables you to specify a breakpoint dependant on the iteration number. |
| **<Operand>** | Choose one of the following operands:<br>• = (equal)<br>• <= (less than or equal to)<br>• => (equal to or greater than)<br>• < (less than)<br>• > (greater than) |
| **to** | Enter an iteration number. |

| | |
|---|---|
| **Break when Parameter** | A check box that enables you to specify a breakpoint dependant on the value of a parameter. |
| **<Parameter Name>** | Choose a parameter from the drop-down list |
| **Value is** | Enter the parameter value for which you want a breakpoint. |
| **OK** | Apply the conditions to the selected breakpoint. All future replays will only stop at this breakpoint if these conditions are met. |

# Call Stack Pane

This debug pane enables you to view information about the functions that are currently on the call stack of your script.

| | |
|---|---|
| **To access** | **View > Debug > Call Stack** |
| **Important information** | • This pane is relevant only when a run session is paused.<br>• You can double-click any element in the Call Stack pane to navigate to the beginning of the relevant function/action.<br>• This pane is read-only.<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63. |
| **See also** | • "VuGen User Interface" on page 59<br>• "Debugging" on page 328 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Function name** | The name of the function currently called. |
| **File name** | The name of the file containing the called function. |
| **Line #** | The line number on which the function definition begins. |

# Watch Pane

The Watch pane enables you to monitor variables while a script runs.

| | |
|---|---|
| **To access** | **View > Debug > Watch** |

| | |
|---|---|
| **Important information** | • This pane is relevant only when a run session is paused.<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Enables you to add a variable to the watch list. |
| | Enables you to edit the selected variable in the watch list. |
| | Deletes the selected variable from the watch list. |
| | Deletes all the variables from the watch list. |
| **Expression** | The variable whose value you want to watch. |
| **Value** | The current value of the variable. The evaluated value is displayed only when a run session is paused. |
| **Type name** | The type of the variable's value after it is evaluated (for example, **Integer** or **Char**). If a variable cannot be evaluated in the current context, the type displayed is **Incorrect expression**. |

# Runtime Data Pane

The Runtime Data pane displays information about the current script execution.

| | |
|---|---|
| **To access** | **View > Debug > Runtime Data** |
| **Important information** | • The Runtime Data pane is accessible during script replay only.<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63. |
| **See also** | "Replay a Vuser Script" on page 296 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Iteration** | Displays the current iteration number. |
| **Action** | Displays the Action name of the currently replayed step. |
| **Line Number** | Displays the line number of the currently replayed step. |
| **Elapsed time** | Displays the time that has elapsed since the start of the replay. |
| **<Parameters>** | Displays all parameters defined, together with the script and their substitution values based on the selected update method (sequential, unique, etc.). VuGen shows this information even if the parameter is not used in the script. |

# .NET Recording Filter Pane

This pane enables you to manage .NET filters.

| | |
|---|---|
| **To access** | **View > .NET Recording Filter** or the Show .NET Recording Filter toolbar button . |
| **See also** | • ".NET Filters Overview" on page 618<br>• ".NET Filters - Advanced" on page 622<br>• "Guidelines for Setting .NET Filters" on page 619 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Filter List>** | A list of the recording filters.<br>• **Environments**. Built-in system filters for .NET Remoting, ADO.NET, Enterprise Services, and WCF (Windows Communication Foundation).<br>• **Custom filters.** Filters that you created manually for this script. TO create a custom filter, click **New**. |
| **<Filter Tree>** | The Filter tree uses symbols to illustrate the elements and their status. For details about each of the icons, see the table below.<br>• Element icons represent the type of element—assembly, namespace, class, method, structure, property, events, or interfaces.<br>• A check mark or X adjacent to the element icon, indicates whether or not the element is included or excluded.<br>• A bold element indicates that it was explicitly included or excluded. This may be a result of being manually included or excluded by the user or by a pre-defined rule in the environment filter. If you reset a bold node, it returns to its original, non-bold state. |

| | |
|---|---|
| **New** | Opens the Create a New Filter dialog box, in which you create an empty filter or a new filter based on an existing one. For more information, see "Create a New Filter Dialog Box [.NET Protocol]" on page 625. |
| **Save** | Saves the changes you made to filter. |
| **Delete** | Deletes the selected custom filter. The filter pane prompts you for a confirmation. |
| **Add Reference** | Opens the Add Reference dialog box with a list of .NET Framework components or assemblies in the Public Assemblies folder. For more information, see "Add Reference Dialog Box [.NET Protocol]" on page 624.<br><br>**Note:** If you add a reference to a DLL for a .NET filter, VuGen adds it to the script's reference list only if the script accesses the DLL's methods during recording. |
| **Remove Reference** | Removes the assembly that is selected in the Filter pane and disables all of the rules associated with it. All disabled rules are enabled after adding the reference to this assembly once again. The Filter mechanism prompts you for a confirmation. |
| | **Include.** Includes the selected element. If you manually include a parent node, the filter mechanism includes the child elements below it, provided that no other rule exists. For example, if you include a class, it will include all its methods unless you specifically excluded a method. |
| | **Exclude.** Excludes the selected element. The child elements are also excluded unless they were included by another rule. By default, when you exclude a *class*, the filter mechanism applies the Exclude attribute to the class, but it allows the recording engine to record activity within the methods of the excluded class. When you exclude a *method*, however, the filter mechanism applies Totally Exclude, preventing the recording engine from recording any activity within the methods of the excluded class. Advanced users can modify these setting in the filter file. |

| | |
|---|---|
| | **Reset.** Removes the manual inclusion or exclusion rule. In this case, the element may be impacted by other parent elements.<br>The inclusion and exclusion rules have the following properties:<br>• The rules are hierarchical—if you add an include or exclude rule to a class, then the derived classes will follow the same rule unless otherwise specified.<br>• A rule on a class only affects its public methods, derived classes, and inner classes.<br>• A rule on a namespace affects all the classes and their public methods.<br>• Note that adding or removing assemblies does not necessarily affect the classes that they contain—you can remove an assembly, yet its methods may be recorded due to the hierarchical nature of the filter.<br>• As part of the filter design, several methods, such as **.cctor()** and **Dispose(bool)**, do not follow the standard hierarchical rules.<br>**Note:** The resetting of a parent node does not override a manual inclusion or exclusion applied to a child node. For example, if you manually **exclude** a method, and then reset its class, which by default **included** all sub-nodes, your method will remain excluded.<br>Properties and events are view-only and cannot be included or excluded through the .NET Recording Filter pane. In addition, several system related elements are protected and may not be altered.<br>For tips about including and excluding elements in the filter, see "Guidelines for Setting .NET Filters" on page 619. |
| | **Navigate.** Navigates to the previous or next tree node visited by the user. |
| | **Show non-public items**. By default, the filter tree shows only public classes and class members. By clicking this button, you instruct the tree view to display non-public items.<br>If you include a class which contains non-public items, they will not be added to the filter automatically. You must explicitly include each non-public item to the filter. |
| **Impact Log** | Opens the Impact Log, which indicates what your last changes were and how they affected your filter. The user actions are listed in descending order, with the latest changes at the top.<br>For each element, the log indicates how each manual inclusion or exclusion were affected. It also provides a link to the affected element in the filter pane hierarchy. |
| **View Impact Log** | Opens the Impact log for the selected filter. The Impact log shows which nodes in the tree were affected by recent actions. |

The following table shows the filter tree icons that represent the various elements:

| Icon | Description | Icon | Description |
|---|---|---|---|
| | assembly | | interface |
| | assembly that couldn't be loaded | | method |
| | assembly that was partially loaded | | static method |
| | class | | namespace |
| | constructor | | property |
| | static constructor | | static property |
| | event | | structure |
| | static event | | |

# Options Dialog Box

The **Options** dialog box is comprised of VuGen application settings. These settings are common to all available protocols in VuGen.

## General Options Tab

This pane enables you to configure general user interface options.

### Task List

| To access | VuGen > Tools > Options > General > Task List |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Comment Tags** | This pane enables you to add, delete or modify tags names that you can use to label comment tasks in your scripts.<br>**Tags.** List of available tags.<br>**Name.** Displays the name of the highlighted tags in the token list. This area enables you to modify, add or delete the current tag.<br>**Add.** Enables you to add a tag to the token list.<br>**Edit.** Enables you to modify the name of tag from the token list.<br>**Delete.** Enables you to delete a tag from the token list. |

### Usage Data Collector

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > General > Usage Data Collector** |

You can help improve the VuGen experience by selecting the **I want to help, VuGen can send the usage data** option. Selecting this option enables automatic collection of anonymous data about your most used features.

### Scripts and Solutions

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > General > Scripts and Solutions** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Settings** | **Default project location**<br>Enables you to specify a path to your saved projects. Default location = `C:\Users\<username>\Documents\SharpDevelop Projects`<br>**Load previous solution on startup**<br>Enables you to automatically load the previous solution.<br>By default this option is enabled. |
| **Start Page Settings** | **Display Start Page on startup**<br>By default this option is enabled.<br>**Close Start Page after script loads**<br>By default this option is enabled. |

### ALM

| To access | VuGen > Tools > Options > General > ALM |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Connect to ALM in CAC mode** | Connects to Application Lifecycle Management using CAC (Common Access Card). This enables you to log in without providing a username and password. |

### Message Dialogs

Using these check boxes, you indicate which messages should be issued as popups and which should be hidden.

| To access | VuGen > Tools > Options > General > Message Dialogs |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Update Messages** | |
| **Update Messages > Check for updates and display Update Available message** | VuGen checks for updates at startup, and displays a message if an updated version is available. |
| **Information Messages** | |
| **Proxy Recording Tips Message** | This tip instructs you how to proceed when recording an application that runs on another machine or device. |
| **Web - HTTP/HTML Protocol JavaScript Message** | This message appears when you create a Web - HTTP/HTML script, informing you that you can create your Vuser script in JavaScript. |
| **Web - HTTP/HTML Protocol Mobile Recording Message** | This message announced the ability to use the Web - HTTP/HTML protocol to record mobile applications. |
| **Question Messages** | |

| | |
|---|---|
| **Closing VuGen and Browser Message** | This question appears when you close a TruClient script that is being edited in Interactive mode, warning you that the TruClient browser will be closed. |
| **Warning Messages** | |
| **Community Search Privacy Warning** | This warning message appears when you select the **Search Community** command from the context menu. It notifies you that the selected script text will be automatically passed into the search engine. |
| **Paste runtime settings Warning** | This warning appears when you paste runtime settings from the clipboard. It notifies you that the operation cannot to be undone. |
| **Action Reordered Warning** | This notification informs you that rearranging actions within the solution tree will not affect the run logic. |
| **Start Recording with IPv6 Warning** | This warning notifies you that you cannot record an application with IPv6 with the current protocol. |
| **Discard Correlation Warning** | The Design Studio issues this message when you discard a correlation. It also explains how to restore it. |
| **Long Time Correlation Warning** | The Correlation Studio issues this message when the auto correlation process is expected to take a long time. |
| **Git Reset Head Warning** | VuGen issues this message when you choose to reset HEAD. |

### Community

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > General > Community** |
| **See also** | For details, see Community Search in the "Editor Pane" on page 77. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Community Search Sites** | |

| | |
|---|---|
| | Adds a new search site to the list of Community search sites.<br>• **Name**: Enables you to specify a name of the search site that is displayed on the VuGen toolbar.<br>• **URL**: Enables you to specify the URL of the search site.<br> **Example:**<br>http://www.bing.com/search?q=%QUERY%<br>http://www.google.com/search?q=%QUERY%<br>http://www.google.de/search?q=%QUERY% (localized google site, e.g. de for Germany)<br>http://en.wikipedia.org/wiki/%QUERY% |
| | Enables you to edit the properties of the custom search site. |
| | Deletes the search site from the list of available sites. |
| | Moves the search site lower down on the list of available sites. |
| | Moves the search site higher up on the list of available sites. |

## Editor Options Tab

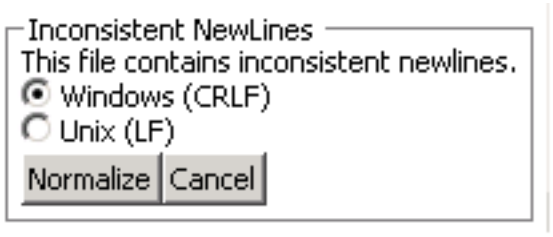
This pane enables you to configure the text editor options.

### General

| | |
|---|---|
| **To access** | **Tools > Options > Editor > General** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Font** | |

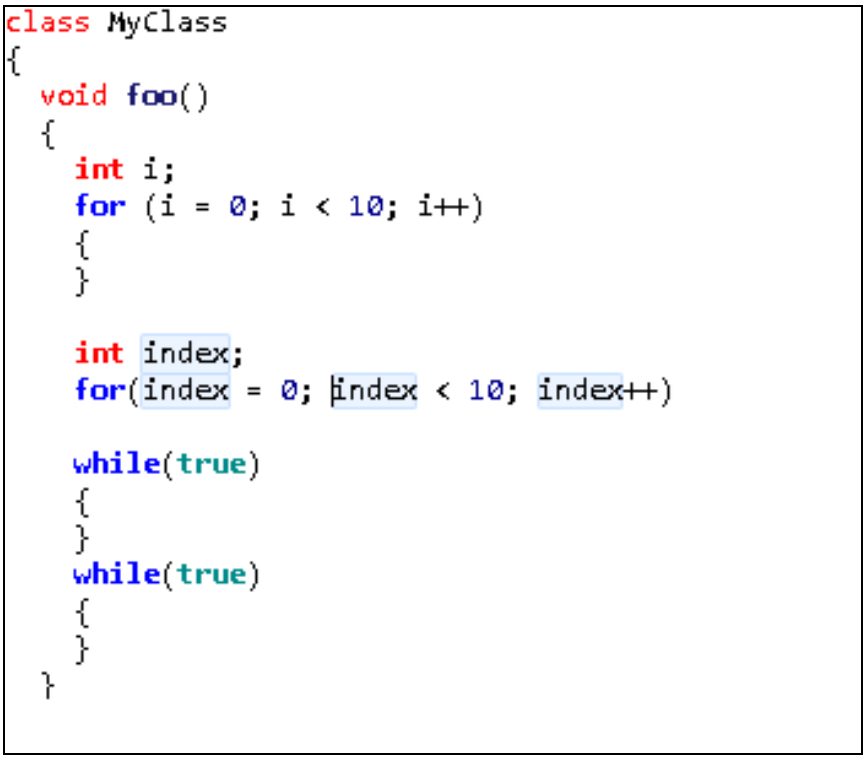
| | |
|---|---|
| **Text Font** | Enables you to select the font. |
| **Size** | Enables you to select the font size. |
| **General Options** | |
| **Word wrap** | Automatically wraps text to the next line.<br>By default this option is disabled. |
| **Show Class/Function Browser** | Show or hide the Class/Function Browser. When enabled, you can navigate quickly to a specific class or function in your script by selecting it from the drop-down list in the browser.<br>By default this option is enabled. |
| **Show line numbers** | Enable line numbering of script in the Editor.<br>By default this option is enabled. |
| **Check for line ending inconsistencies** | Enable the editor to check for end of line inconsistencies in your script. If enabled, a utility appears in the script editor that enables you to normalize line endings based either on a Windows standard (CRLF) or a Linux standard (LF).<br>  <br>This option is enabled by default. |
| **Enable working URL hypertext links in the Editor** | Enable URLs in scripts to function as hypertext links. Disabling this option may increase performance.<br>By default this option is enabled. |

### Markers and Rulers

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Editor > Markers and Rulers** |

User interface elements are described below:

| UI Element | Description |
|---|---|

| Markers and Rulers | |
|---|---|
| **Show spaces** | Enable markers that indicate where tabs exists.<br><br>By default this option is disabled. |
| **Show tabs** | Enable markers that indicate where line ends.<br><br>By default this option is disabled. |
| **Show end-of-line makers** | Enable markers that indicate where line ends.<br><br>By default this option is disabled. |
| **Underline errors** | Enable underlining of errors.<br><br>By default this option is enabled. |
| **Highlight matching brackets** | Enable highlighting of matching brackets.<br><br>By default this option is enabled. |
| **Highlight matching occurrences of non-keywords** | When this option is enabled, you can select a non-keyword in your script and the Editor will highlight all other occurrences in your script.<br><br>  <br><br><br>By default this option is enabled. |

### Behavior

| To access | VuGen > Tools > Options > Editor > Behavior |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Tabs** | |
| **Indentation** | Enables you to set the spacing for tab indentation.<br>Default indentation is four spaces. |
| **Convert tabs to spaces** | Converts tabs to spaces.<br>By default this option is disabled. |
| **Use smart indentation** | Automatically applies the indentation format from the previous line.<br>By default this option is enabled. |
| **Behavior** | |
| **Enable zoom with mouse wheel** | Enables you to use the mouse wheel to zoom.<br>By default this option is enabled. |
| **Cut or Copy entire line when nothing is selected** | Allows you to cut or copy an entire line without highlighting it in the editor, as long as the cursor is within that line.<br>By default this option is enabled. |
| **Enable Ctrl + Click for "Go to Definition"** | Enables the Ctrl + Click shortcut for "Go To Definition". This lets you move the cursor to the definition of a function you control-click on in the editor.<br>By default this option is enabled.<br>To enable this option for an extra file containing custom functions, right-click on the file in the Solution Explorer and select "Add to Parsing List". Parsing list functionality applies only to C language functions and the Web HTTP/HTML protocol JavaScript functions. |
| **Recording and Code Generation** | |
| **Use Compact mode for JavaScript scripts wherever possible** | When using JavaScript as the coding language, use Compact mode. This mode reduces the number of lines in the script by selectively removing line breaks. |

### Code Color

| To access | VuGen > Tools > Options > Editor > Highlighting |
|---|---|
| **Important information** | Highlighting options enable you to customize the color of script elements. |

User interface elements are described below:

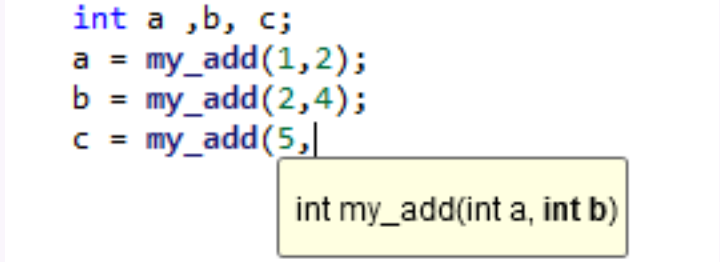
| UI Element | Description |
|---|---|
| **<Language Selection>** | Drop-down list of script languages for which you can customize appearance. |
| **<Element Selection>** | List of code elements whose appearance you can customize. |
| **Foreground color** | Enables you to select a color from the pallet. The select color is applied to the foreground of the code element. |
| **Background color** | Enables you to select a color from the pallet. The select color is applied to background of the code element. |

### Code Completion

| To access | VuGen > Tools > Options > Editor > Code Completion |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable code completion features** | Code completion features are enabled. For details on code completion, see "Editor Pane" on page 77.<br><br>By default this option is enabled. |

| | |
|---|---|
| **Enable syntax tooltip** | Enables tooltips that display function arguments. Each argument is highlighted as you are defining it, moving to the subsequent argument when the delimiter is entered.<br><br>**Example:**<br><br>  <br><br><br>By default this option is enabled. |
| **Show tooltips when mouse pointer stops over an identifier** | When this option is enabled, a description of the code elements is displayed when the mouse hovers over the identifier. This option is disabled when **Show tooltips in debug mode only** is enabled.<br><br>By default this option is enabled. |
| **Show tooltips in debug mode only** | Tooltips are displayed in debug mode only.<br><br>By default this option is disabled. |
| **Include in the code completion list** | |
| **ANSI C keywords** | Enables you to include ANSI C keywords in code completion list.<br><br>By default this option is enabled. |
| **LoadRunner API Steps** | Enables you to include LoadRunner API steps in the code completion list.<br><br>By default this option is enabled. |
| **LoadRunner API Constants** | Enables you to include LoadRunner API constants in the code completion list.<br><br>By default this option is disabled. |
| **User-defined functions** | Enables you to include user-defined functions in the code completion list.<br><br>By default this option is enabled. |

| | |
|---|---|
| **Function parameters, local and global variables** | Enables you to include function parameters, local, and global variables in the code completion list.<br>By default this option is enabled. |

### Folding

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Editor> Folding** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable folding features** | This option enables expanding and collapsing of script sections.<br>By default this option is enabled. |
| **Enable steps folding** | This option enables expanding and collapsing of script steps.<br>By default this option is disabled. |
| **When step length is more than [ ] characters** | Enables you define the number of characters in a step before implementing folding.<br>By default this option is disabled. |
| **When step consists of more than [ ] lines** | Enables you to define the number of lines to a step before implementing folding.<br>By default this option is disabled. |

## Scripting Options Tab

This pane enables you to configure options related to recording, replaying and debugging scripts.

### Recording

| | |
|---|---|
| **To access** | **Tools > Options > Scripting > Recording** |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Enable Recording Floating Toolbar transparency mode** | Display a transparent floating recording toolbar. When you click or hover on the toolbar it becomes opaque.<br>By default this option is disabled. |
| **Enable 'Cancel Recording' button** | Enable the **Cancel Recording** button on the floating recording toolbar.<br>**Note:** When the **Cancel Recording** button is enabled, there may be a delay when you start recording into an existing script. This delay occurs while VuGen makes a copy of the existing script.<br>By default, this option is enabled. |
| **Open Start Recording Dialog Box after new script is created** | Automatically open the Start Recording Dialog Box after a new script is created.<br>By default this option is disabled. |
| **Automatically close transactions** | Enable this function if you want VuGen to insert an end transaction step for open transactions before recording a subsequent action.<br>By default this option is disabled. |
| **Check the Internet connection before recording** | Enable this function if you want VuGen to check for an Internet connection before recording starts.<br>**Note:** The internet connection is checked by pinging google.com. If the Google server is not available (for example no external internet, Google is blocked, and so on) this could indicate a false-positive. In such cases, disable this check box. |
| **Check Edge recording permission before recording** | Enable this option if you want VuGen to check, before you start recording a script, if Microsoft Edge can record activity on intranet sites. |
| **Check Edge cache before recording** | Enable this option if you want VuGen, before you start recording a script, to check the Microsoft Edge cache: if the cache is not empty, some of the business process steps may not be recorded. |
| **Generate Recording Summary report** | Enables the generation of a Recording Summary report.<br>To automatically display the report after generating the code, select **Display the Recording Summary report when recording is finished**.<br>**Note:** The Recording Summary report is available for Web HTTP/HTML scripts only. |

## Replay

| To access | VuGen > Tools > Options > Scripting > Replay |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Layout** | **Do not switch the layout during Replay**. By default, when you replay a script, VuGen automatically switches the UI layout to the **Debug** layout. This option enables you to maintain your selected layout during replay.<br><br>By default this option is disabled. |
| **Animated Run** | **Animated Run**<br><br>Runs the script in animated mode, highlighting the line in the script that is currently running. In non-animated mode, VuGen runs the script, but does not indicate the line being executed.<br><br>By default this option is enabled.<br><br>**Animated Run Delay**<br><br>You can set a delay of the highlighting in the animated run, allowing you to better view the effects of each step. You set the delay in milliseconds.<br><br>The default delay is 1.<br><br>**Animate Functions in the Action section only**<br><br>Animates the content of the Action sections only, not the init or end sections.<br><br>By default this option is enabled. |
| **Results** | **Enable result of replay summary to be saved to a named folder after each script run**<br><br>When this option is enabled the dialog box prompts you to name a results file before running a script in VuGen. When not enabled, VuGen automatically names the directory 'result1'. Subsequent script runs will automatically overwrite previous results files unless you specify a different name.<br><br>**Note:** Results are stored in a subdirectory of the script.<br><br>By default this option is disabled. |

| | |
|---|---|
| **During Replay** | • **Show runtime view during replay**<br>Enables the runtime viewer<br>• **Auto Arrange Window**<br>Select this option to arrange the two viewers side by side. This option is disabled by default.<br>• **Collect replay statistics**<br>Select this option to enable the collection of replay-time statistics. The data collected is displayed in the Replay Summary report. This option is enabled by default. For details, see "Replay Summary Pane" on page 131.<br>• **Display NV Insights Report**<br>Select this option to enable the "NV Insights Report" on page 327. You open the NV Insights report from the Replay Summary page after you run a script. This option is enabled by default when the NV for Load Generator and VuGen component is installed on the VuGen machine.<br>**Note:** This option is available only if the required Network Virtualization component is installed. For details on how to install the required Network Virtualization component, see the Network Virtualization for LoadRunner & Performance Center Help. |
| **After replay** | **After replay show**<br>Instructs VuGen how to proceed after the replay:<br>• **Script.** Show script in the Editor.<br>• **Replay summary.** Go directly to the Replay Summary window in the Editor. (Default) |

### Script Management

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Scripting > Script Management** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **List of file types, by extension, that can be edited in the Editor** | Enables you to modify the list of valid file extensions that can be edited in the Editor. |

### Comparison

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Scripting > Comparison** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Path to comparing tool** | The comparison tool to be used when comparing two scripts.<br>VuGen is installed with a default comparison tool (WinMerge). To use a different comparison tool, browse to the location of the tool on your local machine. |
| **Command line arguments** - comparison tool | A list of the command line arguments to use with your comparison tool.<br>The mandatory default arguments **%1** and **%2** should not be modified. |
| **Path to merge tool** | The merge tool to use for combining files, for example, when working with Git repositories. For details, see "Working with Git" on page 143.<br>VuGen is installed with a default merge tool (WinMerge). To use a different merge tool, browse to the location of the tool on your local machine. |
| **Command line arguments** - merge tool | A list of the command line arguments to use with your merge tool.<br>The mandatory default arguments **/e /u /dl %title1 /dr %title2 %1 %2** should not be modified. |

### Step Navigator

| To access | **VuGen > Tools > Options > Scripting > Step Navigator** |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable Editor highlighting** | This option enables you to highlight filtered steps in your script. |
| **Background color** | Enables you to select a background color to apply to the filtered steps in your script. |
| **Border color** | Enables you to select a border color to apply to the filtered steps in your script. |

### Thumbnails

| To access | **VuGen > Tools > Options > Scripting > Thumbnails** |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Enable Thumbnail Explorer** | Enables slide view of generated thumbnails in the Thumbnail Explorer. Double clicking a thumbnail sets the cursor at the associated step in the script. |
| **Highlight the thumbnail associated with a step** | Sync the display of the Thumbnail Explorer while scrolling through steps in the Editor. |
| **Show important thumbnails by default** | VuGen displays thumbnails directly related to the business process and filters out less important thumbnails by default. |
| **Enable automatic creation** | Enables the automatic creation of thumbnails during the application's idle time. |
| **Cache thumbnails to script folder** | Optimize VuGen's performance by saving rendered thumbnails to a cache file. Thumbnails are loaded from the cache file after the initial generation. |

### Output Pane

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Scripting > Output Pane** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Format** | **Word wrap**<br>Enables word wrapping in the Output pane. |
| **Font** | **Text Font**<br>Enables you to select a font for the text that appears in the Output pane.<br>**Size**<br>Enables you to select a font size for the text that appears in the Output pane. |
| **Use color coding to display text in the Output pane** | Enables color coding of the text that appears in the Output pane. Color coding may slow down system response while a very large output log is displayed (rendered) in the Output pane or as you scroll through the output log.<br>**Note:** Restart VuGen for changes to this option to take effect. |

### Java

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Scripting > Java** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Eclipse IDE Location** | **Browse**<br>Enables you to set the location of the Eclipse program, **eclipse.exe**.<br>For details on supported Eclipse versions, see the System Requirements. |

### Citrix

| To access | VuGen > Tools > Options > Scripting > Citrix |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Show client during replay** | Shows the Citrix client during script replay. |
| **Show Bitmap Selection popup** | Issues a popup message when inserting a Get Text or Sync Bitmap function before selecting a bitmap or text in the snapshot. |

### Correlation

| To access | VuGen > Tools > Options > Scripting > Correlation |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable correlation from replay snapshots** | Enables the **Create Correlation** context menu item in the Replay snapshot's Response Body pane.<br>This option instructs VuGen to format the data so that it will be open for correlations. When disabled, VuGen only generates raw data which cannot be altered. |

### Snapshots

| To access | VuGen > Tools > Options > Scripting > Snapshots |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Enable snapshot viewing** | Enables the viewing of snapshots when you click on a step in the editor or step navigator. This option allows you to improve VuGen's performance when working with very large scripts. |
| **Enable enhanced XML view** | Enables the following XML viewer visual features:<br>• XML tree<br>• coloring<br>Clear this option to reduce the amount of memory consumed by the XML view. |
| **Enable snapshot caching** | Allows snapshots to be cached.<br>Clear this option where you are working with large snapshots and are running out of memory. |
| **Do not load text snapshots larger than** | Text-based snapshots are not loaded if they are larger than the specified size. |
| **Do not load binary snapshots larger than** | Binary [hexadecimal] based snapshots are not loaded if they are larger than the specified size. |
| **Do not load XML snapshots larger than** | XML-based snapshots are not loaded if they are larger than the specified size. |
| **Do not highlight XML snapshots larger than** | XML-based snapshots are not highlighted if they are larger than the specified size. |
| **Enable HTTP Page View option in the Snapshot pane** | Enables the display of HTTP protocol snapshots in the Page View. Disabled by default because it may affect VuGen performance. It is recommend to use the HTTP Data view instead. |

### Parameters

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Scripting > Parameters** |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Left parameter delimiter** | Enables you to specify a left parameter delimiter.<br>The following characters are valid: !, #, $, %, &, (, ), [, ], {, }, \|, ~, `, <, >, ?<br>**Note:** If you change the left parameter delimiter, the specified delimiter will be applied to new Vuser scripts only, not to existing scripts. |
| **Right parameter delimiter** | Enables you to specify a right parameter delimiter.<br>The following characters are valid: !, #, $, %, &, (, ), [, ], {, }, \|, ~, `, <, >, ?<br>**Note:** If you change the right parameter delimiter, the specified delimiter will be applied to new Vuser scripts only, not to existing scripts. |
| **Parameter background color** | Enables you to specify the background color for parameters in a Vuser script. |
| **Parameter border color** | Enables you to specify the border color for parameters in a Vuser script. |
| **Restore Default Delimiters** | Resets both the left and right parameter delimiters to their default values. |

### Parser

| | |
|---|---|
| **To access** | **VuGen > Tools > Options > Scripting > Parser** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable C Language Parser** | Disabling the C language parser may improve application performance when you are working with very large scripts. However, the following features will be disabled:<br>• Editing step arguments in the **Editor**<br>• Statement completion<br>• Snapshots<br>• Tasks<br>• Thumbnails<br>• Additional step-related functionality<br>This option is enabled by default. |

# Search and Replace Dialog Boxes

These dialog boxes enable you to find and replace text strings in Vuser scripts and solutions.

## Search Dialog Box

| | |
|---|---|
| **To access** | • **Search > Quick Find**<br>• **Search > Find in Files** |
| **Important information** | VuGen's find mechanism does not perform a cyclic search and will stop when it reaches the end of the last file in the specified scope.<br>**Tip:** To perform another search, click **Find Next** at the end of the search. |

User interface elements are described below (unlabeled elements are shown in angle brackets). Some of the UI elements in the table below only appear in the **Quick Find** dialog box, while others only appear in the **Find in Files** dialog box:

| UI Element | Description |
|---|---|
| **<Search-type drop-down>** | Enables you to specify the type of search to perform: **Quick Find** or **Find in Files**.<br>**Note:** The Search dialog box user interface changes depending on the selection. |
| **Find text** | The text or expression to search for. |
| | Shows the Regular Expression Builder. The Regular Expression Builder enables you to build a regular expression in the **Find text** box. |
| **Regular Expression** | Indicates that the **Find text** string is a regular expression. |
| **Scope** | The scope of the search: Selection, Current Action, Current Script, or Entire Solution. You move the slide bar to indicate your selection. |
| **Include in search** | Indicates the entities to include in the search: Code editor, snapshots, and logs. |
| **Directory** | For a "Find in Files" search, the folder that contains the files that will be searched. |
| **Options** | Additional search options, such as case and whole word matching, or the direction of the search (for a Quick Find search). |
| **Find Next** | Finds the next occurrence of the text or regular expression in the **Find text** box. |

| | |
|---|---|
| **Find All** | Finds all occurrences of the text or regular expression that appears in the **Find text** box. The results appear in the Search Results pane. In the Search Results pane, you can:<br>• Display the results by file, or as a flat list.<br>• Right-click an entry and select **Locate** to show the corresponding text in the Vuser script or wherever it is located.<br>• Right-click an entry and select **Copy** to copy the selected search result to the clipboard.<br>• Right-click and select **Copy All** to copy all the search results to the clipboard. |

## Replace Dialog Box

| | |
|---|---|
| **To access** | **Search > Quick Replace** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Find text** | Specify the text to search for. |
| | Shows the Regular Expression Builder. The Regular Expression Builder enables you to build a regular expression in the **Find text** box. |
| **Regular Expression** | Select **Regular Expression** to indicate that the **Find text** string is a regular expression. |
| **Replace with** | The text that will replace the **Find text**. |
| **Scope** | The scope of the search: Selection, Current Action, Current Script, or Entire Solution. You move the slide bar to indicate your selection. |
| **Options** | Enables you to specify search options, such as case and whole word matching, or the direction of the search. |
| **Find Next** | Finds the next occurrence of the text or regular expression in the **Find text** box. |
| **Replace** | Replaces the selected text with the text in the **Replace with** box. |
| **Replace All** | Replaces all found occurrences of the text or regular expression in the **Find text** box with the text in the **Replace with** box. |

# Incremental Search

**Ctrl+E** or **Search > Incremental Search** starts incremental searching. The floating icon appears, to indicate that the feature is active. The search begins at the cursor position.

As you type each character, the matching text is highlighted. If some text is already highlighted, it is included in the search string.

If you enter only lower case letters, the search is not case sensitive. If you enter an upper case letter, the search is case sensitive.

**Ctrl+Shift+E** or **Search > Reverse Incremental Search** starts searching up.

**Escape** or **Enter** ends the search.

# Business Process Report Dialog Box

This dialog box enables you to create a Business Process report.

| | |
|---|---|
| **To access** | **Tools > Business Process Report** |
| **Relevant tasks** | "Create a Business Process Report" on page 163 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Title** | The title that will appear on the report. |
| **Author** | Your name, as it will appear in the report. |
| **Comment** | Additional comments that you want to appear in the report. |
| **Location** | The location where you want the report to be saved.<br><br>**Default value:** Script folder. |
| More | Expands the Business Process Report dialog box to display more options. |
| **Table of contents** | Includes a table of contents in the report. If you disable this option, a table of contents will not appear in the report.<br><br>**Default value:** Enabled. |

| UI Element | Description |
|---|---|
| **Recording summary** | A summary of the recording session, as it appears in the Replay Summary pane, is included in the report.<br><br>**Default value:** Enabled. |
| **Transactions and Rendezvous lists** | The report includes a list of all of the transactions and rendezvous points that were defined in the script.<br><br>**Default value:** Enabled. |
| **Parameters list** | A list of all the parameters that were defined in the script. This list corresponds to the parameters listed in the Parameter List dialog box **Design > Parameters > Parameter List**.<br><br>**Default value:** Enabled. |
| **Thumbnails** | The report includes an actual snapshot of each recorded step, adjacent to the step name and description.<br><br>**Default value:** Enabled. |
| **Step descriptions** | The report includes a short description of each step.<br><br>**Default value:** Enabled. |
| **Script name** | The .usr file name of the script. |
| **Output format** | Creates the report in the selected format. The following formats are available:<br>• Microsoft Word<br>• Adobe PDF<br>• HTML |
| **Document template** | The path and file name of the template to use for the report. The default template is usually stored in the product's **dat** folder.<br>To change the report template, click the browse button and specify a new template with a .docx extension. If you want to create a new template, we recommend that you use an existing template as a basis for the new one. This will make sure that the required bookmarks and styles are maintained within the new template. |

# Replay Summary Pane

This pane provides summarized replay results and links to script replay details.

| | |
|---|---|
| **To access** | Use one of the following:<br>• **Solution Explorer >** Right click **Replay Runs >** and then select **Open Replay Summary**<br>• Click the **View summary** link in the **Output Pane** |
| **Important information** | To enable transaction breakdown data, select **Tools > Options > Scripting > Replay > Collect replay statistics**.<br>**Note:** Enabling the **Collect replay statistics** option will affect replay performance. |
| **See Also** | • "Output Pane" on page 100<br>• "Replay a Vuser Script" on page 296 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Results Dashboard** | Displays basic script information including:<br>• **Script name**<br>• **Replay Status:** Displays a replay status of the script as either Script Passed or Script Failed.<br>• **Elapsed time:** Total time passed during script replay.<br>• **Started at.** Starting time of the script replay.<br>• **Ended at.** Ending time of the script replay.<br>• **Think time.** Total duration of Think time passed during script replay.<br>• **Wasted time.** Total duration of Wasted time passed during script replay.<br>• **Save HAR file.** For Web HTTP/HTML scripts only. Saves the traffic information to a HAR file.<br>• **Export to PDF.** Saves the current replay summary to a PDF file.<br>• **Open NV Insights Report.** For details, see "NV Insights Report" on page 327.<br>• **Select result scope:** (Available if you enable the collection of replay statistics.) Enables you to select a specific iteration result set or to display averages of all iterations. |

| UI Element | Description |
|---|---|
| **<Script Performance>** | A list of the script's actions and transactions with basic information. Click on an action or transaction to show or hide its details below the table (only available if you enable the collection of replay statistics) .<br>• **Name:** The name of the action or transaction.<br>• **Duration:** When the scope is a single iteration, the time displayed is the duration of the transaction. When the scope is an average, the time displayed is the average duration of all iterations.<br>• **Duration Trend:** A Sparkline representation of values over all iterations (for multiple iterations only).<br>• **Status:** The number of iterations with a status of Passed/Total iterations. |
| **Connection statistics**<br>**Details** | A collection of statistics per action or transaction, as per the selected scope (the script average, a specific action, or a specific iteration).<br>**Note:** These statistics are only available for Web - HTTP/HTML protocol scripts.<br>• **Count:** The number (or average number for multiple iterations) of connections to the server per domain.<br>• **Hit Count:** The number (or average number for multiple iterations) of files requested from the server.<br>• **Hit Count %:** The amount of each hits per item (or average for multiple iterations) as a percentage of the total hits to all items. Expand the node to view all of the items.<br>• **Hit Count % Trend:** A Sparkline representation of the hit count values over all iterations.<br>• **Size:** The size (or average size for multiple iterations) of the data returned from the server per domain.<br>• **Size %:** The percentage (or average percentage for multiple iterations) of data size returned from the server per domain, of the total returned data.<br>• **Size % Trend:** A Sparkline representation of values over all iterations. |

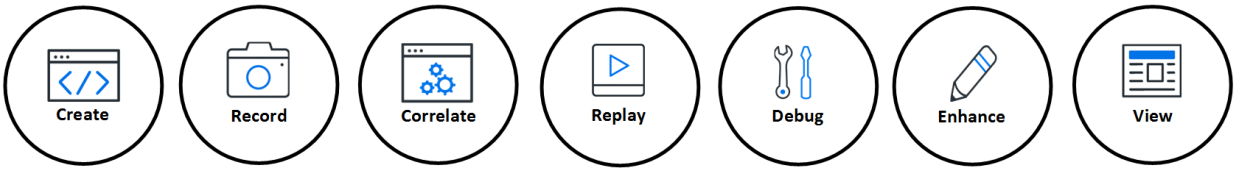
| UI Element | Description |
|---|---|
| **Responses per content-type**<br><br>**Details** | Content-type statistics per transaction or action for the current scope (the script average, a specific action, or a specific iteration).<br><br>**Note:** These statistics are only available for Web HTTP/HTML protocol scripts.<br><br>• **Responses per content-type:** A list of the content type returned from the server, for example, an image, a JavaScript, a CSS (expand to view list).<br>• **Count:** The number (or average number for multiple iterations) of connections per content type.<br>• **Count %:** The percentage (or average percentage for multiple iterations) of connections to the server content type from the total number of connections.<br>• **Count % Trend:** A Sparkline representation of the count values over all iterations.<br>• **Size:** The size (or average size for multiple iterations) of the data returned from the server per content type.<br>• **Size %:** The percentage (or average percentage for multiple iterations) of data returned from the server for each content type from the total returned data.<br>• **Size % Trend:** A Sparkline representation of values of the data size per content type over all iterations. |
| **Responses per HTTP status Details** | HTTP Status statistics per action or transaction for the current scope (the script average, a specific action, or a specific iteration).<br><br>**Note:** These statistics are only available for Web - HTTP/HTML protocol scripts.<br><br>• **Responses per HTTP Status:** List of the HTTP status codes returned .<br>• **Count:** The number (or average number for multiple iterations) of connections per HTTP status code.<br>• **Count %:** The percentage (or average percentage for multiple iterations) of connections per HTTP status code from the whole action or transaction.<br>• **Size % Trend:** A Sparkline representation of values over all iterations. |
| **Replay Statistics Summary** | A list of script statistics and their values after replay, per scope (the script average, a specific action, or a specific iteration). |

| UI Element | Description |
|---|---|
| **<Replay Button>** | Enables you to replay your script from the **Replay Summary pane**. |
| **<Modify Runtime Settings>** | Enables you to access **Runtime Settings** for your active script. |
| **<Modify Content Check Rules>** | VuGen's **ContentCheck** mechanism enables you to detect all types of errors sent by the web server.<br>For details, see **Internet Protocol > ContentCheck** view in the runtime settings. |

# VuGen Workflow

The **VuGen Workflow** section provides you with an understanding or the steps you need to follow in order to create an effective load testing script.

Select an image to learn more.




# Creating or Opening Vuser Scripts

This section describes how to create new Vuser scripts, and open existing ones.

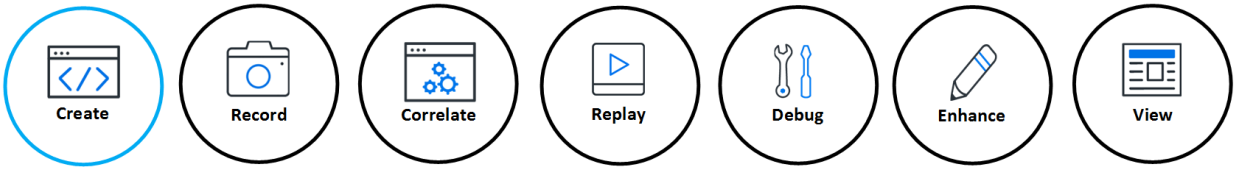
### What do you want to do?

- Learn more about creating scripts
- Create a new script
- Open a script from ALM
- Create a multiple protocol script
- Save my script to an archive file

## Creating Vuser Scripts - Overview

Creating a Vuser script includes the steps shown below. This topic provides an overview of the first step, creating a Vuser script.

Select an image to learn more.




The first step in developing a Vuser script is to create a blank script. For details on how to create a blank Vuser script, see "Create and Open Vuser Scripts" on the next page. The contents and structure of the blank Vuser script vary slightly based on the protocol of the script. Therefore, before you create a blank Vuser script, you must know the protocol to use for the script. After you create a blank Vuser script, you are ready to perform the next step in the script creation workflow - recording user actions into the script. For details, see "Recording" on page 154.

When you create a Vuser script, VuGen creates a series of configuration files, data files, and source code files that comprise the Vuser script. These files contain Vuser runtime and setup information. For details on the files that comprise a Vuser script, see "Script Directory Files" on page 150.

VuGen enables you to:

- Create or open a script from a template. For task details, see "Create and Open Vuser Script Templates" on page 153.
- Open or work with a .zip script. You can unzip or work with a script in .zip format. For task details, see "How to Work with .zip Files" on page 152.
- Work withApplication Lifecycle Management:
  - Open a script stored in Application Lifecycle Management.
  - Use Application Lifecycle Management to store and retrieve Vuser scripts, scenarios, and analysis results. You can store scripts in an ALM project and organize the scripts into unique groups.

  For more information, see "Working with Application Lifecycle Management" on the next page.

## Create and Open Vuser Scripts

| To: | Do this: |
|---|---|
| **Create a new Vuser script** | 1. Open VuGen and select **File > New Script and Solution**.<br>2. In the **Create a New Script** dialog box, select **Single Protocol** or **Multiple Protocols** from the **Category** list.<br>3. Select a protocol from the **Protocols** list.<br>4. In the **Script Name** box, enter a name for the script.<br>**Note:** Do not name scripts *init*, *run* or *end* because these names are used internally by VuGen.<br>5. Click **Create** to create the Vuser script.<br>For user interface details, see "Create a New Script Dialog Box" on page 150.<br>After you create a new Vuser script, you can record user activity into the script. For details, see "Record a Vuser Script" on page 158. |
| **Create or open a script from a template** | For task details, see "Create and Open Vuser Script Templates" on page 153. |
| **Open an existing script stored on the local machine or network drive** | Select **File > Open > Script/Solution**. |
| **Open or work with a .zip script** | For task details, see "How to Work with .zip Files" on page 152. |

| To: | Do this: |
|---|---|
| **Open a script stored in Application Lifecycle Management** | You can store scripts on ALM and modify them in VuGen. For details, see "Working with Application Lifecycle Management" below. |

## Compare Scripts Side-by-Side

Vuser scripts can be compared and displayed side by side using the comparison tool.

1. Right-click the primary script in **Solution Explorer** and select **Set as first comparing object**.
2. Right-click the secondary script in the **Solution Explorer** and select one of the following:
   - **Compare**
   - **Compare with external object** to compare an asset to a file outside of the solution. This sets the highlighted asset as the primary asset and opens Windows Explorer to enable you to select the secondary asset.

> **Note:** You can change the comparison tool from **Options > Scripting> Comparison**. For more information, see "Scripting Options Tab" on page 119.

## Working with Application Lifecycle Management

This section describes how to manage your Vuser scripts by integrating with Application Life Cycle Management.

### Managing Scripts Using ALM - Overview

VuGen works together with Application Lifecycle Management (ALM). ALM provides efficient functionality for storing and retrieving Vuser scripts, scenarios, and analysis results. You can store scripts in an ALM project and organize the scripts into unique groups.

In order for VuGen to access an ALM project, you must connect VuGen to the Web server on which the ALM project is located. You can connect to either a local or remote Web server.

For details on working with ALM, see the ALM Help Center.

### Connect to ALM

To store and retrieve scripts from ALM, you need to connect to an ALM project. You can connect or disconnect from an ALM project at any time during the testing process.

You can connect to one version of ALM from VuGen and a different version from your browser. For more information, see the **Important Information** section in "ALM Connection Dialog Box [VuGen]" on page 141.

To connect to a project in ALM:

1. Determine the type of authentication required for the ALM server: User name/password or CAC (Common Access Card). For CAC mode, enable CAC authentication in VuGen's **General** options. For details, see "General Options Tab" on page 109.
2. Select **Version Control > ALM** > **ALM Connection**. The ALM Connection dialog box opens.
3. In the **Step 1: Connect to server** section, enter a user name and password (not relevant for CAC authentication) and click **Connect**. VuGen connects to the ALM server.

   To disconnect from ALM, click **Disconnect**.
4. In the **Step 2: Login to project** section, enter the domain and project details, and then click **Login**. VuGen logs in to the specified project.

   To log out of the project, click **Logout**.
5. Click **Close** to close the ALM Connection dialog box.

> **Note:** If you authenticated through CAC mode and disconnected from the ALM server, you need to restart VuGen before reconnecting in CAC mode.

## ALM Version Control - Overview

VuGen supports version control features in Vuser scripts saved in ALM projects that use version control.

The version control features change the process of opening and saving a script. Scripts with version control are either in a state of checked-in or checked-out. When you are working with a script in a checked-out state, any changes you make will not be saved on the ALM server until you check in the script. If you save the script from within VuGen, a temporary file is saved on your machine that protects your changes in case your computer crashes.

If you are working with a script in a checked-in state, the script is read-only and you cannot make any changes until you check out the script.

If a particular script is being saved to ALM for the first time, and the project uses version control, the script automatically starts in a checked-out state.

## Work with Scripts in ALM Projects

The following steps describe the workflow of how to work with Vuser scripts that are saved in an ALM project.

> **Note:** To work with scripts in ALM projects with version control, see "Work with Version-Controlled Scripts in ALM Projects" on the next page.

1. Connect to ALM

   Open a connection to the ALM server and project that contains the script. For details, see "Connect to ALM" on the previous page.
2. Open the script

Select **File > Open > Script/Solution**. In the Open VuGen Script or Solution dialog box, select the script to open and then click **Open**.

3. Save the script

   Select **File > Save Script**. If the script is in a project that uses version control and is not checked out, the script is saved as a temporary file on your local machine.

## Work with Version-Controlled Scripts in ALM Projects

> **Relevant for:** Scripts in ALM projects that support version control AND have the Performance Center addition installed.
>
> If these two conditions are not met, see "Work with Scripts in ALM Projects" on the previous page.

The following steps describe how to work with scripts saved in ALM projects for which version control is enabled.

1. Connect to ALM.

   Open a connection to the ALM server and project that contains the script. For details, see "Connect to ALM" on page 138.

2. Open the script.

   Select **File > Open > Script/Solution**. In the Open VuGen Script or Solution dialog box, select the script to open and then click **Open**.

3. Check in/out the script.

   If the ALM project has version control, each script is always defined as being either checked-in or checked-out. For more details, see "ALM Version Control - Overview" on the previous page. To check in and check out scripts, select **Version Control > ALM > Check In/Check Out**.

   > **Note:** If the ALM project has version control, the file is locked when it is checked out.
   >
   > If the ALM project is not version controlled, the file is not locked when checked out of the project.

4. Cancel a check out. (Optional)

   If you checked out a script and do not want to save the changes, you can return the status of the script to checked-in without saving by selecting **Version Control > ALM > Undo Check Out**.

5. Save the script.

   Select **File > Save Script**. If the script is in a project that uses version control and is not checked out, the script is saved as a temporary file on your local machine.

## Save VuGen Vuser Scripts to ALM Projects

The following steps describe how to save a Vuser script to an ALM project.

1. Open or create the Vuser script

   Create or open the desired script in VuGen.

2. Connect to ALM

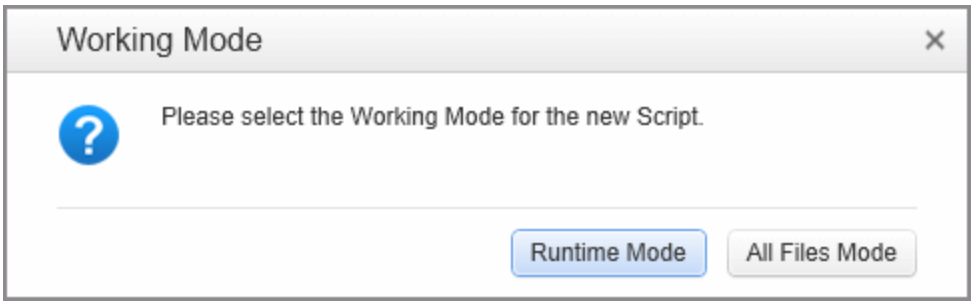
   Open a connection to the ALM server and project in which you want to store the script. For details, see "Connect to ALM" on page 138.

3. Save the script to ALM

   a. Select **File > Save Script as**. The Save Script As dialog box opens.

   b. Click **ALM Test Plan**, and then specify the name and location for the script.

   c. Click **Save**. After a short time, the Working Mode dialog box opens.




   d. Select one of the following options:

   **Runtime Mode.** Copies only the files needed to replay the script. This option does not copy recording snapshot files and other unnecessary files. This results in a shorter transfer time.

   **All Files Mode.** Copies all of the files associated with this script. This results in a longer transfer time.

## Compare Previous Versions of a Script

If your Vuser script is saved in an ALM project that uses version control, you can compare previous versions of the script. The following steps describe how to do this.

1. Connect to ALM

   Open a connection to the ALM server and project that contains the script that you want to view or modify. For details, see "Connect to ALM" on page 138.

2. Open the script

   Select **File > Open > Script/Solution**. In the Open VuGen Script or Solution dialog box, select the script to open and then click **Open**.

3. Compare previous versions of the script

   a. Select **Version Control > ALM > Version History**. The Version History dialog box opens.

   b. Select two previous versions of the script and then click **Compare Versions**. WDiff opens and displays the two versions of the script.

## ALM Connection Dialog Box [VuGen]

This dialog box enables you to connect to an ALM project from within VuGen.

| To access | Version Control > ALM > ALM Connection |
|---|---|
| **Important information** | • You can connect to one version of ALM from VuGen and a different version of ALM from your browser.<br>• You can connect to different versions of ALM only if one of the versions is ALM 11.00 or higher.<br><br>**Note:** Before you connect to results stored on ALM through the this dialog box, it is recommended that you first connect to the ALM server through your browser. This automatically downloads the ALM client files to your computer.<br><br>• You can configure more advanced settings, such as a proxy setting, by using the Webgate Customization tool (webgatecustomization.exe) and then sign into ALM using the ALM Connection Dialog Box. The Webgate Customization tool can be found on your ALM server at the following address: `http://<ALM Server>/qcbin/Apps/`. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Step 1: Connect to Server** | • **Server URL.** The URL of the server on which ALM is installed.<br>• **User name.** Your ALM project user name (not relevant for CAC authentication).<br>• **Password.** Your ALM project password (not relevant for CAC authentication).<br>• Connect. Connects to the server specified in the **Server URL** box.<br>• Disconnect. Disconnects from the current ALM server. |
| **Step 2: Login to Project** | • **Domain.** The domain that contains the ALM project. Only those domains containing projects to which you have permission to connect to are displayed.<br>• **Project.** Enter the ALM project name or select a project from the list. The list includes only those projects to which you have permission to connect.<br>• Login. Logs into the ALM project.<br>• Logout. Logs out of the current ALM project. |
| **Restore connection on startup** | Automatically reconnect to the ALM server the next time you start VuGen, using the same credentials. |

# Working with Git

The following sections describe how to work with a Git repository to store and retrieve your scripts.

## Managing Scripts with Git - Overview

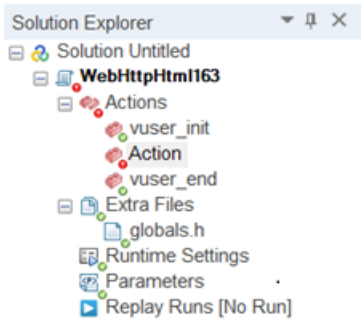
VuGen integrates with Git Hub, allowing you to upload scripts from a Git repository and perform common actions such as Pull, Push and Track.

VuGen's **Version Control** menu allows you to import a script from a Git repository. For details, see "Import from Remote Repository Dialog Box" on page 149.

You can also take a local script and add it to a Git repository, directly from the Solution Explorer. Using the right-click menu, you can perform typical "Git Operations" on the next page. You can also view history and manage your changes.

To limit the files added to your repository, VuGen uses a **.gitignore** file, which can be edited. For details, see "Configure Ignore List Dialog Box" on page 147.

For scripts stored in a Git repository, the Solution Explorer pane indicates which files changed since the last commit.




For details about working with a Git repository script, see "Work with Scripts in Git" below.

## Work with Scripts in Git

The following steps describe the workflow of how to work with Vuser scripts and a Git repository. For an overview of the GitHub integration, see "Managing Scripts with Git - Overview" above.

1. **Open or create a Git repository script in VuGen**
   - **For a local script not part of a Git Repository:**
     Create or open a local script in VuGen. Click the parent level of the script in the Solution Explorer and select **Git > Create Local Git Repository**. The Track Script dialog box opens. Accept the default gitignore content, or click **Configure Ignore List** to edit the list or use an external file. For details, see "Configure Ignore List Dialog Box" on page 147. After you edit the ignorelist, click **Track**.
   - **For a script stored in a Git Repository:** Download the script from the Git repository.
     Select **Version Control > Git Options > Import from Remote Repository**. Specify the repository

information and click **Import**. The script opens in the VuGen editor. For details, see "Import from Remote Repository Dialog Box" on page 149.

2. **Edit the script**

   Edit or record the script as you would any other script.

3. **Perform a Pull operation (optional)**

   Click the parent level of the script in the Solution Explorer and expand the **Git** entry from the right-click menu. Select **Pull** to retrieve the updated files from the Git repository. For details, see "Git Operations" below. If there are conflicts between the files, VuGen opens the "Resolve Conflicts Dialog Box" on page 148.

4. **Commit the changes**

   Save the file in VuGen. Click the parent level of the script in the Solution Explorer and select **Git > Commit...** from the right-click menu. In the Commit Changes dialog box, specify the change's details and click **Commit**.

5. **Push the changes to the Git repository**

   Click the parent level of the script in the Solution Explorer and expand the **Git** entry from the right-click menu. Select **Push** to send the changes to the Git repository. For details, see "Git Operations" below. If there are conflicts between the files, VuGen opens the "Resolve Conflicts Dialog Box" on page 148.

6. **Track or untrack scripts (optional)**

   To disable tracking for a script, click its parent level in the Solution Explorer and select **Git > Untrack Script**. To re-enable tracking, select **Git > Activate Script Tracking**.

For additional options, such as undoing changes or viewing history, see "Git Operations" below.

### Troubleshooting

If you are unable to perform push, pull, or import operations:

- Make sure you have an active Internet connection.
- Make sure that a proxy server is not blocking the connection. Configure the Git machine to allow access via your proxy server. For details, see the Git documentation.

If you push to a remote repository and then change the connection to use a second remote repository, and push the commits to that second repository, the push fails. To push to the second repository:

1. Change the connection to point to the second repository.
2. Perform a new commit.
3. Push to the new repository.

## Git Operations

VuGen allows you to perform Git operations from the script's context (right-click) menu.

| | |
|---|---|
| **To access** | 1. Open a script stored in a Git repository.<br>• For a new script that is not part of a Git repository, select **Git > Create local Git repository** from right click menu.<br>• For a script that already exists in GitHub, import it, as described in "Import from Remote Repository Dialog Box" on page 149.<br>2. In the Solution Explorer pane, click the parent level of the script and expand the **Git** item in the right-click menu. |
| **Important information** | • If the script is on a private repository and you need to Pull or Push, you must provide credentials.<br>• Make sure your machine is configured to integrate with Git and that you set the relevant proxy settings, if applicable. |
| **Relevant tasks** | • "Work with Scripts in Git" on page 143 |
| **See also** | • "Import from Remote Repository Dialog Box" on page 149<br>• "Configure Ignore List Dialog Box" on page 147 |

The Git context menu operations are described below in alphabetical order:

| UI Element | Description |
|---|---|
| **Activate Script Tracking** | (Available only for untracked scripts) Adds the current script to the staging area for tracking. To ignore specific files from the tracking, use the **Configure Ignore List** option.<br> <br>**Note:** If you rename a tracked file outside of VuGen, the Solution Explorer will display its new name. |
| **Check for Modifications...** | Displays the changes to the script files since your last commit. Double-click on an entry to open the changes in a Diff tool.<br>Expand the **Internal files** and **User files**sections to see which files were modified. |
| **Commit...** | Opens the **Commit Changes** dialog box, allowing you to:<br>• Type a description of the change in the **Message** area.<br>• See which files will be committed. Expand the **Internal files** and **User files**sections to see the file list.<br>• Edit the signature for the change by clicking **Edit...**.<br>• Add the script files to the staging area and commit the changes with the specified message. |

| UI Element | Description |
|---|---|
| **Configure Ignore List...** | Allows you to modify the **.gitignore** content or import the contents of a previously saved **.gitignore** file.<br><br>Place each file that you want to ignore on a separate line. Use an asterisk as a wildcard. For complete syntax information, see the Git documentation. |
| **Connect to Git...** | Opens the **Connect to Git** dialog box, allowing you to specify the connection information:<br><br>• **Remote Repository URL.** The URL of the repository to use for Push, Pull, and Commit operations.<br>• **User name**, **Password**. The credentials to use for connecting to the repository. They are saved internally to ensure persistent connectivity. |
| **Create Local Git Repository...** | Opens the Track Script dialog box. Click **Track** to add the local script to a Git repository.<br><br>To indicate which script files to exclude from the repository, click **Configure ignored list** to open the "Configure Ignore List Dialog Box" on the next page. |
| **Pull** | Pulls the changes from the repository to your local script files. |
| **Push...** | Opens the **Push Changes** dialog box, when changes are detected. If you are not connected to Git, VuGen opens the Connect to Git dialog box, prompting you for credentials.<br><br>In the **Changes** section, you select the change that you want to push to the repository.<br><br>Expand the **Internal files** and **User files**sections to see which files will be included in the push operation. |
| **Refresh** | Recalculates and updates the tree icons in VuGen's **Solution Explorer** pane, with the latest actions and files. |
| **Resolve Conflicts...** | Opens the **Resolve Conflicts** dialog box, to help you resolve conflicts between local files and those on the repository.<br><br>Select the check box adjacent to each file whose conflict you want to mark as resolved.<br><br>Double-click on a conflicted file to open the merge tool.<br><br>For additional details, see "Resolve Conflicts Dialog Box" on page 148. |

| UI Element | Description |
|---|---|
| **Reset HEAD** | Sets the files in the working directory and staged snapshot of the current branch, HEAD, to their content at your last commit.<br><br>**Caution:** This operation will discard all of your uncommitted changes. |
| **Revert...** | Allows you to revert to a specific commit.<br><br>In the **Revert a Commit** dialog box, select a commit from the list and click **Revert**. The revert command will only undo the changes made in the selected commit. All other commits are unaffected.<br><br>**Note:** When reverting a commit that has more than one parent, the revert will use the first parent. |
| **Show log...** | Opens the **Log** dialog box, listing the changes and their associated files.<br><br>In the **Files** section, expand the **Internal files** and **User files**headers to see the files that were modified for the selected change. |
| **Untrack Script** | (Available for a tracked script only) Removes the script and all its files from the list of tracked files. |

## Configure Ignore List Dialog Box

This dialog box allows you to set the gitignore content for the current script.

| | |
|---|---|
| **To access** | Right-click the script's parent node in the Solution Explorer, and select **Git > Configure Ignore List** from the context menu (for scripts already in the repository). |
| **Relevant tasks** | "Work with Scripts in Git" on page 143 |
| **See also** | "Git Operations" on page 144 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<ignore file list>** | An editable area for listing the files to ignore. Use an asterisk (*) as a wildcard. Place each entry on a separate line. |
| **Import** | Allows you to import an existing .gitignore file to the script. |

| | |
|---|---|
| **Select** | Opens the Ignore File Templates dialog box, allowing you to select a predefined template for an ignore list. You can select one of the following templates:<br>• **Default template.** Ignores all temporary files and those generated automatically by VuGen.<br>• **Replay Files template.** Ignores all files that are not required for replay. |
| **Wrap text** | Wraps the text in the ignore file list to the next line. |

## Resolve Conflicts Dialog Box

This dialog box allows you to import scripts or resources from a Git repository.

| | |
|---|---|
| **To access** | 1. Open a script stored in a Git repository.<br>• For a new script that is not part of a Git repository, select **Git > Create local Git repository** from right click menu.<br>• For a script that already exists in GitHub, import it, as described in "Import from Remote Repository Dialog Box" on the next page.<br>2. In the Solution Explorer pane, click the parent level of the script and expand the **Git** item in the right-click menu and select **Resolve Conflicts**.<br>This dialog box will also open automatically if you attempt to perform a **Pull** operation for files that are in conflict. |
| **Relevant tasks** | "Work with Scripts in Git" on page 143 |
| **Important Information** | You set a merge tool from the "Options Dialog Box" on page 109 (**Tools > Options > Scripting > Comparison**). Double click an entry to open it in the merge tool. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **<file list>** | A list of **User** and **Internal** files that are in conflict.<br><br>Double-click on a file to open a merge tool that will allow you to resolve the conflict. For details about setting up a merge tool, see above. |
| **Mark as Resolved** | Marks the files whose check boxes are selected, as resolved.<br>To mark all files as resolved, choose **Select all files**. |

## Import from Remote Repository Dialog Box

This dialog box allows you to import scripts or resources from a Git repository.

| | |
|---|---|
| **To access** | **Version Control > Git Options > Import from Remote Repository** |
| **Relevant tasks** | "Work with Scripts in Git" on page 143 |
| **See also** | "Git Operations" on page 144 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Remote repository URL** | The URL of the repository from which you want to import the resource. |
| **Repository name** | The name of the repository to import. |
| **Local Path** | The local path on which to save the Git resources. |

# Multiple Protocol Scripts

When you record a single protocol, VuGen records only the specified protocol. When you record in multi-protocol mode, VuGen records the actions in several protocols. To see which Vuser types are supported for multi-protocol recording, click the **Multiple Protocols** node in the Create a New Script dialog box. For details, see "Create a New Script Dialog Box" on the next page.

Another variation between Vuser types is multiple-action support. Most protocols support more than one action section. Currently, the following protocols support multi-actions: Oracle NCA, Web - HTTP/HTML, RTE, and C Vusers.

For most Vuser types, you create a new Vuser script each time you record—you cannot record into an existing script. However, when recording a Java, Web - HTTP/HTML, Oracle NCA, or RTE Vuser script, you can also record within an existing script.

Since VuGen supports a large variety of protocols, some of the recording steps that follow apply only to specific protocols.

For protocol-specific information, see:

| | |
|---|---|
| Java | "Java Record Replay Protocol" on page 567 |
| MQTT | "MQTT Protocol" on page 605 |

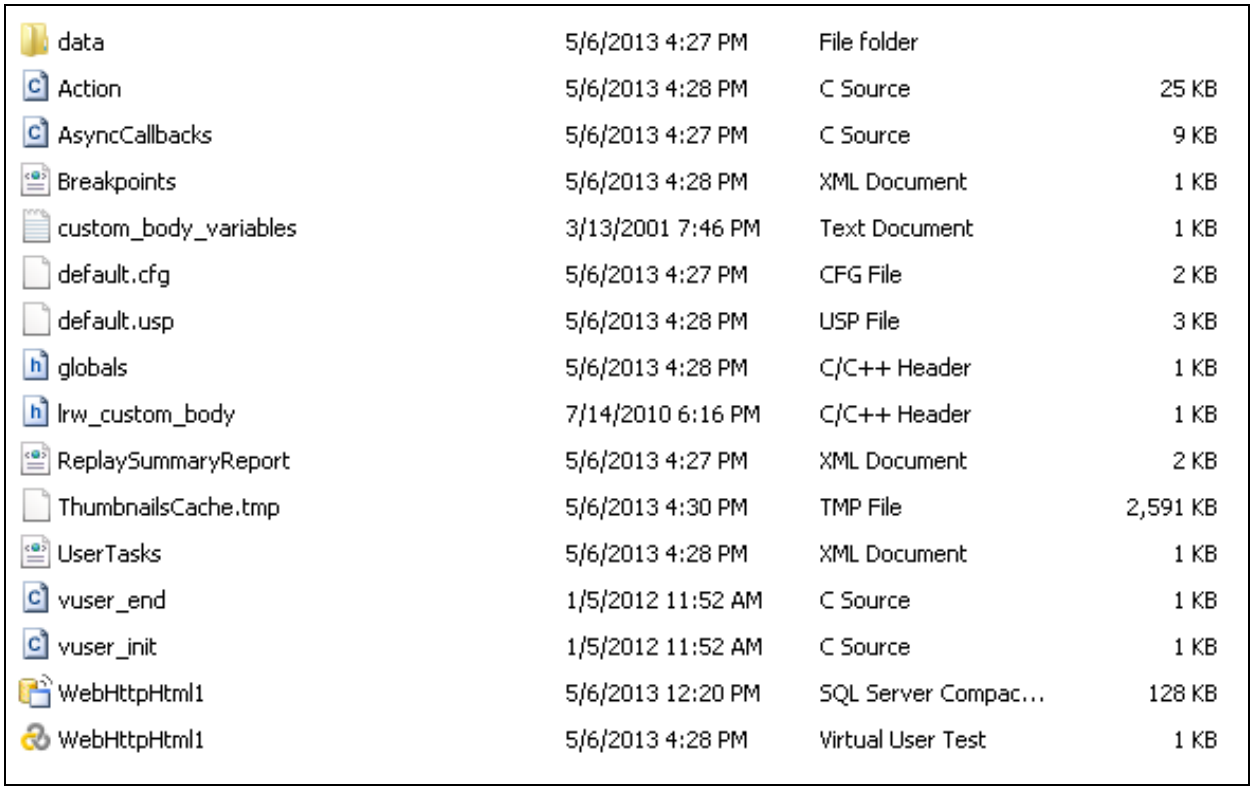
In SOA (Service Oriented Architecture) systems, it is essential that you test the stability of your applications and services before deployment. VuGen allows you to create basic Web Service scripts. **Unified Functional Testing (UFT)**, Micro Focus's functional testing tool, contains additional features that help you create a comprehensive testing solution for your SOA environment. For more information, contact an Customer Support representative.

# Script Directory Files

While you create a Vuser script, VuGen creates a series of configuration files, data files, and source code files that comprise the Vuser script. These files contain Vuser runtime and setup information. VuGen saves these files together with the script in the script folder.

## To access the files in the script folder:

Right-click the script name in the **Solution Explorer** and select **Open Script Folder**.



| | | | |
|---|---|---|---|
| data | 5/6/2013 4:27 PM | File folder | |
| Action | 5/6/2013 4:28 PM | C Source | 25 KB |
| AsyncCallbacks | 5/6/2013 4:27 PM | C Source | 9 KB |
| Breakpoints | 5/6/2013 4:28 PM | XML Document | 1 KB |
| custom_body_variables | 3/13/2001 7:46 PM | Text Document | 1 KB |
| default.cfg | 5/6/2013 4:27 PM | CFG File | 2 KB |
| default.usp | 5/6/2013 4:28 PM | USP File | 3 KB |
| globals | 5/6/2013 4:28 PM | C/C++ Header | 1 KB |
| lrw_custom_body | 7/14/2010 6:16 PM | C/C++ Header | 1 KB |
| ReplaySummaryReport | 5/6/2013 4:27 PM | XML Document | 2 KB |
| ThumbnailsCache.tmp | 5/6/2013 4:30 PM | TMP File | 2,591 KB |
| UserTasks | 5/6/2013 4:28 PM | XML Document | 1 KB |
| vuser_end | 1/5/2012 11:52 AM | C Source | 1 KB |
| vuser_init | 1/5/2012 11:52 AM | C Source | 1 KB |
| WebHttpHtml1 | 5/6/2013 12:20 PM | SQL Server Compac... | 128 KB |
| WebHttpHtml1 | 5/6/2013 4:28 PM | Virtual User Test | 1 KB |

## See also:

- "Files Generated During Recording" on page 246

# Create a New Script Dialog Box

This dialog box enables you to create a new Vuser script.

| | |
|---|---|
| **To access** | Do one of the following:<br>• **File > New Script and Solution**<br>• **File > Add > New Script**<br>• Click the button in VuGen. |

| | |
|---|---|
| **Relevant tasks** | • "Creating or Opening Vuser Scripts" on page 136<br>• "Record a Vuser Script" on page 158<br>• "How to Record a Script with TruClient - Mobile Web" on page 602 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Category / Protocol** | The protocols to display in the **Protocol** pane and the type of script to create (for all except for User Templates):<br>• **User Templates**. Displays any user-defined templates you have created. Available only if at least one user-defined template exists. For details, see "Create and Open Vuser Script Templates" on page 153.<br>• **Single Protocol.** Shows all of the protocols. When you click **OK**, it creates a single protocol script using the selected protocol.<br>• **Multiple Protocols.** Shows all of the protocols that can be included in a multi-protocol script. Select the check boxes adjacent to the protocols that you want to include.<br>• **Mobile and IoT.** Shows all of the protocols for creating a script for a mobile application or Internet of Things (IoT) application.<br>**Mobile.** For details, see "Select a Recording Method for Mobile Applications" on page 599.<br>**IoT.** For details, see "MQTT Protocol" on page 605.<br>• **Popular.** Shows a list of the most commonly used protocols.<br>• **Recent.** Shows a list of the most recently used protocols. |
| **Filter** | Enables you to filter the protocol list by entering text. For example, if you type "Java" into the Filter box, the Protocol list will display only those protocols that include the word Java. |
| **Script Name** | Enables you to specify the name of your script.<br>If you create a single protocol script,the default name is `<protocol name>x` where x represents the numerical sequence of the script created. For example, the name of the third script created for the Web- HTTP/HTML protocol would be WebHttpHtml3.<br>If you create a multi-protocol script, the default name is `<protocol name_multi>x` where protocol name is the first protocol you selected from the list and x represents the numerical sequence of the script created. |
| **Location** | Enables you specify the file location of your script. You can use the browse button to navigate to a location on your file system.<br>**Tools > General > Projects and Solutions** enables you to specify a default location. |

| | |
|---|---|
| **Solution Name** | This option is displayed only when a solution is not open in the Solution Explorer. You can specify a name for the solution. If you leave it blank, the default name is 'Untitled'. |
| **Create folder for solution** | Enables you to create a folder for your solution. |
| **Solution Target** | Displays the file path of the solution. |
| | Displays the protocols in list view. |
| | Displays the protocols in icon view. |

## How to Work with .zip Files

Compressing script files into a .zip file conserves disk space and makes it easier to copy or transfer your scripts from one machine to another.

| | |
|---|---|
| Import from a .zip file | To open a script stored in a .zip file, select **File > Manage Zip Files > Import from Zip File**. After you select a .zip file, VuGen prompts you for a location in which to store the unzipped files.<br>**Note:** If you import a script from an archive file that was archived only with runtime files, you will not be able to regenerate the script to its original recorded state. For details, see "Regenerate a Vuser Script" on page 239. |
| Export to a .zip file | To save the entire script folder as a .zip file, select **File > Manage Zip Files > Export to Zip File**.<br>You can indicate whether to save all the files or only the runtime files.<br>**Note:** If you export the script with the runtime files only, the user of the imported script will not be able to regenerate the script to its original recorded state. |
| .zip and Email | To create a .zip file and send it as an email attachment, select **File > Manage Zip Files > Zip and Email**. When you click **OK** in the **Zip and Email** dialog box, VuGen compresses the file according to your settings and opens an email compose form with the .zip file as an attachment. |
| Edit script in .zip file | To work from a .zip file, while not expanding or saving the script files, select **File > Manage Zip Files > Edit Script in Zip File**. When you modify the script and save it, the changes are stored directly in the .zip file. |

# Vuser Script Templates

The User-Defined Template enables you to save a script with a specific configuration as a template. You can then use this template as a basis for creating future scripts.

User-defined templates support the following files and data:

- Runtime settings
- Parameters
- Extra files
- Actions
- Snapshots

Generic settings, such as Recording options and General options, are not supported as they are not relevant to a specific script.

## Template Guidelines

- If you configure a script for a specific protocol and then save the script as a template, further scripts based on that template will work only with that same protocol.
- After you create a template, you cannot edit it directly in VuGen. To make any changes, open the template and save it again with another name or overwrite the existing template.
- If you regenerate an original script from a template, you will lose all of your manual changes.

# Create and Open Vuser Script Templates

This topic describes how to work with Vuser script templates.

## Create a Vuser Script Template

1. Open a script in VuGen.
2. Select **File > User-Defined Templates > Export to Template**.
3. Enter a name and location for the template.
4. Click **OK** to create the template.

## Create a Vuser Script From a Template

Select **File > New Script and Solution > VuGen > User Templates** and select the template file (only available after your create at least one template).

## Rename a Vuser Script Template

1. Select **File > User-Defined Templates > Manage from Explorer**
2. In the Explorer dialog box:

a. Rename the content file (.zip).
b. Rename the description file (.xpt).

3. Using a text editor, modify the following tags in the .xpt file:
   a. **Name tag**: `<Name>NewName </Name>`
   b. **File tag src property:** `<File name="template_temp.zip" src="NewName.zip" binary="True" />`

### Template Guidelines

- If you configure a script for a specific protocol and then save the script as a template, further scripts based on that template will work only with that same protocol.
- After you create a template, you cannot edit it directly in VuGen. To make any changes, open the template and save it again with another name or overwrite the existing template.
- If you regenerate an original script from a template, you will lose all of your manual changes.

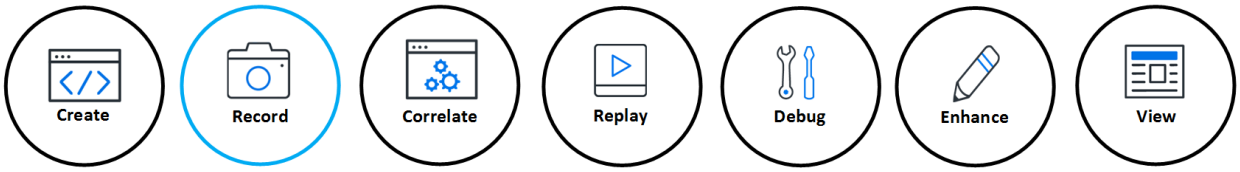
# Recording

The **Recording** section describes script sections, script recording, working with templates, and other recording tools.

## Recording - Overview

Creating a Vuser script includes the steps shown below. This topic provides and overview of the second step, recording a Vuser script.

Select an image to learn more.




After you create an empty Vuser script, you are ready to use VuGen to record typical user-actions into the script. While you record the script, VuGen's floating Recording toolbar gives you access to the main recording functionality, such as pausing and stopping the recording, and inserting transactions and rendezvous points. For details on how to record a Vuser script, see "Record a Vuser Script" on page 158.

Each Vuser script contains at least three sections: vuser_init, one or more action sections, and vuser_end. When you run multiple iterations of a Vuser script, only the *Actions* sections of the script are repeated—the *vuser_init* and *vuser_end* sections are not repeated. Before you record, and during recording, you can select the section of the script into which VuGen will insert the recorded functions. For details on the script sections, see "Vuser Script Sections" on the next page.

Before you start recording, make sure that the recording options are set correctly for the script. For more information about the recording options, see "Recording Options" on page 164.

When you have finished recording the user actions, VuGen generates the Vuser script and performs various other post-recording operations. You can replay the script to make sure that it functions correctly. For details, see "Replaying" on page 295.

To resolve situations where you cannot install VuGen on the client machine, VuGen allows you to record scripts using a LoadRunner proxy. Proxy recording may be required with certain Linux machines, Mac OS machines, and mobile devices. For details, see "Recording via a Proxy - Overview" on page 233

After you successfully record a Vuser script, you can replay the script. For details, see "Replay Overview" on page 295.

### See also:

- "Vuser Script Sections" below

## Vuser Script Sections

Each Vuser script contains at least the following sections:

| Script Section | Used when recording... | Is executed when... |
|---|---|---|
| *vuser_init* | a login to a server | the Vuser is initialized (loaded) |
| *Actions* | client activity | the Vuser is in **Running** status |
| *vuser_end* | a logoff procedure | the Vuser finishes or is stopped |

Before and during recording, you can select the section of the script into which VuGen will insert the recorded functions.

When you run multiple iterations of a Vuser script, only the *Actions* sections of the script are repeated—the *vuser_init* and *vuser_end* sections are not repeated. For more information on the iteration settings, see the General > Run Logic view in the Runtime settings.

### VuGen Script Editor

You use the VuGen script editor to display and edit the contents of each of the script sections. You can display the contents of only a single section at a time. To display a section in the script editor, double-click the name of the section in the Solution Explorer.

### Java Classes

When working with Vuser scripts that use Java classes, you place all your code in the Actions class. The Actions class contains the following methods: init, action, and end. These methods correspond to the sections of scripts developed using other protocols—you insert initialization routines into the init method, client actions into the action method, and log off procedures in the end method.

```
public class Actions{
    public int init() {
```

```
        return 0;}
    public int action() {
        return 0;}
    public int end() {
        return 0;}
}
```

For more details, see "Java Vuser (Manual) Protocol" on page 583.

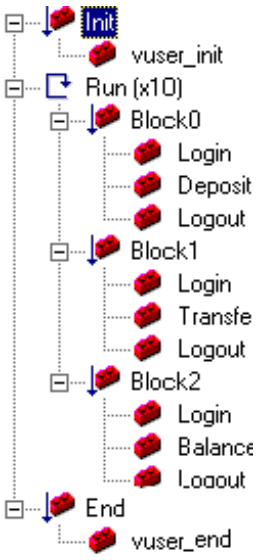
### Script Section Structure Example

Every Vuser script contains three sections: *vuser_init*, *Run (Actions)*, and *vuser_end*. You can instruct a Vuser to repeat the *Run* section when you run the script. Each repetition is known as an *iteration*.

The *vuser_init* and *vuser_end* sections of a Vuser script are not repeated when you run multiple iterations.

When you run scripts with multiple actions, you can indicate how to execute the actions, and how the Vuser executes them:

In the following example, *Block0* performs a deposit, *Block1* performs a transfer, and *Block2* submits a balance request. The *Login* and *Logout* actions are common to the three blocks.




**Sequence.** You can set the order of actions within your script. You can also indicate whether to perform actions sequentially or randomly.

**Iterations.** In addition to setting the number of iterations for the entire *Run* section, you can set iterations for individual actions or action blocks. This is useful, for example, in emulating a commercial site where you perform many queries to locate a product, but only one purchase.

**Weighting.** For action blocks running their actions randomly, you can set the *weight* or percentage of each action within a block.

In most cases, the name of the header file corresponds to the prefix of the protocol. For example, Database functions that begin with an **lrd** prefix, are listed in the **lrd.h** file.

### Header Files

Header files commonly contain forward declarations of classes, subroutines, variables, and other identifiers. In most cases, the name of the header file corresponds to the prefix of the protocol. For example, Database functions that begin with an **lrd** prefix, are listed in the **lrd.h** file.

The following table lists the header files associated with the most commonly used protocols:

| Protocol | File |
|---|---|
| Ajax (Click & Script) | **web_ajax.h** |
| Citrix | **ctrxfuncs.h** |
| COM/DCOM | **lrc.h** |
| Database | **lrd.h** |
| FTP | **mic_ftp.h** |
| General C function | **lrun.h** |
| IMAP | **mic_imap.h** |
| LDAP | **mic_mldap.h** |
| MAPI | **mic_mapi.h** |
| Oracle NCA | **orafuncs.h** |
| POP3 | **mic_pop3.h** |
| RDP | **lrrdp.h** |
| SAP GUI | **as_sapgui.h** |
| Siebel | **lrdsiebel.h** |
| SMTP | **mic_smtp.h** |
| Terminal Emulator | **lrrte.h** |
| Web (HTML\HTTP) | **as_web.h** |
| Web (Click & Script) | **web_api.h** |
| Web Services | **wssoap.h** |
| Windows Sockets | **lrs.h** |

# Record a Vuser Script

1. Create a new script or open an existing script. For details, see "Creating or Opening Vuser Scripts" on page 136.
2. (Recommended) Modify the Windows DEP settings.
   a. Open **Start > Control Panel > System**.
   b. In the Advanced tab, click **Performance settings**.
   c. In the Performance Options Data Execution Prevention tab, select the first option, **DEP for essential services only**.

      If you cannot change this option, click **Add**. Browse to the client program, for example IEXPLORE.EXE.
   d. If neither of these options is possible, try to disable DEP completely.
      i. Open a command prompt.
      ii. Run the following command: **bcdedit.exe /set {current} nx AlwaysOff**
      iii. Reboot the machine.
      iv. Verify that the settings took effect by running the following at the command line: **BCDEdit /enum**
      v. Verify that **nx** is **AlwaysOff**.
3. (Optional) Configure the recording options.

   The recording options affect the way a Vuser script is recorded and how it is generated after the recording. For details, see "Recording Options" on page 164.

   **Tip:**

   - **Web HTTP/HTML scripts:** To generate a Recording Summary report after recording is finished see "Enable the Recording Summary report" on the next page.
   - If the business process you want to record contains asynchronous push communication, select **Recording Options > HTTP > Advanced** > **Use streaming mode when recording with the LoadRunner Proxy**.
   - If you are not able to successfully record a script with VuGen, select **Recording Options > HTTP > Advanced Node** and check the **Use LR Proxy to record a local application** option. Then rerecord your business process.
4. Start the recording session.

   To start recording, click the **Record** button on the VuGen toolbar, make the relevant selections in the Start Recording dialog box, and click **Start Recording**. VuGen's floating toolbar appears, VuGen opens your application and begins recording your actions.

- For user interface details, see "Start Recording Dialog Box" on page 240.
- For details on the script sections into which you can record, see "Vuser Script Sections" on page 155.

5. Perform business processes on your application.

   Perform the desired business processes that you wish to record. The floating toolbar allows you to insert transactions, rendezvous points, and comments. You can also use the floating toolbar to specify into which section of the script to record. For user interface details, see "Floating Recording Toolbar" on page 244.

   Click the **Stop** button on the floating toolbar when you are finished recording.

> **Note:** To cancel the recording session, click the **Cancel Recording** button on the floating toolbar. When you cancel a recording, VuGen removes all of the code that was added to the script during the current recording session, thereby restoring the script to its status before the current recording session. For details on enabling/disabling the **Cancel Recording** button, see "Scripting Options Tab" on page 119.

## Recording Summary Report

When you record your application, unwanted information is often included in the script, such as Google statistics, advertisements, and third party information.

VuGen's Recording Summary report enables you to make decisions about which parts of the recording to include in your generated script, and which to filter out.

The Recording Summary provides general information about your recording session, as well as detailed information about hosts, headers, and content types. The report also enables you to manage traffic filtering.

> **Note:** This feature is available for Web HTTP/HTML scripts only.

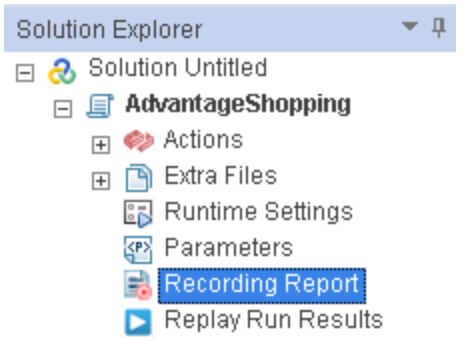
### Enable the Recording Summary report

To configure VuGen to automatically generate a Recording Summary report at the end of a recording session:

1. In VuGen, select **Tools > Options**.
2. On the **Scripting** tab, under **Recording**, select **Generate Recording Summary report**.
3. To open the report after it is generated, select **Display the Recording Summary report when recording is finished**.

## Access the Recording Summary report

Assuming **Generate Recording Summary report** is enabled (see "Enable the Recording Summary report" on the previous page above), when you finish recording your script the Recording Summary report appears in the script tree under the **Recording Report**.

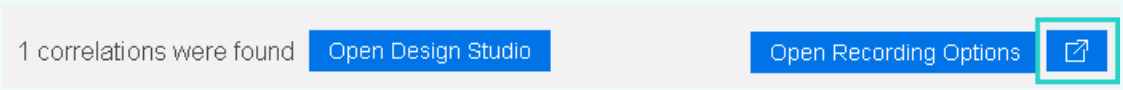



If your options are set to display the report when recording is finished, the report opens in the Script Editor pane.

## Recording Summary report overview

The Recording Summary report displays a dashboard which contains top-level details about your recording session.

**Tip:** Click the button in the upper-right corner to expand the report.




The report displays the following:

| Page | Description |
|---|---|
| **Dashboard** | Displays:<br>• General recording information:<br>  • Total size: Details about the recorded script including throughput, number of requests, and number of connections recorded.<br>  **Note:** When using WinINet mode, the number of connections is not available.<br>  • Date and duration of the recording<br>  • The client application or browser<br>• Number of correlations found during the recording session<br>• The number of headers found during the recording session<br>• Distribution of content types and response codes<br>• Host information, by domain |
| **Hosts** | Displays of the hosts that were recorded during the recording session, grouped by domain.<br>The following information is provided about each host:<br>• Host's IP address<br>• SSL version and SSL Cipher used for the connection<br>• HTTP version<br>• Number of requests made<br>• Total size of data received from the host during the recording session |
| **Headers** | Displays information about the HTTP headers created during the recording.<br>Headers that are included (selected) are added in a **web_add_header** function.<br>**Note:** This page is informational only, and cannot be edited. |
| **Content Types** | Displays information about the content types recorded:<br>• how many times they appeared during the recording session<br>• total size and download time of matching data received |

## Filter recorded script using the Recording Summary report

- To drill down into the report details:

| Detail | Click.... |
|---|---|
| Correlations | **Open Design Studio** button |
| Headers | **x Headers were recorded** or **More details** link |
| Hosts | **More details** link |
| Content Types | **More details** link |

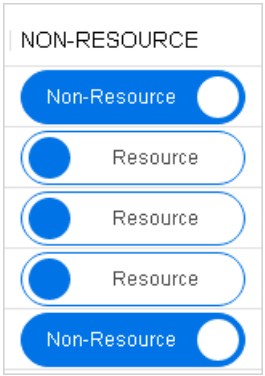
- Use the search bar on the **Hosts**, **Headers**, or **Content-Type** pages to find a particular detail.
- On the **Hosts**/**Headers** pages, select the **Included**/**Excluded** buttons to view a list of hosts/headers in your script. The **Included**/**Excluded** buttons display the number of hosts/headers included or excluded. When both buttons are selected (default), all of the hosts/headers are listed.

  On the Hosts page, you can select whole domains, or particular hosts within domains, to include/exclude in the script.

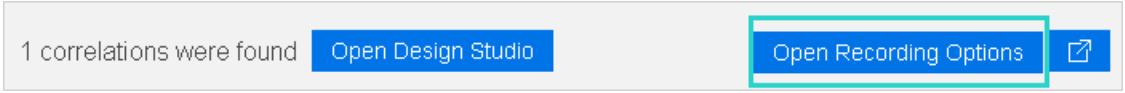
  > **Note:** The Headers page is informational only, and cannot be edited.

- On the **Content Types** page, select the **Non-Resource**/**Resource** buttons to view a list of non-resource and resource content types found in the script. The **Non-Resource**/**Resource** buttons display the number of content types that are non-resources or resources. When both buttons are selected (default), all of the content types found in the script are listed.

  Use the togglers next to each content type to define it as a non-resource or resource.




- When you filter the hosts or content types, the filter is added to the **Recording Options.** To manually edit the filter lists, click the **Open Recording Options** button at the top of the report.



For host and traffic filtering information, see "Network > Mapping and Filtering Recording Options" on page 204.

For Resources/Non-resources filtering information, see "Non-Resources Dialog Box" on page 199.

### Regenerate the script

When you make changes in the report, red asterisks appear next to the changed entities and the **Regenerate** button appears at the bottom of the screen. Click to regenerate the script.

The changes are applied to the script.

**Note:** Manual enhancements to a recorded action step are lost when regenerating the script. For more details, see "Regenerate a Vuser Script" on page 239.

## Create a Business Process Report

At the final stage of script creation, you can create a report that describes your business process.

### Types of information included in business process reports

You can edit or update the business process report template according to your requirements, as well as customize the contents of the report by indicating the type of information to include.

| Item | Supported types |
|---|---|
| Script export formats | • Microsoft Word<br>• Acrobat PDF<br>• HTML |
| Template types | • Pre-designed template<br>• Built-in VuGen template (Microsoft Word 2007 (.docx) format) |

| Item | Supported types |
|---|---|
| Protocols | • TruClient<br>• Citrix<br>• Oracle NCA<br>• Oracle - Web<br>• RDP<br>• SAP GUI<br>• SAP - Web<br>• Web - HTTP/HTML<br>• Web Services |

### Create a business process report

Select **Tools > Business Process Report** and fill in the required data. For user interface details, see "Business Process Report Dialog Box" on page 130.

### Configure additional report options

Click the **More** button to modify options such as the table of contents, snapshots, and the document template. For user interface details, see "Business Process Report Dialog Box" on page 130.

## Recording Options

The **Recording Options** sections describes the many different options that affect your Vuser script during the recording and generation stages of creating a script.

### Citrix > Configuration Recording Options

Enables you to set the window properties and encryption settings for the Citrix client during the recording session.

| To access | **Record > Recording Options > Citrix > Configuration** |
|---|---|
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Encryption Level** | The level of encryption for the ICA connection: **Basic**, **128 bit for login only**, **40 bit**, **56 bit**, **128 bit**, or **Use Server Default** to use the machine's default. |

| UI Element | Description |
|---|---|
| **Window Size** | The size of the client window.<br>**Default value:** 800 x 600. |

## Citrix > Code Generation Recording Options

Enables you to configure the way VuGen captures information during recording.

| To access | **Record > Recording Options > Citrix > Code Generation** |
|---|---|
| **Important information** | • This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223.<br>• Text synchronization steps that you add manually during the recording are not affected by the above settings—they appear in the script even if you disable the above options.<br>• The synchronization options also work for regenerating a script. For example, if you originally recorded a script with **Add text synchronization calls** disabled, you can regenerate after recording to include text synchronization. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Use Citrix Agent input in Code Generation** | Use the Citrix Agent input to generate a more descriptive script with additional synchronization functions.<br>**Default value:** enabled.<br>• **Automatically generate text synchronization calls.** Adds text synchronization **Sync on Text** steps before each mouse click.<br>**Default value:** disabled. |

## Citrix > Login Recording Options

Enables you to you set the connection and login information for the recording session.

| To access | **Record > Recording Options > Citrix > Login** |
|---|---|
| **Important information** | • The **Login** node is available only when creating a single protocol Citrix script. When creating a multi-protocol Citrix+Web script, the login information is retrieved from the Web interface.<br>For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223.<br>• If you do not provide login information, you are prompted for the information when the client locates the specified server. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Connection** | • **Network Protocol**. The preferred protocols are TCP/IP and TCP/IP+HTTP. Most Citrix Servers support TCP/IP, however some Citrix clients do not. Certain servers are configured by the administrators to allow only TCP/IP with specific HTTP headers. If you encounter a communication problem, select the TCP/IP+HTTP option.<br>• **Server.** The Citrix server name. To add a new server to the list, click **Add**, and enter the server name (and its port for TCP/IP + HTTP).<br>**Note:** Multiple servers apply only when you specify a Published Application. If you are connecting to the desktop without a specific application, then list only one server.<br>• **Published Application.** The name of the **Published Application** as it is recognized on Citrix server. The drop-down menu contains a list of the available applications. If you do not specify a published application, VuGen uses the server's desktop. If you added or renamed a published application, close the Recording options and reopen them to view the new list. Additionally, you can also enter the name of a published application manually if you know it exists (useful in cases where the drop-down list is inaccurate).<br>To change the name of the published application on the Citrix client, you must make the change on the Citrix Server machine. Select **Manage Console > Application** and create a new application or rename an existing one.<br>**Note:** If you do not specify a published application, Citrix load balancing will not work. To use load balancing when accessing the server's desktop, register the desktop as a published application on the server machine, and select this name from the Published Application drop-down list. |
| **Define connection parameters** | Allows you to manually define the logon and connection details. |
| **Logon Information** | Specify the **User Name**, **Password**, and **Domain** of the Citrix user. Optionally, you can also specify the **Client Name** by which the Citrix server identifies the client. |
| **Use ICA file for connection parameters** | Specify an ICA file with the connection configuration information for the application. For details, see ICA File Structure. |

## ICA File Structure

Citrix ICA client files are text files that contain configuration information for the applications accessed through the Citrix client.

> **Tip:** When you load an ICA file using the Recording Options, VuGen saves the file together with your script, eliminating the need to copy the ICA file to each load generator machine.

ICA Files must have an **.ica** extension and must conform to the following format:

```
[WFClient]
Version=
TcpBrowserAddress=

[ApplicationServers]
AppName1=

[AppName1]
Address=
InitialProgram=#
ClientAudio=
AudioBandwidthLimit=
Compress=
DesiredHRES=
DesiredVRES=
DesiredColor=
TransportDriver=
WinStationDriver=
Username=
Domain=
ClearPassword=
```

The following example shows a sample ICA file for using Microsoft Word on a remote machine through the Citrix client:

```
[WFClient]
Version=2
TcpBrowserAddress=235.119.93.56

[ApplicationServers]
Word=

[Word]
Address=Word
InitialProgram=#Word
ClientAudio=On
AudioBandwidthLimit=2
```

```
Compress=On
DesiredHRES=800
DesiredVRES=600
DesiredColor=2
TransportDriver=TCP/IP
WinStationDriver=ICA 3.0
Username=test
Domain=user_lab
ClearPassword=test
```

For more information, see the Citrix website www.citrix.com.

## Citrix > Recorder - Recording Options

Enables you to configure how window names are generated when the window title changes during recording and to configure whether to save snapshots of the screens.

| | |
|---|---|
| **To access** | **Record > Recording Options > Citrix > Recorder** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Save snapshots** | Saves a snapshot of the Citrix client window for each script step, when relevant. We recommend that you enable this option to provide you with a better understanding of the recorded actions. Saving snapshots, however, uses more disk space and slows down the recording session. |

| | |
|---|---|
| **Window name** | In some applications, the active window name changes while you are recording. If you try to replay the script as is, the Vuser uses the original window name and the replay may fail. You can specify a naming convention for the windows in which VuGen uses a common prefix or common suffix to identify the windows as follows:<br>• **Use new window name as is.** Set the window name as it appears in the window title. (default)<br>• **Use common prefix for new window names.** Use the common string from the beginning of the window titles as a window name.<br>• **Use common suffix for new window names.** Use the common string from the end of the window titles as a name.<br>Alternatively, you can modify the window names in the actual script after recording. In the Script view, locate the window name, and replace the beginning or end of the window name with the "*" wildcard notation.<br> **Example:** ctrx_sync_on_window ("My Application*", ACTIVATE, ...CTRX_LAST); |

## Correlations > Configuration Recording Options

This recording option pane enables you to configure settings for the Correlation tab.

| | |
|---|---|
| **To access** | **VuGen > Recording Options > Correlations > Configuration** |
| **Important information** | • "Correlating Values" on page 254<br>• "Correlations > Rules Recording Options" on page 171<br>• This recording option is only available for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |
| **Relevant tasks** | "Correlate Scripts Using Design Studio" on page 268 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Scan for correlations applying:** | |
| **Rules Scan** | Apply correlation rules when performing correlation scan.<br>For details, see "Automatic Correlation" on page 256 and "Correlations > Rules Recording Options" on page 171. |
| **Automatically correlate values found** | Design Studio will automatically correlate dynamic values found using the rules scan. |

| UI Element | Description |
|---|---|
| **Record Scan** | Scan for correlations with the record based engine. For details, see "Automatic Correlation" on page 256. |
| **Replay Scan** | Scan for correlations with the replay based engine. For details, see "Automatic Correlation" on page 256. |
| **Record and Replay scan configuration** | |
| **API used for correlations** | Select the API function to be used for correlation:<br>**Boundary based**: web_reg_save_param_ex<br>**Regular Expression**: web_reg_save_param_regexp<br>**Note:** If you change the API function, the changes will only take effect after a new scan. |
| **Excluded strings** | Enables you to enter strings that should be ignored by the record and replay scan.<br>For details, see "Exclude Strings or Content Types from the Correlation Scan" on page 258. |
| **Excluded content types** | Enables you to enter content types that should be ignored by the record and replay scan. For details, see"Exclude Strings or Content Types from the Correlation Scan" on page 258. |
| **Ignore values shorter than [ ]** | Enables you to define how short a dynamic value can be before it is ignored by the record or replay scan.<br>Default length is 4 characters. |
| **Ignore values longer than [ ]** | Enables you to define how long a dynamic value can be before it is ignored by the record or replay scan.<br>Default length is 400 characters. |
| **Warm me if the dynamic string size is greater than 10 KB** | Issues a warning if you try to correlate a string whose size is 10 KB or larger. |
| **Ignore case when searching for correlation values** | Disable case sensitivity during correlation scan. |
| **Record scan configuration** | |

| UI Element | Description |
|---|---|
| **Heuristic level** | Enables you to set the filter level that controls the amount of correlation results that are returned. The higher the filter level, the shorter the scan will take to run.<br>**High.** Design Studio performs a detailed scan returning a highly refined result set.<br>**Medium.** Design Studio performs a less detailed scan returning more results. This is the default setting.<br>**Low.** Design Studio performs a more detailed scan returning the most results. This setting may produce many unwanted results. |
| **Replay scan configuration** | |
| **Scan for differences between snapshots using** | Select a comparison method:<br>• **HTML Comparison.** Display the differences in HTML code only.<br>• **Text Comparison.** Display all text, HTML, and binary differences. |

## Correlations > Rules Recording Options

This dialog box enables you to manage correlation rules that automatically correlate dynamic values during code generation. You can new and delete applications. You can also as add, delete, import, export and test correlation rules.

| | |
|---|---|
| **To access** | Do one of the following:<br>• **Record > Recording Options > Correlations > Rules**<br>• **Design Studio > Options > Correlations tab > Rules** |
| **Important information** | • "Correlation Overview" on page 251<br>• "Correlating Values" on page 254 |
| **Relevant tasks** | "Correlate Scripts Using Design Studio" on page 268 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Application List >** | A list of applications and their rules.<br>• Select the check box adjacent to the application to activate it during recording.<br>• Expand the application tree to select the check box adjacent to the rules to activate them during recording. |

| UI Element | Description |
|---|---|
| New Application | Add a new application to **<Application List>**. |
| New Rule | Enter a new rule for the selected application in Correlation Rules.<br>For details, see "New Rule Pane" below. |
| Delete | Delete the selected application or rule from the list. |
| Import... | Import a file containing correlation rule definitions. |
| Export... | Export a file containing a correlation rule definition. |
| Test... | Test a correlation rule.<br>For details, see " Token Substitution Testpad Dialog Box" on page 175. |

## New Rule Pane

Enables you to define a new custom rule.

| | |
|---|---|
| **To access** | **Record > Recording Options > Correlation > Rules > New Rule** |
| **Important information** | This pane is available only for specific protocols. For a complete list of protocols and their associated nodes, see the "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Advanced... | Opens the "Advanced Correlation Properties Dialog Box" on page 174. |

| | |
|---|---|
| **Action** | Specify the type of action for the rule from the following options:<br>• **Search for Parameters in all of the Body Text.** Searches the entire body—not just links, form actions or cookies. It searches the text for a match using the borders that you specify.<br>• **Search for parameters inlinks and form actions.** Searches within links and forms' actions for the text to parameterize. This method is for application servers where you know the context rules. You define a left boundary, a right boundary, an alternate right boundary, and an instance of the left boundary within the current link.<br>• **Search for Parameters from cookie headers.** Similar to the previous rule, except that the value is extracted from cookie text (exactly as it appears in the recording log) instead of from a link or form action.<br>• **Parameterize form field value.** Saves the named form field value to a parameter. It creates a parameter and places it in the script before the form's action step. For this option, you need to specify the field name.<br>• **Text to enter a web_reg_add_cookie function by** method inserts a **web_reg_add_cookie** function if it detects a certain string in the buffer. It only adds the function for those cookies with the specified prefix. For this option, you need to specify the search text and the cookie prefix. |

| | |
|---|---|
| **Scan Type** | The scan type. Select a scan type, and fill in the relevant details:<br>**Regular Expression:**<br>• **RegExp String.** A regular expression to which this rule will apply.<br>**Boundary Based:**<br>• **Left boundary.** The left-most boundary where the rule will apply.<br>• **Right boundary.** The right-most boundary where the rule will apply. Use the drop-down menu to define this boundary as either the end of a string, a newline character, or a user-defined text.<br>• **Match Case.** Matches the case when looking for boundaries<br>• **Use '#' for any digit.** Replaces all digits with a hash sign. The hash signs serve as wildcard, allowing you to find text strings with any digit.<br>**Example:** If you enable this option and specify **ABC###** as the left boundary, **ABC193** and **ABC284** will be valid matches.<br>**Attribute Based:** Available for **Search for Parameters in all of the Body Text** actions only.<br>• **TagName.** The name of HTML element.<br>• **Extract.** The name of HTML attribute whose value will be saved to the parameter.<br>• **Name/ID**. The value of the HTML element's "name"/"Id" attributes.<br>• **Class.** The value of the HTML element's "class" attribute.<br>• **Type.** The value of the HTML element's "type" attribute. For example, for the INPUT element, Type can be "hidden".<br>**XPath query:**<br>• **XPath string.** An XPath expression to which this rule will apply. This element only applies to a **XPath query** scan type. When VuGen detects a value that matches this expression, it creates a **web_reg_save_param_xpath** function.<br>**JSON query:**<br>• **JSONPath String.** A JSON query expression to which this rule will apply. This element only applies to a **JSON query** scan type. When VuGen detects a value that matches this expression, it creates a **web_reg_save_param_json** function. For details about creating a JSON query, see Internet resources on JSONPath expressions. |
| **Parameter prefix** | Uses a prefix in all automatically generated parameters based on this rule. Prefixes prevent you from overwriting existing user parameters. In addition, prefixes allow you to recognize the parameter in your script. For example, in Siebel Web, one of the built-in rules searches for the **Siebel_row_id** prefix. |

## Advanced Correlation Properties Dialog Box

Enables you to set the advanced options for correlation rules.

| | |
|---|---|
| **To access** | **Record > Recording Options > Correlation > Rules > New Rule > Advanced** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see the "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Always create new parameter** | Creates a new parameter for this rule even if the value replaced by the parameter has not changed from the previous instance. |
| **Replace with parameter only for exact matches** | Replaces a value with a parameter only when the text exactly matches the found value. |
| **Reverse search** | Performs a backwards search. |
| **Left boundary instance** | The number of occurrences of the left boundary in order for it to be considered a match. |
| **Offset** | The offset of the string within the found value. |
| **Length** | The length of the string, starting with the offset, to save to the parameter. If this is not specified, the parameter continues until the end of the found value. |
| **Alternate right boundary** | Alternative criteria for the right boundary, if the previously specified boundary is not found. Select one of the following options: **User-defined Text, Newline Character, End Of Page.** |

### Token Substitution Testpad Dialog Box

Enables you to test correlation rules before applying them.

| | |
|---|---|
| **To access** | **Record > Recording Options > Correlations > Rules > Test** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Runs the test. |
| **Applied rules** | A list of the rules and their values that were applied during the test. |
| **Source string for substitution** | Enter the source string for substitution. |
| **Substitution Result** | The results of the test. |

## Database > Database Recording Options

Enables you to set the recording options for database protocols.

| | |
|---|---|
| **To access** | **Record > Recording Options > Database > Database** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Advanced... | Opens the "Database > Advanced Recording Options Dialog Box" below. |
| **Automatic transactions** | If enabled, every **lrd_exec** and **lrd_fetch** function is marked as a transaction with **lr_start_transaction** and **lr_end_transaction** functions.<br>**Default value:** Disabled. |
| **Script options** | Generates comments into recorded scripts, describing the **lrd_stmt** option values. In addition, you can specify the maximum length of a line in the script.<br>**Default value:** 80 characters. |
| **Think time** | VuGen automatically records the operator's think time. You can set a threshold level, below which the recorded think time will be ignored. If the recorded think time exceeds the threshold level, VuGen places an **lr_think_time** statement before LRD functions. If the recorded think time is below the threshold level, an **lr_think_time** statement is not generated.<br>**Default value:** five seconds. |

## Database > Advanced Recording Options Dialog Box

Enables you set the advanced recording options for database protocols.

| | |
|---|---|
| **To access** | **Record > Recording Options > Database > Database > Advanced** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Recording log options** | You can set the detail level for the trace and ASCII log files.<br>**Trace file detail level:**<br>• **Off**<br>• **Error Trace.** Only logs error messages.<br>• **Brief Trace.** Logs errors and lists the functions generated during recording.<br>• **Full Trace.** Logs all messages, notifications, and warnings.<br>**ASCII file detail level:** Instructs VuGen to generate ASCII type logs of the recording session.<br>• **Off**<br>• **Brief detail.** Logs all of the functions.<br>• **Full detail.** Logs all of the generated functions and messages in ASCII code. |
| **Code generation buffer size** | Specify in kilobytes the maximum size of the code generation buffer.<br>**Default value:** 128 kilobytes. |

## Data Format Extension > Chain Configuration Recording Options

Enables you to add, delete, and modify chains, and to manage the DFEs that are included in the chains.

| | |
|---|---|
| **To access** | **Record > Recording Options > Data Format Extension > Chain Configuration** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |
| **See also** | • "Data Format Extensions (DFEs) - Overview" on page 741<br>• "Data Format Extension List" on page 750<br>• "Implementing GWT-DFE Support" on page 759<br>• "Add Prefix/Postfix to Chain Dialog Box" on the next page |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Chains pane** | Displays a list of the DFE chains that are defined for the script. |
| | **Add Chain**. Enables you to add a new chain. |

| | |
|---|---|
| | **Edit Chain Name**. Enables you to modify the name of the selected chain. |
| | **Delete Chain**. Deletes the selected chain. |
| **Chain: <chain name> pane** | |
| | **Add DFE**. Enables you to add a DFE to the selected chain in the **Chains** pane. For more information on Data Format Extensions, see "Data Format Extension List" on page 750. |
| | **Edit DFE**.For a *Prefix Postfix Extension*, edit the prefix and postfix to cut. For a *GWT Extension*, specify the classpath. |
| | **Add Custom Path**. Lets you add a custom classpath for GWT. You can use this to specify Linux paths. For example, /tmp/stockwatcher.war. |
| | **Delete DFE**. Deletes the selected DFE from the chain. |
| | **Move Up/Down**. Moves the selected Data Format Extension up or down in the chain. Extensions are run in the order in which they appear in the extensions list. |
| **Name** | The display name of the Data Format Extension. |
| **Tag** | The unique ID of the extension. |
| **Provider** | The creator of the Data Format Extension. |
| **Continue Processing** | Determines how the chain behaves after the DFE is applied:<br>• **True:**<br>The data is passed on to the next DFE in the chain, whether or not the data was converted.<br>• **False:**<br>If the DFE converted the data that it received, the chain is terminated - no further DFEs are applied to the data.<br>If the DFE did not convert the data that it received, the data is passed on to the next DFE in the chain.<br>**Note:** If the chain contains only a single DFE, the **Continue Processing** setting is not significant. |

## Add Prefix/Postfix to Chain Dialog Box

This dialog enables you to add or edit a prefix/postfix extension to the selected chain.

| | |
|---|---|
| **To access** | 1. Go to **Record > Recording Options > Data Format Extension > Chain Configuration** node.<br>2. In the **Chain: <Chain name>** area, click the button.<br>3. Select **Prefix Postfix Extension**and click **OK**. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Case sensitive** | Sets the extension to cut from the defined prefix and postfix of the string only if the letter cases match. |
| **Display name** | The name of Prefix/Postfix Extension. |
| **Postfix to cut** | The section you want to cut, from the end of the string. |
| **Prefix to cut** | The section you want to cut, from the beginning of the string. |
| **Tag name** | The unique ID of the Prefix/Postfix Extension. |

## See also:

- Data Format Extension > Chain Configuration Node
- Data Format Extensions
- Data Format Extension List

### Add Data Format Extension

This dialog box enables you to select the data format extension type.

| | |
|---|---|
| **To access** | **VuGen > Recording Options > Data Format Extension > Chain Configuration >** |
| **Important information** | • "Data Format Extension > Chain Configuration Recording Options" on page 177<br>• "Add Prefix/Postfix to Chain Dialog Box" on the previous page |
| **Relevant tasks** | "Apply DFE Chains to Sections of the HTTP Message" on page 748 |
| **Data Format Extension** | **Description** |

| | |
|---|---|
| Base64 Extension | Decodes strings that are encoded with a Base64 encoder. |
| GWT Extension | Transforms GWT data to XML format. |
| URL Encoding Extension | Decodes strings that are encoded with URL encoding format. |
| JSON to XML Extension | Transforms JSON data to XML format. |
| XML Extension | Receives data and checks to see if it conforms with XML syntax. This check allows VuGen to perform correlations based on XPath and to display snapshot data in an XML viewer. |
| Prefix Postfix Extension | Enables you to cut data from the beginning and/or end of a string which you do not want decoded. You can add and customize as many prefix/postfix extensions as required. Each postfix/prefix extension created should have a unique display name and tag name. |
| Binary to XML Extension | Transforms Microsoft WCF binary XML into XML format. |
| Remedy to XML Extension | Transforms Remedy request data into XML format.<br>This extension does not transform Remedy response data, which is JavaScript code. |
| XSS Extension | Enables you to test sites that use Cross Site Scripting (XSS) defense code. |

## Data Format Extension > Code Generation Recording Options

Enables Data Format Extensions during code generation, and enables you to define chains for each section of the HTTP message.

| | |
|---|---|
| **To access** | **Record > Recording Options > Data Format Extension > Code Generation** |
| **Important information** | This node is available for specific protocols only. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |
| **See also** | • "Data Format Extensions (DFEs) - Overview" on page 741<br>• "Data Format Extension List" on page 750 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Enable data format extension** | Enables you to select chains for each message section of the HTTP message.<br>By default, this option is not selected. |
| **Configuration** | |
| **Format** | • **Code and snapshots**. (default) Enables Data Format Extensions on the code and snapshot data.<br>• **Snapshots**. Enables Data Format Extensions on snapshot data, but does not format the data in the script itself. |
| **Verify formatted data** | Checks the results of the formatted data by converting it back to the original state and verifying that it matches the original data.<br>**Note:** Available for Base64 extension only. |
| **Chain Assignment** | |
| | Imports the Data Format Extensions from a file. |
| | Exports the Data Format Extensions to a file. |
| **<Message sections list>** | Displays a list of the following sections of the HTTP message included in the script:<br>• Body<br>• Headers<br>• Cookies<br>• Query String<br>When you select a message section from the list, the title of the section chains pane (described below) reflects your selection and the pane displays the list of chains for that section. |
| **<Section Chains>** | |

| | |
|---|---|
| | **Add Chain**. Adds chain to selected message section.<br><br>**Note:**<br>• Enabled for Headers and Cookies sections only. Enables you to add additional chains to the selected message section.<br>• For VuGen to correctly match the chain to the Headers or Cookies section, the name in the **Name** column must match the name of the Headers or Cookies section. |
| | **Delete Chain**. Removes chain from corresponding message section.<br><br>**Note:** You cannot delete the default options from any of the message sections. |
| | **Reset**. Clears the selected chain in the Chain column. |

## Flex > RTMP Recording Options

This node enables you to include the flex_rtmp_recive_stream step in Flex RTMP scripts.

| | |
|---|---|
| **To access** | **Record > Recording Options > Flex > RTMP** |
| **Important Information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Generate single step for RTMP/T stream handling** | Generates a single step, **flex_rtmp_receive_stream**, when recording a stream. This step does not replay certain actions, such as pause and seek. If your script requires these actions, clear the check box to record all receive and send steps. |

## Flex > Configuration Recording Options

Enables you to set an external JVM (Java Virtual Machine) path.

| | |
|---|---|
| **To access** | **Record > Recording Options> Flex > Configuration** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Use External JVM** | Enables you to use an external JVM. If you choose this option, you must specify the path:<br><br>**External JVM Path** The full path of the external JVM.<br><br>VuGen must be restarted for the changes to be applied. |
| **UseGraniteDS configuration** | Defines the server side Data Service configuration.<br><br>If you select this option, do not select **Use Flex LCDS/BlazeDS jars** to serialize the messages. Ensure that the granit-config.xml file matches the one deployed on the server. |
| **Maximum Formatted Request/Response size to print** | Enables you to specify the maximum character length of a request or response body to be captured in the log files. The option only affects the **flex_amf_call** and **flex_remoting_call** steps.<br><br>For example, if you specify a value of 1048576 characters (1MB), only responses or requests with a length less than a megabyte will be printed on the Code Generation or Replay log. |

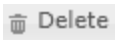
## Flex > Externalizable Objects Recording Options

This dialog box enables you to configure how to handle externalizable objects in Flex scripts.

| | |
|---|---|
| **To access** | **Record > Recording Options > Flex > Externalizable Objects** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |
| **Relevant tasks** | "Serialize Flex Scripts" on page 545 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Do not serialize externalizable objects** | Generate script using default settings. |

| | |
|---|---|
| **Serialize objects using** | Select the appropriate option:<br>• Select **LoadRunner AMF serializer** if you are not using the Adobe LiveCycle Data Services or Adobe BlazeDS server.<br>• Select **Custom Java classes** and select one or both of the available options:<br>  • Select **Use Flex LCDS/BlazeDS jars** if you are using Flex LCDS or BlazeDS jars to serialize the messages. If you selected **UseGraniteDS configuration** in the **Configuration** node, do not select **Use Flex LCDS/BlazeDS jars**.<br>  • Select **Use additional jars** to add additional jars to serialize the messages. You must copy the jar files from the server and specify their location in the **Classpath Entries** list described below. Copy only those jars that contain the class that is externalizable. Ensure that the files exist in the same location on all load generator computers. If you add jars with the same names as the Flex LCDS or Blaze DS jars chosen by selecting the first check box, these files will be overwritten. |
| **Classpath Entries List** | |
| | **Down Arrow.** Moves a classpath entry down the list. |
| | **Up Arrow.** Moves a classpath entry up the list. |
| | **Add Classpath.** Adds a new line to the classpath list. |
| | **Add Classpath Folder.** Adds all files from the folder to the classpath list. |
|  | **Delete.** Permanently removes a classpath. |

## See also:

- Flex Overview
- Externalizable Objects in Flex Scripts

## FTP > Configuration Recording Options

Enables you to indicate additional parameters to use when recording FTP applications.

| | |
|---|---|
| **To access** | **Record > Recording Options > FTP > Configuration** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Welcome messages** | In order for VuGen to record FTP traffic, it must detect an FTP welcome message. The default welcome messages contain these strings: **FTP**, **ncFTP**, and **proFTPDA**.<br><br>This setting lets you specify a custom list of welcome messages issued by an FTP server when you connect. The recording of the FTP traffic only begins after the welcome message is detected.<br><br>You can specify entire words or word fragments and all types of characters. Use the pipe "\|" symbol as a separator between the strings. |

## General > Code Generation Recording Options

This pane of the Recording Options dialog box enables you to define what tasks VuGen performs automatically after generating a Vuser script.

| To access | **Record > Recording Options > General > Code Generation** |
|---|---|
| **Relevant tasks** | "Create an Asynchronous Vuser Script" on page 397<br><br>"Correlate Scripts Using Design Studio" on page 268 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Correlations Scan | Instructs VuGen to analyze the Vuser script to locate dynamic values that may need to be correlated. This scan is performed after a new script is recorded and after an existing script is regenerated. |
| Async Scan | Instructs VuGen to analyze the Vuser script to locate asynchronous communication. This scan is performed after a new script is generated and after an existing script is regenerated. |
| Async Options... | Opens the "Asynchronous Options Dialog Box" on page 423.<br><br>**Tip:** We recommend that you use a 100% display size. |

## General > Protocol Recording Options

Enables you to set the script generation preferences by setting the scripting language and options.

| To access | **Record > Recording Options > General > Protocols** |
|---|---|

| | |
|---|---|
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Active Protocols List** | A list of the protocols which comprise your multiple protocol script. VuGen lets you modify the protocol list for which to generate code during the recording session. Select the check boxes adjacent to the protocols you want to record in the next recording session. Clear the check boxes adjacent to the protocols you do not want to record in the next recording session. |

## General > Recording - Recording Options

Enables you to specify what information to record and which functions to use when generating a Vuser script, by selecting a recording level.

| | |
|---|---|
| **To access** | **Record > Recording Options > General > Recording** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Opens the "Advanced HTML Dialog Box" on the next page. |
| | Opens the "Advanced URL Dialog Box" below. |
| **HTML-based script** | This is the default recording level for Web - HTTP/HTML Vusers. It instructs VuGen to record HTML actions in the context of the current Web page. It does not record all resources during the recording session, but downloads them during replay. |
| **URL-based script** | Record all requests and resources from the server. It automatically records every HTTP resource as URL steps (**web_url** statements), or in the case of forms, as **web_submit_data**. It does not generate the **web_link**, **web_image**, and **web_submit_form** functions, nor does it record frames. We recommend this option for non-browser applications. |

### Advanced URL Dialog Box

Enables you to set the advanced options for scripts using the URL recording mode.

| To access | **Record > Recording Options > General > Recording > URL Advanced** |
|---|---|
| **Important information** | This dialog box is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Use Defaults | Restores the default settings of this dialog box. |
| **Create concurrent groups for resources after their source HTML page** | Records the resources in a concurrent group (enclosed by **web_concurrent_start** and **web_concurrent_end** statements) after the URL. Resources include files such as images and **js** files. If you disable this option, the resources are listed as separate **web_url** steps, but not marked as a concurrent group. |
| **Enable EUC-Encoded Web Pages** | (For Japanese windows only) Instructs VuGen to use EUC encoding. For more information, see "EUC-Encoding (Japanese Windows only)" on page 228. |
| **Use web_custom_request only** | Records all HTTP requests as custom requests. VuGen generates a **web_custom_request** function for all requests, regardless of their content. Recommended for non-browser applications. |

**Note:** The functions included within the concurrent group are not executed immediately. Instead, they are registered for concurrent execution. When the concurrent group is closed, all of the functions registered as concurrent are executed in parallel. The maximum number of items submitted in parallel depends on the browser being emulated. For example with Internet Explorer 9, up to six resources are downloaded simultaneously.

## Advanced HTML Dialog Box

Enables you to set the advanced options for HTTP-based scripts.

| To access | **Record > Recording Options > General > Recording > HTML Advanced** |
|---|---|
| **Important information** | This dialog box is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| Use Defaults | Restores the default settings of this dialog box. |
| **Non-HTML generated elements** | Many Web pages contain non-HTML elements, such as applets, XML, ActiveX elements, or JavaScript. These non-HTML elements usually contain or retrieve their own resources. Using the following options, you can control how VuGen records non HTML-generated elements.<br>• **Record within the current script step.** Does not generate a new function for each of the non HTML-generated resources. It lists all resources as arguments of the relevant functions, such as **web_url**, **web_link**, and **web_submit_data**. The resources, arguments of the Web functions, are indicated by the **EXTRARES** flag.<br>• **Record in separate steps and use concurrent groups**. Creates a new function for each one of the non HTML-generated resources and does not include them as items in the page's functions (such as **web_url** and **web_link**). All of the **web_url** functions generated for a resource are placed in a concurrent group (surrounded by **web_concurrent_start** and **web_concurrent_end**).<br>• **Do not record.** Does not record any non-HTML generated resources. |
| **Script type** | • **A script describing user actions.** Generates functions that correspond directly to the action taken. It creates URL (**web_url**), link (**web_link**), image (**web_image**), and form submission (**web_submit_form**) functions. The resulting script is very intuitive and resembles a context sensitive recording.<br>• **A script containing explicit URL's only.** Records all links, images and URLs as **web_url** statements, or in the case of forms, as web_submit_data. It does not generate the **web_link**, **web_image**, and **web_submit_form** functions. The resulting script is less intuitive. This mode is useful for instances where many links within your site have the same link text. If you record the site using the first option, it records an ordinal (instance) for the link, but if you record using the second option, each link is listed by its URL. This facilitates parameterization and correlation for that step. |

## General > Script Recording Options

Enables you to set the script generation preferences by setting the scripting language and options.

| | |
|---|---|
| **To access** | **Record > Recording Options > General > Script** |
| **Important information** | This node is available for specific protocols only. In addition, the list of options differs between protocols.<br>For a complete list of protocols and their associated nodes, see the "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Scripting Language** | Select the language in which to generate the Vuser script, C or JavaScript. The default is C.<br><br>**Note:** This option is available for Web - HTTP/HTML Vuser scripts only. |
| **Add a comment for each action** | Insert informative logging messages before each message invocation (non-C only).<br><br>**Default value:** enabled. |
| **Close all AUT processes when recording stops** | Automatically closes all of the AUT's (Application Under Test) processes when VuGen stops recording.<br><br>**Default value:** disabled. |
| **Correlate arrays** | Tracks and correlates arrays of all data types, such as string, structures, numbers, and so on.<br><br>**Default value:** enabled. |
| **Correlate large numbers** | Correlates long data types such as integers, long integers, 64-bit characters, float, and double.<br><br>**Default value:** disabled. |
| **Correlate simple strings** | Correlates simple, non-array strings and phrases.<br><br>**Default value:** disabled. |
| **Correlate small numbers** | Correlates short data types such as bytes, characters, and short integers.<br><br>**Default value:** disabled. |
| **Correlate structures** | Tracks and correlates complex structures.<br><br>**Default value:** enabled. |
| **Declare primitives as locals** | Declares primitive value variables as local variables rather than class variables (C, C#, and .NET only).<br><br>**Default value:** enabled. |
| **Explicit variant declaration** | Declares variant types explicitly in order to handle ByRef variants (Visual Basic for Applications only).<br><br>**Default value:** enabled. |

| UI Element | Description |
|---|---|
| **Generate fixed think time after end transaction** | Adds a fixed think time, in seconds, after the end of each transaction. When you enable this option, you can specify a value for the think time.<br>**Default value:** disabled, 3 seconds when enabled. |
| **Generate recorded events log** | Generates a log of all events that took place during recording.<br>**Default value:** disabled. |
| **Generate think time greater than threshold** | Uses a threshold value for think time. If the recorded think time is less than the threshold, VuGen does not generate a think time statement. You also specify the threshold value.<br>The default values is 3—if the think time is less than 3 seconds, VuGen does not generate think time statements. If you disable this option, VuGen will not generate any think times.<br>**Default value:** enabled, 3 seconds. |
| **Insert output parameters values** | Inserts output parameter values after each call (C, C#, and .NET only).<br>**Default value:** disabled. |
| **Insert post-invocation info** | Insert informative logging messages after each message invocation (non-C only).<br>**Default value:** enabled. |
| **Maximum number of lines in action file** | Create a new file if the number of lines in the action exceeds the specified threshold. The default threshold is 60000 lines (C, C#, and .NET only).<br>**Default value:** disabled. |
| **Replace long strings with parameter** | Save strings exceeding the maximum length to a parameter. This option has an initial maximum length of 100 characters. The parameters and the complete strings are stored in the **lr_strings.h** file in the script's folder in the following format:<br>**const char <paramName_uniqueID> ="string".**<br>This option allows you to have a more readable script. It does not effect the performance of the script.<br>**Default value:** enabled. |
| **Reuse variables for primitive return values** | Reuse the same variables for primitives received from method calls. This overrides the **Declare primitives as locals setting**.<br>**Default value:** enabled. |

| UI Element | Description |
|---|---|
| **Track processes created as COM local servers** | Track the activity of the recorded application if one of its sub-processes was created as a COM local server (C and COM only).<br>**Default value:** enabled. |
| **Use full type names** | Use the full type name when declaring a new variable (C# and .NET only).<br>**Default value:** disabled. |
| **Use helpers for arrays** | Use helper functions to extract components in variant arrays (Java and VB Scripting only).<br>**Default value:** disabled. |
| **Use helpers for objects** | Use helper functions to extract object references from variants when passed as function arguments (Java and VB Scripting only).<br>**Default value:** disabled. |
| **Use protected application recording** | Use this option if VuGen is unable to record your application. Your application may block access to VuGen, and recording with this option selected may enable access.<br>**Default value:** disabled. |
| **Warn me if the application being recorded encounters an error** | Selecting this option enables VuGen to prompt you to cancel the recording if the recorded application crashes or if no events are recorded for 3 minutes. If you choose to cancel the recording, no script is generated. In proxy mode, if the recorded application crashes, there is no error message. Instead, a no-events error is displayed after 3 minutes.<br>**Default value:** enabled.<br>**Note:** This option is available for Web - HTTP/HTML Vuser scripts only. |

## GUI Properties > Web Event Configuration Recording Options

Enables you to set the level of detail recorded in a script (web event recording).

| | |
|---|---|
| **To access** | **Record > Recording Options > GUI Properties > Web Event Configuration** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Basic Event Configuration Level** | • Always records click events on standard Web objects such as images, buttons, and radio buttons.<br>• Always records the submit event within forms.<br>• Records click events on other objects with a handler or behavior connected.<br>• Records the mouseover event on images and image maps only if the event following the mouseover is performed on the same object. |
| **Custom Settings** | Opens the "Custom Web Event Recording Configuration Dialog Box" below, where you can customize the event recording configuration. |
| **High Event Configuration Level** | In addition to the objects recorded in the **Medium** level, it records mouseover, mousedown, and double-click events on objects with handlers or behaviors attached. |
| **Medium Event Configuration Level** | In addition to the objects recorded in the **Basic** level, it records click events on the <DIV>, <SPAN>, and <TD> HTML tag objects. |

### Custom Web Event Recording Configuration Dialog Box

Enables you to customize the level of web event recording.

| | |
|---|---|
| **To access** | **Record > Recording Options > GUI Properties > Web Event Configuration > Custom Settings** |
| **Important information** | This dialog box is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Object List>** | A list of the web objects. Each web object can be customized according to the other settings in this dialog box. |
| **<Object Menu>** | • **Add.** Adds a new HTML tag object to the object list. Type in the name of the tag.<br>• **Delete.** Deletes an object from the object list. |
| **Event Menu** | • **Add.** Adds an event to the Event Name column of this object.<br>• **Delete.** Deletes an event from the Event Name column of this object. |
| **Event Name** | A list of events associated with the object. |

| UI Element | Description |
|---|---|
| **File Menu** | • **Load Configuration.** Loads a previously created custom configuration.<br>• **Save Configuration As.** Saves the current configuration. |
| **Listen** | The criteria which determines when VuGen listens for an event.<br>• **Always**. Always listen to the event.<br>• **If Handler**. Listens to the event if a handler is attached to it. A handler is code in a Web page, typically a function or routine written in a scripting language, that receives control when the corresponding event occurs.<br>• **If Behavior**. Listens to the event if a DHTML behavior is attached to it. A DHTML behavior encapsulates specific functionality or behavior on a page. When applied to a standard HTML element on a page, a behavior enhances that element's default behavior.<br>• **If Handler or Behavior**. Listens to the event if either a handler or a behavior is attached to it.<br>• **Never**. Never listens to the event.<br>For more information, see "Tips for Working with Event Listening and Recording" on page 231. |
| **Record** | The criteria which determines when VuGen records an event.<br>• **Enabled.** Records the event each time it occurs on the object as long as VuGen listens to the event on the selected object, or on another object to which the event bubbles. Bubbling is the process whereby, when an event occurs on a child object, the event can travel up the chain of hierarchy within the HTML code until it encounters an event handler to process the event.<br>• **Disabled.** Does not record the specified event and ignores event bubbling where applicable.<br>• **Enabled on next event.** Same as **Enabled**, except that it records the event only if the subsequent event occurs on the same object. For example, suppose a mouseover behavior modifies an image link. You may not want to record the mouseover event each time you happen to move the mouse over this image. Because only the image that is displayed after the mouseover event enables the link event, however, it is essential that the mouseover event is recorded before a click event on the same object.<br>For more information, see "Tips for Working with Event Listening and Recording" on page 231. |
| **Reset Settings** | Resets the custom settings to the settings of your choice: **basic**, **medium**, or **high**. |

## GUI Properties > Advanced Recording Options

Enables you to set advanced recording options for Click & Script Vusers.

| | |
|---|---|
| **To access** | **Record > Recording Options > GUI Properties > Advanced** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

### Recording Settings Properties

| UI Element | Description |
|---|---|
| **Record rendering-related property values** | Records the values of the rendering-related properties of DOM objects (for example, **offsetTop**), so that they can be used during replay. Note that this may significantly decrease the replay speed.<br>**Default value:** disabled. |
| **Record 'click' by mouse events** | Records mouse clicks by capturing mouse events instead of capturing the click() method. Enable when the recorded application uses the DOM click() method, to prevent the generation of multiple functions for the same user action.<br>**Default value:** enabled. |
| **Record socket level data** | Enables the recording of socket level data. If you disable this option you will need to manually add the starting URL before recording. In addition, you will be unable to regenerate the script on an HTML level.<br>**Default value:** enabled. |
| **Generate snapshots for Ajax steps** | Enables generation of snapshots for Ajax steps. Enabling this option can result in errors during recording.<br>**Default value:** disabled. |

### Code Generation Settings Properties

| UI Element | Description |
|---|---|
| **Enable generation of out-of-context steps** | Creates a URL-based script for ActiveX controls and Java applets, so that they will be replayed. Since these functions are not part of the native recording, they are referred to as out-of-context recording.<br>**Default value:** disabled. |
| **Enable automatic browser title verification** | Enables automatic browser title verification.<br>**Default value:** disabled. |

**, continued**

| | |
|---|---|
| **Perform a title verification** | • **for each navigation.** Performs a title verification only after a navigation. When a user performs several operations on the same page, such as filling out a multi-field form, the title remains the same and verification is not required.<br>• **for each step.** Performs a title verification for each step to make sure that no step modified the browser title. A modified browser title may cause the script to fail.<br>• **using the URL if the title is missing.** For browser windows without a title, perform a title verification for each step using its URL. |

## HTTP Properties > Advanced Recording Options

Enables you to customize the code generation settings in the area of think time, resetting contexts, saving snapshots, generating **web_reg_find** functions, and masking passwords.

| | |
|---|---|
| **To access** | **Record > Recording Options > HTTP Properties > Advanced** |
| **Important information** | • This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223.<br>• Some options in this node are not available in multi-protocol scripts. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Headers... | Opens the "Headers Dialog Box" on page 198. |
| Content Types... | Opens the "Content Type Filters Dialog Box" on page 199. |
| Non-Resources... | Opens the "Non-Resources Dialog Box" on page 199. |
| **Reset context for each action** | Resets all HTTP contexts between actions. Resetting contexts allows the Vuser to more accurately emulate a new user beginning a browsing session. This option resets the HTML context, so that a context-less function is always recorded in the beginning of the action. It also clears the cache and resets the user names and passwords.<br>Available only for Web and Oracle NCA protocols |
| **Save snapshot resources locally** | Saves a local copy of the snapshot resources during record and replay, thereby creating snapshots more accurately and displaying them quicker. |

| | |
|---|---|
| **Generate web_reg_find functions for page titles** | Generates **web_reg_find** functions for all HTML page titles. VuGen adds the string from the page's title tag and uses it as an argument for **web_reg_find**.<br>• **Generate web_reg_find functions for sub-frames.** Generates **web_reg_find** functions for page titles in all sub-frames of the recorded page.<br>Available only for Web and Oracle NCA protocols |
| **Add comment to script for HTTP errors while recording** | Adds a comment to the script for each HTTP request error. An error request is defined as one that generated a server response value of 400 or greater during recording. |
| **Support charset** | • **UTF-8.** Enables support for UTF-8 encoding. This instructs VuGen to convert non-ASCII UTF-8 characters to the encoding of your locale's machine in order to display them properly in VuGen. You should enable this option only on non-English UTF-8 encoded pages. The recorded site's language must match the operating system language. You cannot record non-English Web pages with different encodings (for example, UTF-8 together with ISO-8859-1 or shift_jis) within the same script.<br>• **EUC-JP.** If you are using Japanese Windows, select this option to enable support for Web sites that use EUC-JP character encoding. This instructs VuGen to convert EUC-JP strings to the encoding of your locale's machine in order to display them properly in VuGen. VuGen converts all EUC-JP (Japanese UNIX) strings to the SJIS (Japanese Windows) encoding of your locale's machine, and adds a **web_sjis_to_euc_param** function to the script. (Kanji only) |

| | |
|---|---|
| **Parameterize server names** | (Web - HTTP/HTML only) VuGen identifies server names and IP addresses when you regenerate a Vuser script. These server names and IP addresses are contained in specific arguments associated with specific functions in the Vuser script. [See the table below for details.] When this option is enabled, VuGen replaces the identified server names and IP addresses with parameters. Parameterizing server names and IP addresses enables you to run the Vuser script in different environments by simply changing the server and IP address values in the parameter file. For an introduction to parameters, see "Parameterization Overview" on page 357.<br><br>To identify data for parameterization, VuGen searches the arguments that are listed for the following functions:<br>• **web_url**: URL; Referrer<br>• **web_custom_request**: URL; Referrer<br>• **web_image**: URL; Referrer<br>• **web_submit_data**: Action; URL; Referrer<br>• **web_submit_form**: Action; URL; Referrer<br>**Note:** By default, this option is not selected. |
| **Generate steps with missing responses** | Generate steps for HTTP requests that are missing server responses. |
| **Generate web_add_cookie functions** | Detect the time when a cookie is created, and generate **web_add_cookie** or **web_add_cookie_ex** functions. If you clear this option, the above functions will not appear in the script. |
| **Generate steps for WebSocket traffic** | Generate code and correlate WebSocket functions. If you clear this option, WebSocket functions will not appear in the script. |
| **Replace passwords with masked parameters** | When generating a script, replace actual passwords with a masked string. |
| **Generate API calls for specific HTTP status codes** | Generates API calls for the specified custom (non-standard) HTTP status codes. You should separate multiple entries with semicolons. For example: 302;303;304 |
| **Use the LoadRunner Proxy to record a local application** | Provides an alternative way to record if the standard VuGen recording mechanism is not compatible with your application. This applies when you have selected to record a Web browser or Windows application. For details, see "Record a Script via a Proxy" on page 235.<br><br>**Note:** After recording, clear this option to restore the default mode. |

| | |
|---|---|
| **Use streaming mode when recording with the LoadRunner Proxy** | Streaming mode enables HTTP data portions received from the server to be forwarded to the application with buffering. This allows you to record asynchronous push communication.<br>**Note:** When this option is enabled, the remote recording toolbar is disabled. |

## Headers Dialog Box

Enables you to automatically send additional HTTP headers with every HTTP request submitted to the server.

| | |
|---|---|
| **To access** | **Record > Recording Options > HTTP > Advanced > Headers** |
| **Important information** | • This dialog box is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223.<br>• The following standard headers are considered risky: Authorization, Connection, Content-Length, Cookie, Host, If-Modified-Since, Proxy-Authenticate, Proxy-Authorization, Proxy-Connection, Referer, and WWW-Authenticate. They are not recorded unless selected in the Header list. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| + | **Plus.** Adds a new entry. |
| − | **Minus.** Deletes an entry. |
| Restore List | Restores the current list to the default values and entries. |
| Use Defaults | Restores all lists to the default values and entries. |
| **<Drop-down menu>** | Controls the options for this dialog box:<br>• **Do not record headers**<br>• **Record headers in list**<br>• **Record headers not in list** |
| **<Header list>** | List of headers which may or may not be recorded. The lists vary depending on which drop-down item is selected. Each item can be selected or deselected using its individual check box. |

## Content Type Filters Dialog Box

Enables you to filter content types for your recorded script. You can specify the type of the content you want to record or exclude from your script.

| | |
|---|---|
| **To access** | **Record > Recording Options > HTTP > Advanced > Content Types** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | **Plus.** Adds a new entry. |
| | **Minus.** Deletes an entry. |
| | Restores the current list to the default values and entries. |
| | Restores all lists to the default values and entries. |
| **<Drop-down menu>** | Controls the options for this dialog box:<br>• **Do not filter content types.**<br>• **Filter content types in list.**<br>• **Filter content types not in list.** |
| **<Header list>** | List of content types which may or may not be filtered. The lists vary depending on which drop-down item is selected. Each item can be selected or deselected using its individual check box. |

## Non-Resources Dialog Box

When you record a script, VuGen indicates whether or not it will retrieve the resource during replay using the Resource attribute in the **web_url** function. If the Resource attribute is set to 0, the resource is retrieved during script execution. If the Resource attribute is set to 1, the Vuser skips the resource type.

You can exclude specific content types from being handled as resources. For example, you can indicate to VuGen that **gif** type resources should not be handled as a resource and therefore be downloaded unconditionally.

| | |
|---|---|
| **To access** | **Record > Recording Options > HTTP > Advanced > Non-Resources** |
| **Important information** | This dialog box is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | **Add.** Adds a new entry to the list. |
| | **Remove.** Deletes an entry from the list. |
| | Restores the default list. |
| **<Non-Resource Content Type list>** | List of items which should not be recorded as resources. Each item can be selected or deselected using its individual check box. |

## Java > VM Recording Options

Enables you to indicate additional parameters to use when recording Java applications.

| | |
|---|---|
| **To access** | **Record > Recording Options > Java Environment Settings > Java VM** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

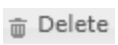
| UI Element | Description |
|---|---|
| **Additional VM Parameters** | List the Java command line parameters here. These parameters may be any Java VM argument. The common arguments are the debug flag (**-verbose**) or memory settings (**-ms**, **-mx**). In additional, you may also pass properties to Java applications in the form of a **-D** flag. For more information about the Java VM flags, see the JVM documentation. |
| **Prepend CLASSPATH to -Xbootclasspath parameter** | Instructs VuGen to add the Classpath before the **Xbootclasspath** (prepend the string). |
| **Use classic Java VM** | Instructs VuGen to use the classic version of VM (for example, not Sun's Java HotSpot). |
| **Use the specified Additional VM Parameters during replay** | Instructs VuGen to use the same Additional VM parameters in replay. |

## Java > Classpath Recording Options

Enables you to specify the location of additional classes that were not included in the system's classpath environment variable. You may need these classes to run Java applications and insure proper recording.

| | |
|---|---|
| **To access** | **Record > Recording Options > Java Environment Settings > Classpath** |
| **Important information** | For Java 8 and later.<br>For the "Java Record Replay Protocol" on page 567, the classpaths are recorded when the script is recorded. Check that no unnecessary paths are recorded and that all necessary paths are recorded correctly. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | **Down Arrow.** Moves a classpath entry down the list. |
| | **Up Arrow.** Moves a classpath entry up the list. |
| | **Add Classpath.** Adds a new line to the classpath list. |
|  | **Delete.** Permanently removes a classpath. |
| **Classpath Entries List** | A list of classpath entries. |

## Microsoft .NET > Recording - Recording Options

This screen enables you to set the recording options for .NET Vuser scripts.

| | |
|---|---|
| **To access** | **Record > Recording Options > Microsoft .NET > Recording** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Code Generation** | Allow you to indicate whether to show warnings, a stack trace, or all event subscriptions during code generation.<br>• **Show Warnings.** Shows warning messages that are issued during the code generation process.<br>• **Show Stack Trace.** Shows the recorded stack trace if it is available.<br>• **Show All Event Subscriptions.** Generates code for all event subscriptions that were recorded. If this option is disabled, VuGen will only generate code for events in which both the publisher (the object which invokes the event) and the subscriber (the object informed of the event) are included in the filter.<br>**Default value:** disabled. |
| **Debug Options** | Enables you to trace the stack and specify its size.<br>• **Stack Trace.** Traces the contents of the stack for each invocation within the script. It allows you to determine which classes and methods were used by your application. This can be useful in determining which references, namespaces, classes, or methods to include in your filter. Enabling the trace may affect your application's performance during recording.<br>**Default value:** disabled<br>• **Stack Trace Limit.** The maximum number of calls to be stored in the stack. If the number of calls exceeds the limit, VuGen truncates it.<br>**Default value:** 20 calls. |
| **Filters** | • **Ignore all assemblies by default.** Ignores all assemblies that are not explicitly included by the selected filter. If you disable this option, VuGen looks for a matching filter rule for all assemblies loaded during the recording. |
| **Logging** | The Logging options let you set the level of detail that is recorded in the recording log file.<br>• **Log severity.** Sets the level of logging to **Errors Only** (default), or **Debug**. The severity setting applies for all the logs that you enable below. You should always use the **Errors Only** log unless specifically instructed to do otherwise by Software Support, since detailed logging may significantly increase the recording time.<br>• **Instrumentation Log.** Logs messages related to the instrumentation process.<br>**Default value:** enabled.<br>• **Recording Log.** Logs messages issued during recording.<br>**Default value:** enabled.<br>• **Code Generation Log.** Logs messages issued during the code generation stage.<br>**Default value:** enabled. |
| **Remote Objects** | For information about this property, see "Remote Objects Property" on the next page. |

| | |
|---|---|
| **Serialization** | • **Serialization format.** The format of the serialization file that VuGen creates while recording a class that supports serialization: **Binary**, **XML**, or **Both**. The advantage of the binary format is that it is more compressed. The advantage of the XML format is that you can manipulate the data.<br>• **Serialize long arrays.** For long arrays containing serializable objects (for example, an array of primitives), use VuGen's serialization mechanism. Enabling this option generates **LrReplayUtils.GetSerializedObject** calls if the array size is equal to or larger than the **Threshold value for long array size**.<br>• **Threshold value for long array size.** The minimum size for an array to be serialized if **Serialize long arrays** is set.<br><br>**Tip:** For XML serialization, you can view the content of the XML file. To view the file, select **View XML** from the right-click menu. |

### Remote Objects Property

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Record in-process objects** | Records activity between the client and server when the server is hosted in the same process as the client. Since the actions are not true client/server traffic, it is usually not of interest. When in-process methods are relevant, for example, in certain Enterprise Service applications, you can enable this option to capture them.<br><br>**Default value:** disabled. |
| **Asynchronous calls** | Specifies how VuGen should handle asynchronous calls on remote objects and their callback methods<br>• **Call original callbacks by default.** Uses the recorded application's original callback when generating and replaying the script. If the callback method is explicitly excluded by a filter, the callback will be excluded even if you enable this option.<br>• **Generate asynchronous callbacks.** This option defines how VuGen will handle callbacks when the original callbacks are not recorded.<br>For more information, see "Asynchronous Calls" on page 615. |

| | |
|---|---|
| **WCF duplex binding** | • **Generate dummy callback handler.** Replaces the original callback in duplex communication with a dummy callback, performing the following actions:<br>  • **Store arguments.** When the server calls the handler during replay, it saves the method arguments to a key-value in memory map.<br>  • **Synchronize replay.** It stops the script execution until the next response arrives. VuGen places the synchronization at the point that the callback occurred during recording. This is represented in the script by a warning:<br>• **Generate unique client base address.** If your application employs dual HTTP Binding, since HTTP is inherently not a duplex protocol, the framework uses a standard port to receive response data being passed to the callback. When you attempt to run multiple instances of your application, you may be unable to do so using the same port number. This option replaces the original client base address's port number with a unique port.<br>For background information about WCF duplex binding, see "Recording WCF Duplex Communication" on page 611. |

## Network > Mapping and Filtering Recording Options

Enables you to set the port mapping and traffic filtering for the recording or code generation. This option lets you include or exclude specific IPs or ports for your Vusers.

| | |
|---|---|
| **To access** | **Record > Recording Options > Network > Mapping and Filtering** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Capture level** | **For port mapping only:** The level of data to capture (relevant only for HTTP based protocols):<br>• **Socket level data.** Capture data using trapping on the socket level only. Port mappings apply in this case (default).<br>• **WinINet level data.** Capture data using hooks on the **WinINet.dll** API used by certain HTTP applications. The most common application that uses these hooks is Internet Explorer. Port mappings are not relevant for this level.<br>• **Socket level and WinINet level data.** Captures data using both mechanisms. WinINet level sends information for applications that use **WinINet.dll**. Socket level sends data only if it determines that it did not originate from **WinINet.dll**. Port mapping applies to data that did not originate from **WinINet.dll**. |

| | |
|---|---|
| **Network-level server address mappings for** | **For port mapping only:** Specifies the mappings per protocol. For example, to show only the FTP mappings, select FTP. |
| **<Port mapping list>** | A list of the port mappings.<br><br>You can temporarily disable the settings for an entry by clearing the check box adjacent to it. When the check box is cleared, VuGen ignores the custom settings for the entry and uses the default settings—it does not cause the host:port to be ignored. To filter a host and port, use Traffic Filtering. |
| **<Traffic filtering list>** | A list of the traffic filters, indicating the server name, port and filtering level.<br>• Click **New Entry** or **Edit Entry** to set these values.<br>• Clear the check box adjacent to an entry to disable it temporarily.<br>• When you launch VuGen, all unchecked port mapping entries are converted into traffic filter entities. |
| New Entry | In the **Port mapping** section: Opens the Server Entry dialog box, allowing you to add a new mapping. For user interface details, see "Server Entry - Port Mapping Dialog Box" on the next page.<br>In the **Traffic filtering** section: Opens the New Entry dialog box, allowing you to add a new traffic filter. For user interface details, see "Server Entry - Traffic Filtering Dialog Box" on page 208. |
| Edit Entry | Opens the Server Entry or New Entry dialog box, allowing you to edit the selected entry. |
| Options... | For port mapping, this button opens the Advanced Settings dialog box to enable auto-detection of the communication protocol and SSL level. For details, see "Advanced Port Mapping Settings Dialog Box" on page 207. |

**Note:**
- In LoadRunner versions prior to 12.02, only port mapping was available, but not traffic filtering.
- If you upgrade from a version of LoadRunner prior to 12.02, the first time you open VuGen, it will automatically convert all of the unchecked port mapping entries into traffic filters.

## See also:

- Port Mapping and Traffic Filtering Overview
- Blog: The Port Mapping dialog – one dialog that does it all

## Server Entry - Port Mapping Dialog Box

Enables you to define a server from the server list in the network port mapping node.

| | |
|---|---|
| **To access** | **Record > Recording Options > Network: Mapping and Filtering > Port Mapping > New Entry / Edit Entry** |
| **Important information** | This dialog box is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| Section | UI Element | Description |
|---|---|---|
| **Socket Service** | **Target Server** | The IP address or host name of the target server for which this entry applies.<br>**Default value:** Any Server. |
| | **Port** | The port of the target server for which this entry applies. Entering **0** specifies all ports.<br>If you do not specify all of the port and server names, VuGen uses the following priorities in assigning data to a service:<br>• Priority 1: port and server specified<br>• Priority 2: port not specified, server specified<br>• Priority 3: port specified, server not specified<br>• Priority 4: port and server not specified<br>A map entry with a high priority does not get overridden by an entry with a lower priority. For example, if you specify that traffic on server twilight using port 25 be handled as SMTP and then you specify that all servers on port 25 be handled as HTTP, the data will be treated as SMTP.<br>• **Forced mapping**. If you specify a mapping for a port number, server name, or combination server:port, VuGen forces the network traffic to use that service. For example, if you were to specify <Any> server on port 80 to use FTP, VuGen uses the FTP protocol to record that communication, even though the actual communication may be HTTP. In this instance, the Vuser script might be empty. |
| | **Service ID** | A protocol or service name used by the recorder to identify the type of connection (i.e. HTTP, FTP, and so on). You can also specify a new name. The name may not exceed 8 characters. |
| | **Service Type** | The type of service, currently set to TCP. |

| | | |
|---|---|---|
| | **Record Type** | The type of recording—directly or through a proxy server. |
| | **Connection Type** | The security level of the connection: **Plain** (non-secure), **SSL**, or **Auto**. If you select Auto, the recorder checks the first 4 bytes for an SSL signature. If it detects the SSL signature, it assumes that SSL is being used. |
| **SSL Configuration** | **SSL Version** | The preferred SSL version to use when communicating with the client application and the server.<br>**Default value:** SSL 2/3. However some services require SSL 3.0 only or SSL 2.0 only. Some new wireless applications require the Transport Layer Security algorithm TLS 1.x, or ALPN. |
| | **SSL Ciphers** | The SSL cipher to use when connecting with a remote secure server.<br>• To enable OpenSSL to choose the default cipher automatically, select **(Default OpenSSL Ciphers)** in the dropdown list .<br>• To define your own cipher and add it to the dropdown list, select **(New SSL Cipher)** and type in the cipher string.<br>• To remove a user-defined cipher, select the cipher in the list and click the button.<br>For more information on OpenSSL cipher strings, refer to the Cipher Strings section in https://www.openssl.org/docs/man1.0.2/apps/ciphers.html |
| | **Use specified client-side certificate** | The default client-side certificate to use when connecting to a remote server. Specify or browse for a certificate file in **txt**, **crt**, or **pem** format, and supply a password. |
| | **Use specified proxy-server certificate** | The default server certificate to present to client applications that request a server certificate. Specify or browse for a certificate file in **txt**, **crt**, or **pem** format, and supply a password. Click **Test SSL** to check the authentication information against the server. |

## Advanced Port Mapping Settings Dialog Box

Enables you to set the advanced port mapping settings. For more information, see "Port Mapping Auto Detection" on page 227.

| | |
|---|---|
| **To access** | **Record > Recording Options > Network > Mapping and Filtering**. Click **Options** in the **Port Mapping** section. |

| | |
|---|---|
| **Important information** | This dialog box is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable auto SSL detection** | Automatically detects SSL communication. Note that this applies only to port mappings that were defined as **auto** in the **Connection type** box, or not defined at all. If a server, port, or server:port combination was defined as either **Plain** or **SSL**, then auto SSL detection does not apply.<br><br>**SSL Version:** Specify the version that you want to detect.<br><br>**SSL Ciphers:** Specify the cipher to detect:<br>• To enable OpenSSL to choose the default cipher automatically, select **(Default OpenSSL Ciphers)** in the dropdown list .<br>• To define your own cipher and add it to the dropdown list, select **(New SSL Cipher)** and type in the cipher string.<br>To remove a user-defined cipher, select the cipher in the list and click the button.<br>For more information on OpenSSL cipher strings, refer to the Cipher Strings section in https://www.openssl.org/docs/man1.0.2/apps/ciphers.html |
| **Enable auto detection of SOCKET based communication** | Automatically detects the type of communication. If required, raise the maximum number of transitions, one at a time until VuGen succeeds in detecting the protocol. You can also gradually increase the maximum buffer size by 1024 bytes (1 KB) at a time until VuGen succeeds in detecting the protocol. This allows VuGen to review a larger amount of data in order to find a signature. |
| **Log Level** | Sets the logging level for the automatic socket detection. |

## Server Entry - Traffic Filtering Dialog Box

Enables you to define a new entry for traffic filtering.

| | |
|---|---|
| **To access** | **Record > Recording Options > Network > Mapping and Filtering > New Entry / Edit Entry** in the **Traffic filtering** section. |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Target server** | The IP address or host name of the target server for which this entry applies. You can use an asterisk (*) as a wildcard to include, for example, multiple servers in a single domain.<br>**Default value:** All Servers. |
| **Regular Expression** | Indicates that the **Target server** string is a regular expression. You can use a regular expression to define an "all but one" filter. For example, you can filter out all servers that do not contain "acme.com" with the following expression: ^(?!,*\.acme\.com). |
| **Port** | The ports to which the filtering should be applied:<br>**All ports**, **Specific port**, or **Port range**. |
| **Filtering level** | Where to apply the filtering:<br>• **Recording.** Filter the selected entries when during recording.<br>• **Code generation.** Filter the selected entries during code generation only. If you enable this option, you will be able to retrieve excluded traffic at a later stage, by modifying the traffic filters before regenerating a script. For details, select **Record > Regenerate Script** and click **Options**. |

## Oracle NCA > Code Generation Recording Options

These recording options enable VuGen to identify Oracle NCA Java objects by ID or by WebUtils name. WebUtil is an implementation on top of the standard communication that uses the Java object to send and receive messages, and cannot be handled as part of LoadRunner's synchronous script.

This feature enables adding Java objects that begin with **webutil** to an array in the header file. The array is called by the **nca_java_set_area_class** function before connecting to the server.

| | |
|---|---|
| **To access** | **Record > Recording Options > Oracle NCA > Code Generation** |
| **Important information** | This node is available only for the Oracle NCA protocol. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Java Object configuration** | Enables Java object configuration.<br>Default: **Selected** |

| | |
|---|---|
| **Generate nca_java_set_area_class** | During recording, creates an array of Java objects that is added to the script's header file.<br><br>The array is then called by a **nca_java_set_area_class** step which is added to the script just before the **nca_connect_server** step.<br><br>This functionality replaces the need for **nca_java_action** steps throughout the script.<br><br>Default: **Selected** |
| **Generate nca_java_set_area_class for objects with webutil name only** | When selected, only the Java objects that begin with **webutil** are added to the array in the header file. Other Java objects are added to the script as default **nca_java_action** steps.<br><br>**Note:** If **Generate nca_java_set_area_class** is selected and this option is not selected, all identified Java objects are added to the array.<br><br>Default: **Selected** |
| **Include nca_java_actions comments in the script** | When selected, for each Java object added to the array in **nca_java_set_area_class**, VuGen also adds a commented out **nca_java_action** step in the expected place the script.<br><br>**Tip:** This is useful if you want to use a **nca_java_action** step instead of a call to the corresponding object item in the array. In this case, uncomment the **nca_java_action** action and comment out the corresponding item in array. |

### See also:

- "Oracle NCA Record and Replay Tips" on page 630

## RDP > Code Generation > Advanced Recording Options

Enables you to control the way VuGen creates an RDP script. Only advanced users are advised to modify these settings.

| | |
|---|---|
| **To access** | **Record > Recording Options > RDP > Code Generation - Adv** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Correlate clipboard parameters** | Replaces the recorded clipboard text sent by the user with the correlated parameter containing the same text as received from the server. |

| | |
|---|---|
| **Double-click timeout (msec)** | The maximum time (in milliseconds) between two consecutive mouse button clicks to be considered a double-click.<br><br>**Default value:** 500 milliseconds. |
| **Prefix for clipboard parameters** | The prefix for clipboard parameters generated in the current script. This is useful when merging scripts, allowing you to specify a different prefix for each script.<br><br>**Default value:ClipboardDataParam_**. |
| **Prefix for snapshot names** | The prefix for snapshot file names generated in the current script. This is useful when merging scripts—you can specify a different prefix for each script.<br><br>**Default value: snapshot_**. |

## RDP > Code Generation > Agent Recording Options

Enables you to control the way the agent for Microsoft Agent for Terminal Server functions with VuGen during recording.

| | |
|---|---|
| **To access** | **Record > Recording Options > RDP > Code Generation - Agent** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Use RDP agent** | Generates script using extended information gathered by the RDP agent. The RDP agent must be installed on the server. For details, see "Install/Uninstall the RDP Agent" on page 651.<br><br>**Note:** To utilize this feature, you must enable the **Replay with RDP agent** runtime settings. For details, see **RDP > RDP Agent** view in the runtime settings. |
| **Enable RDP agent log** | Enables the RDP agent log.<br>• **RDP agent log detail level.** Configures the level of detail generated in the RDP agent log with **Standard** being the lowest level of detail and **Extended Debug** being the highest level of detail.<br>• **RDP agent log destination.** Configures the destination of the RDP agent log data. **File** saves the log messages only on the remote server side. **Stream** sends the log messages to the VuGen machine. **FileAndStream** sends the log messages to both destinations.<br>• **RDP agent log folder.** The folder path on the remote server that the RDP agent log file will be generated in. |

## RDP > Code Generation > Basic Recording Options

Enables you to control the way VuGen creates a script—the level of detail, triggers, and timeouts.

| | |
|---|---|
| **To access** | **Record > Recording Options > RDP > Code Generation - Basic** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Always generate connection name** | If selected, function call will contain the **ConnectionName** parameter. If not selected, the functions will only contain this parameter if more than a single **rdp_connect_server** appears in the script.<br><br>**Default value:** disabled. |
| **Automatic generation of synchronization points** | Synchronization points allow the script to pause in the replay while waiting for a window or dialog to pop-up, or some other control to fulfil a certain condition. This option automatically generates **sync_on_image** functions before mouse clicks and drags (enabled by default). The **Sync radius** is the distance from the mouse operation to the sides of the rectangle which defines the synchronization area. The default is 20 pixels. Select one of the following options:<br>• **None.** No synchronization points are automatically added.<br>• **Rectangular**. Creates synchronization points as rectangular boxes centered around the click or drag location.<br>• **Enhanced**. Creates synchronization points designed to select only a particular location and to react to changes in the UI at that location. For example, a button may be selected and a synchronization point created if the button moves. If a synchronization region is not recognized, the rectangular synchronization settings are used. |
| **Generate mouse movement calls** | Generates **rdp_mouse_move** calls in the script. When enabled, this option significantly increases the script size.<br><br>**Default value:** disabled. |
| **Generate raw keyboard calls** | Generates **rdp_raw_key_up/down** calls as if the script level was set to **Raw**. Mouse calls will still be generated according to the script level. If disabled, VuGen generates Keyboard calls according to the script level. If the script level is set to **Raw**, this option is ignored.<br><br>**Default value:** disabled. |

| UI Element | Description |
|---|---|
| **Generate raw mouse calls** | Generates **rdp_mouse_button_up/down** calls as if the script level was set to Raw. Keyboard calls will still be generated according to the script level. If disabled, VuGen generates Mouse calls according to the script level. If the script level is set to **Raw**, this option is ignored.<br><br>**Default value:** disabled. |
| **Script generation level** | The level of the script and the type of API functions to use when generating the script.<br>• **High.** Generate high level scripts. Keyboard events are translated to **rdp_type** calls. Two consecutive mouse clicks with the same coordinates are translated as a double-click.<br>• **Low.** Generate low level scripts. Key up/down events are translated into **rdp_key** events. Modifier keys (Alt, Ctrl, Shift) are used as a **KeyModifier** parameter for other functions. Mouse up/down/ move events are translated to mouse click/drag events.<br>• **Raw.** Generates a script on a raw level, by extracting input events from network buffers and generating calls in their simplest form: key up/down, mouse up/down/move. The **KeyModifier** parameter is not used. |

## RDP > Client Startup Recording Options

Enables you to set the RDP client startup recording options.

| | |
|---|---|
| **To access** | **Record > Recording Options > RDP > Client Startup** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Run RDP client application** | Connects to the terminal server by running the Terminal Services client. |
| **Use custom connection file** | Connects to the terminal server by using an existing connection file. The file should have an ***.rdp** extension. You can browse for the file on your file system or network. |
| **Use default connection file** | Connects to the terminal server by using the **Default.rdp** file in your document's folder. |

## Recording Properties > Correlation Options - Recording Options

Allows you to enable automatic correlation, and control its depth.

| | |
|---|---|
| **To access** | **Record > Recording Options > Recording Properties > Correlation Options** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Advanced Correlation** | Enables correlation on complex objects such as arrays. This type of correlation is also known as deep correlation.<br>**Default value:** enabled. |
| **Correlate Collection Type** | Correlates objects from the Collection class for JDK 1.2 and higher.<br>**Default value:** disabled. |
| **Correlate String Arrays** | Correlate strings within string arrays during recording. If disabled, strings within arrays are not correlated and the actual values are placed in the script.<br>**Default value:** enabled. |
| **Correlate Strings** | Correlate strings in script during recording. If disabled, the actual recorded values are included in the script between quotation marks and all other correlation options are ignored<br>**Default value:** disabled. |
| **Correlation Level** | Indicates the level of deep correlation, the number of inner containers to be scanned.<br>**Default value:** 15. |

## Recording Properties > Log Options Recording Options

Enables you to determine the level of debug information generated during recording.

| | |
|---|---|
| **To access** | **Record > Recording Options > Recording Properties > Log Options** |
| **Important information** | This node is available only for the Java Record Replay protocol.<br>These options are not supported for the Java Record Replay protocol when using Java 8 and later versions. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Class Dumping** | Dumps all of the loaded classes to the script folder.<br>**Default value:** disabled.<br>**Tip:** Under the script's **data** folder, VuGen creates a subfolder named **dump**. This folder will contain a copy of each class file that was loaded. You can use these class files to determine the signatures when defining custom hooks. |
| **Digest Calculation** | Generate a digest of all recorded objects.<br>**Default value:** disabled.<br>• **Exclude from Digest**. A list of objects not to be included in the digest calculation.<br>**Syntax:** `java.lang.Object class format, delimiter = ","` |
| **Log Level** | The level of recording log to generate:<br>• **None**. No log file is created<br>• **Brief**. Generates a standard recording log<br>• **Detailed**. Generates a detailed log for methods, arguments, and return values.<br>• **Debug.**<br>  • Records hooking and recording debug information<br>  • Redirects java console output to **java_output.log**<br>  • All of the above<br>**Note:** The log files will be stored in the script folder's **data** directory. |
| **Synchronize Threads** | For multi-threaded applications, instructs VuGen to synchronize between the different threads.<br>**Default value:** disabled. |

## Recording Properties > Recorder Options - Recording Options

Enables you to set the Java protocol to record as well as other protocol specific recording options.

| | |
|---|---|
| **To access** | **Record > Recording Options > Recording Properties > Recorder Options** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Byte Array Format** | The format of byte arrays in a script: Regular, Unfolded Serialized Objects, or Folded Serialized Objects. Use one of the serialized object options when recording very long byte arrays.<br>**Default value:** Regular. |
| **Bytes as Characters** | Displays readable characters as characters with the necessary casting—not in byte or hexadecimal form.<br>**Default value:** enabled. |
| **Comment Lines Containing** | Comment out all lines in the script containing one of the specified strings. To specify multiple strings, separate the entries with commas.<br>**Default value:** Any line with a string containing <undefined> will be commented out. |
| **Extensions List** | A comma separated list of all supported extensions. Each extension has its own hooks file.<br>**Default value:** JNDI. |
| **Insert Functional Check** | Inserts verification code that compares the return value received during replay, to the expected return value generated during recording. This option only applies to primitive return values.<br>**Default value:** disabled. |
| **Load Parent Class Before Class** | Change the loading order so that parent classed are loaded before child classes. This helps identify hooking for trees with deep inheritance.<br>**Default value:** enabled. |
| **Record LoadRunner Callback** | Records the LoadRunner stub object as a callback. If disabled, VuGen records the original class as the callback.<br>**Default value:** enabled. |
| **Recorded Protocol** | Specifies which protocol to record, For supported protocols, see "Supported Java communication protocols" on page 567.<br>**Default value:** RMI. |
| **Remove Lines Containing** | Remove all lines containing one of the specified strings from the script. To specify multiple strings, separate the entries with commas. This feature is useful for customizing the script for a specific testing goal. |
| **Unreadable Strings as Bytes** | Represents strings containing unreadable characters as byte arrays. This option applies to strings that are passed as parameters to invocations.<br>**Default value:** enabled. |
| **Use _JAVA_ OPTIONS flag** | Forces JVM versions 1.2 and higher to use the _JAVA_OPTION environment variable which contains the desired JVM parameters.<br>**Default value:** disabled. |

| UI Element | Description |
|---|---|
| **Use DLL hooking to attach LoadRunner support** | Use DLL hooking to automatically attach LoadRunner support to any JVM. |

## Recording Properties > Serialization Options - Recording Options

Enables you to control how objects are serialized. Serialization is often relevant to displaying objects in an ASCII representation in order to parameterize their values.

| | |
|---|---|
| **To access** | **Record > Recording Options > Recording Properties > Serialization Options** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, "Protocol Compatibility Table" on page 223. |
| **See also** | "Java Scripts Correlation - Serialization" on page 278 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Unfold Serialized Objects** | Expands serialized objects in ASCII representation and allows you to view the ASCII values of the objects in order to perform parameterization.<br>• **Limit Object Size (bytes).** Limits serializable objects to the specified value. Objects whose size exceeds this value, will not be given ASCII representation in the script.<br>**Default value:** 3072 bytes.<br>• **Ignore Serialized Objects.** Lists the serialized objects not to be unfolded when encountered in the recorded script. Separate objects with commas.<br>**Syntax:** java.lang.Object class format, delimiter = ","<br>• **Serialization Delimiter.** Indicates the delimiter separating the elements in the ASCII representation of objects. VuGen will only parameterize strings contained within these delimiters. The default delimiter is `#'.<br>• **Unfold Arrays.** Expands array elements of serialized objects in ASCII representation. If you disable this option and an object contains an array, the object will not be expanded.<br>**Default value:** enabled—all deserialized objects are totally unfolded.<br>• **Limit Array Entries.** Instructs the recorder not to open arrays with more than the specified number of elements.<br>**Default value:** 200. |

## RTE > Configuration Recording Options

Enables you to set the recording options to match the character set used during terminal emulation.

| | |
|---|---|
| **To access** | **Record > Recording Options > RTE > Configuration** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Character Set** | Match the character set used during terminal emulation. The default character set is ANSI. For Kanji and other multi-byte platforms, you can specify DBCS (Double-byte Character Set). |

## RTE > RTE Recording Options

Enables you to set the general RTE recording options.

| | |
|---|---|
| **To access** | **Record > Recording Options > RTE > RTE** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Generate automatic synchronization commands** | Automatically generates a number of **TE-synchronization** functions, and insert them into the script while you record.<br>• **Cursor.** Generate a **TE_wait_cursor** function before each **TE_type** function.<br>• **Prompt.** Generate a **TE_wait_text** function before each **TE_type** function (where appropriate).<br>• **X-System.** Generate a **TE_wait_sync** function each time a new screen is displayed while recording.<br>**Note:** VuGen generates meaningful **TE_wait_text** functions when recording VT type terminals only. Do not use automatic **TE_wait_text** function generation when recording block-mode (IBM) terminals. |

| | |
|---|---|
| **Generate automatic X-System transaction** | Records the time that the system was in the X SYSTEM mode during a scenario run. This is accomplished by inserting a **TE_wait_sync_transaction** function after each **TE_wait_sync** function. Each **TE_wait_sync_transaction** function creates a transaction with the name **default**. Each **TE_wait_sync_transaction** function records the time that the system spent in the previous X SYSTEM state. |
| **Generate screen header comments** | Generates screen header comments while recording a Vuser script, and inserts the comments into the script. A generated comment contains the text that appears on the first line of the terminal emulator window.<br><br>**Note:** You can generate comments automatically only when using block-mode terminal emulators such as the IBM 5250. |
| **Keyboard record timeout** | When you type text into a terminal emulator while recording, VuGen monitors the text input. After each keystroke, VuGen waits up to a specified amount of time for the next key stroke. If there is no subsequent keystroke within the specified time, VuGen assumes that the command is complete. |

## SAPGUI > Auto Logon Recording Options

Enables you to log on automatically when you begin recording with the SAP GUI protocol. The logon functions are placed in the vuser_init section of the script.

| | |
|---|---|
| **To access** | **Record > Recording Options > SAPGUI > Auto Logon** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable Auto logon** | Enables you to log on automatically when you begin recording. Enter the **Server name**, **User**, **Password**, **Client** name, and interface **Languages** for the SAP server. |

## SAPGUI > Code Generation Recording Options

Enables you to set the code generation settings for the SAP GUI protocol.

| | |
|---|---|
| **To access** | **Record > Recording Options > SAPGUI > Code Generation** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Always generate Object ID in header file** | Places the Object IDs in a separate header file instead of in the script. When you disable this option, VuGen generates the IDs according to the specified string length in the general script setting. This results in a more compact and cleaner script. |
| **Generate Fill Data steps** | Generates Fill Data steps for table and grid controls—instead of separate steps for each cell. |
| **Generate logon operation as a single step** | Generates a single **sapgui_logon** method for all of the logon operations. This helps simplify the code. If you encounter login problems, disable this option. |

## SAPGUI > General Recording Options

Enables you to set the general recording options for the SAP GUI protocol.

| | |
|---|---|
| **To access** | **Record > Recording Options > SAPGUI > General** |
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see "Protocol Compatibility Table" on page 223. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Capture screen snapshots** | Indicates how to save the snapshots of the SAP GUI screens as they appear during recording: **ActiveScreensnapshots**, **Regular snapshots**, or **None.** ActiveScreen snapshots provide more interactivity and screen information after recording, but they require more resources. |
| **Changing events during recording** | **Process Context menus by text.** Processes context menus by their text, generating **sapgui_toolbar_select_context_menu_item_by_text** functions. When disabled, VuGen processes context menus by their IDs, and generates a **sapgui_toolbar_select_context_menu_item** for context menus. This is an advantage when working with Japanese characters. |

## Traffic Analysis > Traffic Filters Recording Options

This dialog box enables you to filter either incoming or outgoing traffic.

| | |
|---|---|
| **To access** | **Record > Recording Options > Traffic Analysis > Traffic Filters** |

| | |
|---|---|
| **Important information** | This node is available only for specific protocols. For a complete list of protocols and their associated nodes, see the "Protocol Compatibility Table" on page 223. |
| **Relevant tasks** | "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **When generating the script** | **Include all IP addresses in the list.** Includes traffic from specified IP addresses.<br>**Exclude all IP address in the list.** Excludes traffic from specified IP addresses. |
| **<Incoming Traffic Tab>**<br>Enables you to identify IP addresses for incoming traffic | |
| | Adds an IP address. |
| **Source IP** | The IP address of the server. |
| | Deletes an IP address. |
| **<Outgoing Traffic Tab>**<br>Enables you to identify IP addresses for outgoing traffic | |
| | Adds an IP address. |
| **Destination IP** | The IP address of the server. |
| **Destination Port** | The destination port of the server. |
| | Deletes an IP address. |

## WinSock Recording Options

Enables you to set the WinSock recording options.

| | |
|---|---|
| **To access** | **Record > Recording Options > WinSock** |

| | |
|---|---|
| **Important information** | When using translation tables on Solaris machines, you must set the following environment variables on all machines running scripts:<br>`setenv LRSDRV_SERVER_FORMAT 0025`<br>`setenv LRSDRV_CLIENT_FORMAT 04e4` |
| **Relevant tasks** | "Record a Windows Sockets Script" on page 863 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Adds a new entry to the list of excluded sockets. |
| | Removes the selected entry from the list of excluded sockets. |
| **Do not include excluded socket in log** | Excludes the sockets on the list from the log. Clearing this option enables logging for the excluded sockets. Their actions are preceded by "Exclude" in the log file. |
| **Exclude Settings/ Socket List** | The host and port of the sockets to exclude from the recording or regeneration of the script. Use the following syntax:<br>• **host:port** format excludes a specific port.<br>• **host** format excludes all ports for the specified host.<br>• **:port** format excludes a specific port on the local host.<br>• ***:port** format excludes a specific port on all hosts. |
| **Encoding Method** | **Use OEM encoding.** Enable data encoding that supports non-English characters.<br>**Use ASCII encoding.** Enable data encoding that is limited to English characters. Use this option to replicate the LR 9.5x data encoding method. |
| **Think Time Threshold** | During recording, VuGen automatically inserts the think time steps when you pause between actions. You can set a threshold level, below which the recorded think time will be ignored.<br>**Default value:** five seconds. |

| | |
|---|---|
| **Translation tables** | The Translation Table lets you specify the format for recording sessions when using the WinSock single protocol, and for code generation when using a WinSock multi protocol. This applies to users running on mainframe machines or AS/400 servers. Both the server and client machines determine the format of the data from translation tables installed on your system. Select a translation option from the list box.<br><br>The first four digits of the listbox item represent the server format. The last four digits represent the client format.<br><br>The translation tables are located in the **ebcdic** folder under the VuGen's installation folder. If your system uses different translation tables, copy them to the **ebcdic** folder.<br><br>**Note:** If your data is in ASCII format, it does not require translation. Select the **None** option, the default value. |

## Recording Options - Miscellaneous Topics

This section contains a variety of topics relating to recording options.

### Protocol Compatibility Table

The following table lists the Vuser protocols and which recording option nodes are available for each protocol.

| Protocol | Recording Options Nodes |
|---|---|
| **.NET** | • General - Script<br>• Microsoft .NET - Recording, Shared DLLs |
| **TruClient** | • None |
| **C Vuser** | • None |
| **Citrix ICA** | • General - Script<br>• Citrix - Configuration, Recorder, Code Generation, Login<br>**Note:** The Citrix **Login** node is available only when creating a single protocol Citrix script. It is not available when creating a multi-protocol Citrix+Web script. |
| **DNS** | • None |
| **FTP** | • General - Script<br>• Network - Mapping and Filtering |

| Protocol | Recording Options Nodes |
|---|---|
| **Flex** | • General - Recording, Script, Protocols, Code Generation<br>• Flex - RTMP, Configuration, Externalizable Objects<br>• Correlations - Configuration, Rules<br>• HTTP Properties - Advanced<br>• Network - Mapping and Filtering<br>• Data Format Extension - Chain Configuration, Code Generation |
| **IMAP** | • General - Script<br>• Network - Mapping and Filtering |
| **Java over HTTP** | • General - Recording<br>• Correlations - Configuration, Rules<br>• HTTP Properties - Advanced<br>• Network - Mapping and Filtering<br>• Java Environment Settings - Java VM, Classpath<br>• Data Format Extension - Chain Configuration, Code Generation |
| **Java Record Replay** | • Java Environment Settings - Java VM, Classpath<br>• Recording Properties - Recorder Options, Serialization Options, Correlation Options, Log Options<br>**Note:** Log Options are not supported when using Java 8 and later versions. |
| **Java Vuser** | • None |
| **LDAP** | • General - Script |
| **MAPI (Microsoft Exchange)** | • None |
| **MQTT** | • None |
| **ODBC** | • General - Script<br>• Database - Database |
| **Oracle - 2-Tier** | • General - Script<br>• Database - Database |

| Protocol | Recording Options Nodes |
|---|---|
| **Oracle NCA** | • General - Script, Protocols, Recording, Code Generation<br>• HTTP Properties - Advanced<br>• Oracle NCA - Code Generation<br>• Correlations - Configuration, Rules<br>• Network - Mapping and Filtering |
| **Oracle - Web** | • General - Script, Recording<br>• Correlations - Configurations, Rules<br>• HTTP Properties - Advanced<br>• Network - Mapping and Filtering<br>• Data Format Extension - Chain Configuration, Code Generation |
| **POP3** | • General - Script<br>• Network - Port Mapping |
| **RDP (Remote Desktop Protocol)** | • General - Script<br>• RDP - Client Startup, Code Generation (Basic, Advanced, and Agent)<br>• Network - Mapping and Filtering |
| **SAP - Web** | • General - Recording, Script, Protocols, Code Generation<br>• Correlations - Configuration, Rules<br>• HTTP Properties - Advanced<br>• Network - Mapping and Filtering |
| **SAP GUI** | • General - Script<br>• SAP GUI - General, Code Generation, Auto Logon |
| **Siebel - Web** | • General - Recording, Script, Protocols<br>• HTTP Properties - Advanced<br>• Network - Mapping and Filtering<br>• Correlations - Rules |
| **SMTP** | • General - Script<br>• Network - Mapping and Filtering |

| Protocol | Recording Options Nodes |
|---|---|
| **Web - HTTP/HTML** | • General - Recording, Script, Protocols, Code Generation<br>• Correlations - Configuration, Rules<br>• HTTP Properties - Advanced<br>• Network - Mapping and Filtering<br>• Data Format Extension - Chain Configuration, Code Generation |
| **Web Services** | • General - Recording, Script, Protocols, Code Generation<br>• Correlation - Configuration, Rules<br>• HTTP Properties - Advanced<br>• Traffic Analysis - Traffic Filters<br>• Network - Mapping and Filtering |
| **Windows Sockets** | • Sockets - Winsock |

### Port Mapping and Traffic Filtering Overview

When you record a business process, a portion of the generated traffic is not related to the actual business process. For example, the Chrome browser accesses many external servers. This overhead may not be meaningful for the load test.

In addition, as a tester, you may not be interested in some of the generated traffic, even if you generate it during a recording session.

Another issue is a non-Internet business process. If you do not have Internet access, VuGen can successfully record the business process, but it will fail if you use a browser that constantly attempts to access the Internet.

The Port Mapping and Traffic Filtering features allow you to specify the behavior of specific traffic or exclude certain server:port combinations from your Vuser script.

# Port Mapping

When recording Vuser scripts that record network traffic on a socket level (HTTP, SMTP, POP3, FTP, IMAP, Oracle NCA and WinSock), you can set the Port Mapping options. Using these options, you can map the traffic from a specific server:port combination to the desired communication protocol.

The available communication protocols to which you can map are FTP, HTTP, IMAP, NCA, POP3, SMTP, and SOCKET. You create a mapping by specifying a server name, port number, or a complete server:port combination. For example, you can indicate that all traffic from the server *twilight* on port 25, should be handled as SMTP. You can also specify that all traffic from the server called *viper*, should be mapped to the FTP protocol, regardless of the port. Additionally, you can map all traffic on port 23 to SMTP, regardless of the server name.

When recording in multi-protocol mode, If at least one of the protocols records on a socket level, the **Mapping and Filtering** node will be available. Wildcards in the server name are not supported for port mapping.

For details on adding new port mappings, see "Server Entry - Port Mapping Dialog Box" on page 206.

# Traffic Filtering

Traffic filtering extends the capabilities of port mapping by letting you list URLs and ports to exclude. In port mapping, you cannot use wildcards.

Using traffic filtering, you add an entry for each server that you want to exclude. You can use wildcards to exclude all traffic associated with a specific domain.

You may also specify a port or a range of ports. For example, you can filter out only SSL traffic coming through port 443. Once you define an entry, you can clear its check box to temporarily disable it.

You can select a filtering level:

- Recording
- Code generation

The advantage of excluding undesired traffic from the recording session, is that your script will be lighter, increasing its performance.

The benefit of only excluding traffic from the code generation, is that traffic will be recorded and will be accessible to you if you need it at a later point. You can then reapply a different filter without having to rerecord the business process.

For details on adding new traffic filters, see the "Server Entry - Traffic Filtering Dialog Box" on page 208.

## Port Mapping Auto Detection

VuGen's advanced Port mapping options let you configure the **auto-detection** options. VuGen's auto-detection analyzes the data that is sent to the server. It checks the data for a signature, a pattern in the data's content, that identifies the protocol. For the purpose of detecting a signature, all of the send buffers until the first receive buffer, are combined. All send buffers that were sent until a receive buffer is returned, are considered a single data **transition**. By default, no mappings are defined and VuGen employs auto-detection. In some protocols, VuGen determines the type in a single transition, (such as HTTP). Other network protocols require several transitions before determining the type. For this purpose, VuGen creates a temporary buffer for each server-port combination. If VuGen cannot determine the protocol type by reading the first transition buffers, it stores the data in a temporary buffer. It continues to read the incoming buffers until it detects a signature of a specific protocol.

By default, VuGen allows 4 transitions and uses a temporary buffer of 2048 bytes in order to detect a protocol signature. If VuGen has not yet determined the type after reaching the maximum number of transitions, or after reaching the maximum buffer size, it assigns the data to the WinSock protocol. If you did not instruct VuGen to record the WinSock protocol (in the multi-protocol selection), VuGen discards the data.

You can change the maximum number of buffers you want VuGen to read in order to detect the protocol type. You can also specify the size of the temporary buffer. In instances where the amount of data in the first send buffers, is greater than the size of the temporary buffer, VuGen cannot auto-detect the protocol type. In this case, you should increase the size of the temporary buffer.

When working with the above network level protocols, we recommend that you allow VuGen to use auto-detection to determine the protocol type. In most cases, VuGen's recorder is able to recognize the signatures of these protocols. It then automatically processes them according to the protocol specifications. In certain instances, however, VuGen may be unable to recognize the protocol. For example:

- The protocol signature closely resembles an existing protocol, resulting in erroneous processing.
- There is no unique signature for the protocol.
- The protocol uses SSL encryption, and therefore cannot be recognized on a WinSock level.

In all of the above cases, you can supply information to uniquely identify the server and port hosting the protocol.

### EUC-Encoding (Japanese Windows only)

When working with non-Windows standard character sets, you may need to perform a code conversion. A character set is a mapping from a set of characters to a set of integers. This mapping forms a unique character-integer combination for a given alphabet. Extended UNIX Code (EUC) and Shift Japan Industry Standard (SJIS) are non-Windows standard character sets used to display Japanese characters on Web sites.

Windows uses SJIS encoding, while UNIX uses EUC encoding. When a Web server is running UNIX and the client is running Windows, the characters in a Web site are not displayed on the client machine properly due to the difference in the encoding methods. This affects the display of EUC-encoded Japanese characters in a Vuser script.

During recording, VuGen detects the encoding of a Web page through its HTTP header. If the information on the character set is not present in the HTTP header, it checks the HTML meta tag.

If you know in advance that a Web page is encoded in EUC, you can instruct VuGen to use the correct encoding by using the recording options. To record a page in EUC-encoding, enable the **EUC** option in the Recording Options **Recording** node (only visible for Japanese Windows).

Enabling the **EUC** option forces VuGen to record a Web page in EUC encoding, even when it is not EUC-encoded. Therefore, you should only enable this option when VuGen cannot detect the encoding from the HTTP header or the HTML meta tag or when you know in advance that the page is EUC-encoded.

During recording, VuGen receives an EUC-encoded string from the Web server and converts it to SJIS. The SJIS string is saved in the script's **Action** function. However, for replay to succeed, the string has to be converted back to EUC before being sent back to the Web server. Therefore, VuGen adds a **web_sjis_to_euc_param** function before the **Action** function, which converts the SJIS string back to EUC.

In the following example, the user navigates to an EUC-encoded Web page and clicks a link. VuGen records the **Action** function and adds the **web_sjis_to_euc_param** function to the script before the **Action** function.

```
web_sjis_to_euc_param("param_link","Search");
web_link("LinkStep","Text={param_link}");
```

For more information, see "Advanced URL Dialog Box" on page 186.

### Script Generation Preference Overview

Before you record a session, VuGen allows you to specify a language for script generation. The available languages for script generation vary per protocol. The most common available languages are C and Java. By default, VuGen generates a script in the most common language for that protocol, but you can change this through the **Script** recording options node.

For user interface details, see "General > Script Recording Options" on page 188.

> **Tip:** If you record a script in one language, you can regenerate it in another language after the recording. For task details, see "Regenerate a Vuser Script" on page 239.

After you select a generation language, you can enable language-specific recording options which instruct the recorder what to include in the script and how to generate it.

If at least one of the protocols you are recording has multi-protocol capabilities, the Script node will be available except when you record HTTP or WinSock as a single protocol script.

### Script Language Options

When you record a session, VuGen creates a script that emulates your actions. The default script generation language is C. The following list specifies which protocols are appropriate for each language:

| | |
|---|---|
| **C** | For recording applications that use complex COM constructs and C++ objects. |
| **C #** | For recording applications that use complex applications and environments (MS .NET protocol only). |
| **Visual Basic .NET** | For VB .NET applications using the full capabilities of VB. |
| **JavaScript** | For Web-based applications , especially those using dynamic HTML applications. |

After the recording session, you can modify the script with regular C, C#, .NET, or JavaScript code and control flow statements.

### Recording Levels - Overview

VuGen lets you specify what information to record and which functions to use when generating a Vuser script by selecting a recording level in the **General > Recording** node of the **Recording Options** dialog box.

The recording level you select depends on your needs and environment.

## HTML-based script

Generates a separate step for each HTML user action. The steps are intuitive, but they do not reflect true emulation of the JavaScript code.

```
/* HTML-based mode - a script describing user actions*/
...
web_url("WebTours",
        "URL=http://localhost/WebTours/",
        "Resource=0",
        "RecContentType=text/html",
        "Referer=",
        "Snapshot=t1.inf",
        "Mode=HTML",
        LAST);
web_link("Click Here For Additional Restrictions",
        "Text=Click Here For Additional Restrictions",
        "Snapshot=t4.inf",
        LAST);
web_image("buttonhelp.gif",
        "Src=/images/buttonhelp.gif",
        "Snapshot=t5.inf",
        LAST);
...
```

## URL-based script

Records all browser requests and resources from the server that were sent due to the user's actions. Automatically records all HTTP resources as URL steps (**web_url** statements). For normal browser recordings, it is not recommended to use the URL-based mode since is more prone to correlation related issues. However, if you are recording pages such as applets and non-browser applications, this mode is ideal.

URL-based scripts are not as intuitive as the HTML-based scripts since all actions are recorded as **web_url steps** instead of **web_link**, **web_image**, and so on.

```
/* URL-based mode - only web_url functions */
...
web_url("spacer.gif",
        "URL=http://graphics.mylab.com/images/spacer.gif",
        "Resource=1",
        "RecContentType=image/gif",
        "Referer=",
        "Mode=HTTP",
        LAST);
web_url("calendar_functions.js",
        "URL=http://www.im.mylab.com/travelp/calendar_functions.js",
```

```
        "Resource=1",
        "RecContentType=application/x-javascript",
        "Referer=",
        "Mode=HTTP",
        LAST);
...
```

You can switch recording levels and advanced recording options while recording, provided that you are not recording a multi-protocol script. The option of combining recording levels is available to advanced users for performance testing.

You can also regenerate a script after recording, using a different method than the original recording. For example, if your record a script on an HTML-based level, you can regenerate it on a URL-based level. To regenerate a script, select **Record > Regenerate Script** and click **Options** to set the recording options for the regeneration.

## See also:

- For user interface details, see "General > Recording - Recording Options" on page 186.

### Serialization Overview

VuGen uses serialization when it encounters an unknown object during the recording, provided that the object supports serialization. An unknown object can be an input argument which was not included by the filter and therefore its construction was not recorded. Serialization helps prevent compilation errors caused by the passing of an unknown argument to a method. If an object is serialized, it is often advisable to set a custom filter to record this object. For details, see "Serialize Flex Scripts" on page 545.

### Tips for Working with Event Listening and Recording

It can sometimes be difficult to find the ideal listen and recording settings. When defining these settings, keep in mind the following guidelines:

- To record an event on an object, you must instruct VuGen to listen for the event, and to record the event when it occurs. You can listen for an event on a child object, even if a parent object contains the handler or behavior, or you can listen for an event on a parent object, even if the child object contains the handler or behavior.

  However, you must enable recording for the event on the source object (the one on which the event actually occurs, regardless of which parent object contains the handler or behavior).

  For example, suppose a table cell with an **onmouse over** event handler contains two images. When a user touches either of the images with the mouse pointer, the event bubbles up to the cell and includes information on which image was actually touched. You can record this mouse over event by:

  - Setting **Listen** on the WebTable mouse over event to **If Handler** (so that VuGen "hears" the event when it occurs), while disabling recording on it, and then setting **Listen** on the Image mouse over event to **Never**, while setting its recording status to **Enable** (to record the mouse over event on the image after it is listened to at the WebTable level).

- Setting **Listen** on the Image mouse over event to **Always** (to listen for the mouse over event even though the image tag does not contain a behavior or handler), and setting the recording status on the Image object to **Enabled** (to record the mouse over event on the image).

- Instructing VuGen to listen for many events on many objects may lower performance, so try to limit listening settings to the required objects.
- In rare situations, listening to the object on which the event occurs (the source object) may interfere with the event.

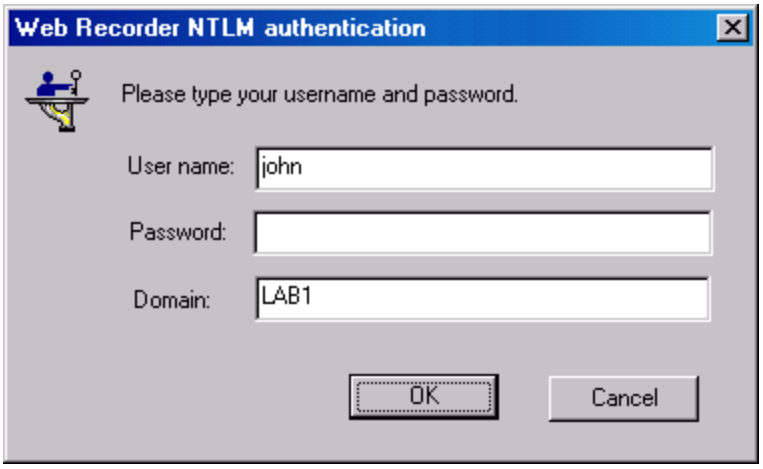
## Providing Authentication Information for Multi-Protocol Scripts

When recording a Web session that uses NTLM authentication, your server may require you to enter details such as a user name and password.

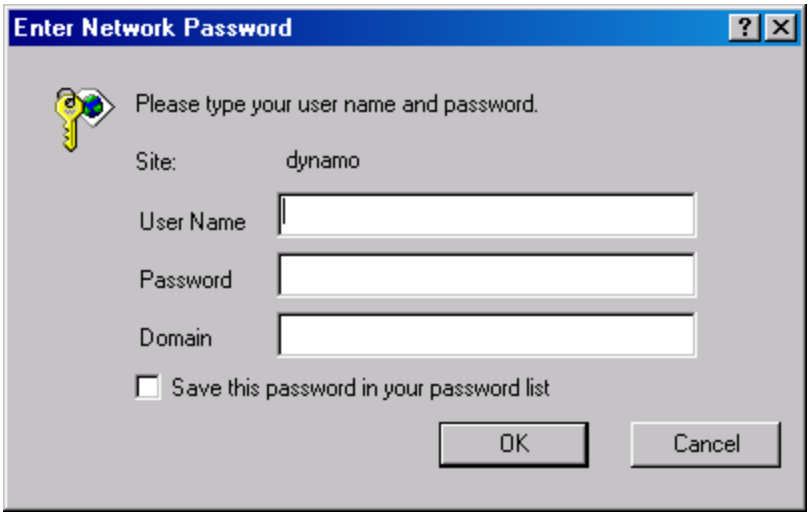
### How Internet Explorer authenticates users

Initially, IE (Internet Explorer) tries to use the NT authentication information of the current user:

- If IE succeeds in logging in using this information and you record a script—then, at the end of the recording VuGen prompts you to enter a password. VuGen retrieves the user name and domain information automatically. If necessary, you can also edit the user name in the Web Recorder NTLM authentication dialog box.




- If IE is unable to log in with the current user's information, it prompts you to enter a user name and password using the standard browser authentication dialog box.




## Generating a web_set_user function

When performing NTLM authentication, VuGen adds a **web_set_user** function to the script.

- If authentication succeeds, VuGen generates a **web_set_user** function with your user name, masked password, and host.

```
web_set_user("domain1\\dashwood",
          lr_unmask("4042e3e7c8bbbcfde0f737f91f"),
         "sussex:8080");
```

- If you cancel the Web Recorder NTLM Authentication dialog box without entering information, VuGen generates a **web_set_user** function for you to edit manually.

```
web_set_user("domain1\\dashwood,
        "Enter NTLM Password Here",
        "sussex:8080");
```

If you enter a password manually, it will appear in the script as-is, presenting a security issue.

### To mask a password:

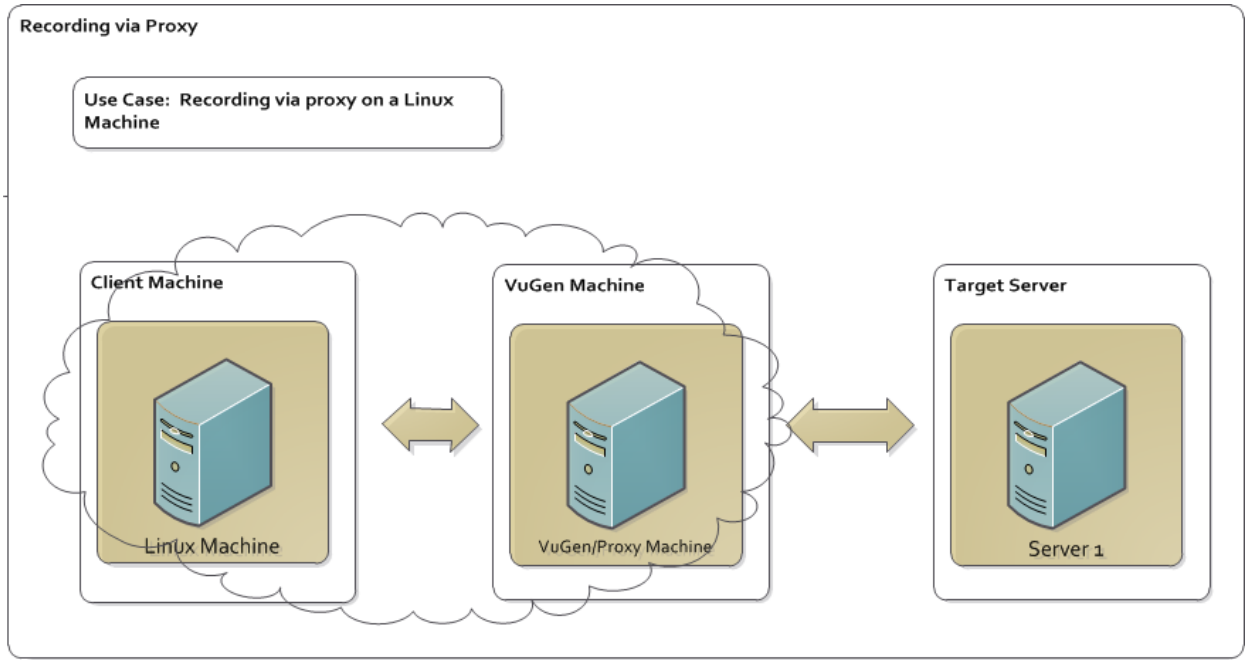
Right-click the password and select **Mask String**. VuGen masks the string and generates an **lr_unmask** function, used to decode the password during replay. For more information about masking strings, see "Encoding Passwords and Text" on page 347.

## Recording via a Proxy - Overview

VuGen allows you to record scripts using a LoadRunner proxy to resolve situations where you cannot install VuGen on the client machine. This may be the case with certain Linux machines, Mac OS machines, and mobile devices.

When using this option, the VuGen machine acts as a proxy server capturing all the traffic from the client machine to the target server. After the business process has been recorded VuGen creates a script.

The following diagram illustrates the basic workflow:




**Considerations for Recording via a Proxy:**

| | |
|---|---|
| **Supported only for some protocols** | Recording via a proxy is not available for all protocols.<br>Partial list of supported protocols: Web - HTTP/HTML, Flex, Java over HTTP, Oracle NCA, and Oracle - Web |
| **Allow proxy configurations** | The client must allow proxy configurations, meaning, you must be able to specify the port and the address of the VuGen machine on the client device or machine. |
| **Same network** | The client device or machine and the VuGen machine must be in the same network. |
| **Delete browsing history** | Make sure to delete the client machine's browsing history prior to recording the business process. This is because VuGen cannot bypass the browser's cache and history settings on the client machine.<br>Deleting the browsing history enables VuGen to accurately record your business process via a proxy. |
| **HTTP forwarding to multiple targets** | If multiple target machines are present, the VuGen proxy can correctly forward data to the right target server according to the Host HTTP header. |

| | |
|---|---|
| **Forwarding to target server via Internet proxy server** | You can configure the VuGen machine to establish a connection with your organization's internet proxy by selecting the **Remote Application via LoadRunner proxy** mode in the **Start Recording** dialog box. For details, see "Start Recording Dialog Box" on page 240. |

## Record a Script via a Proxy

This topic describes various methods for recording a script using a proxy server. In all use cases, the client machine and VuGen machine are in the same network.

### Use Case 1

You want to record a business process but you cannot install VuGen on the client machine or device.

1. Create a new **Web - HTTP/HTML** script.
2. Start Recording.
   a. From the **Start Recording** dialog box, select **Recording Mode > Record > Remote Application via LoadRunner Proxy**. For details, see "Start Recording Dialog Box" on page 240.
   b. Specify the port on which the LoadRunner proxy will listen, by default, port 8888.
   c. Check the **Display recording toolbar on client machine**. This allows you to see and interact with the recording toolbar on the client machine.
   d. Click **Start Recording**.
3. On the client machine, delete the browser cache.
4. Configure the proxy settings to specify the VuGen machine as the proxy server.

   Specify the machine address and port on which the LoadRunner proxy will listen.

   **Below are sample**

| Browser/OS | Path | Configuration |
|---|---|---|
| **Internet Explorer** | • **Internet Options > Connections > LAN Settings > Proxy server**<br>or<br>• **Control Panel and IE Tools > Options menu** | a. Select **Use a proxy server for your LAN**<br>b. Specify **Port**<br>c. Specify **Address** |
| **FireFox** | **Tools > Options > Network > Advanced > Connection > Settings...** | a. Select **Manual proxy configuration**<br>b. Specify **HTTP Proxy**<br>c. Specify **Port**<br>d. Check **Use this proxy server for all protocols** |

5. Record the business process.

a. Navigate to your application.
b. Perform the steps of your business process you want to record.

6. Generate the script.

   Select **Stop Recording** from either the **Recording Toolbar** on the client machine or the **Floating Recording Toolbar** on the VuGen machine. VuGen generates the script.

> **Note:** It is common for business processes to use SSL communication even when not explicitly displaying a URL with HTTPS. In this case, a certificate may be required. Refer to Use Case 3 for more information on obtaining the certificate.

## Use Case 2

You want to record a business process but you cannot install VuGen on the machine (or device) running the application. The client machine requires a proxy to access the Internet.

1. Create a new **Web - HTTP/HTML** script.
2. Start recording.
   a. From the **Start Recording** dialog box, select **Recording Mode > Record > Remote application via LoadRunner Proxy**.
   b. If necessary, change the port on which the LoadRunner proxy listens. The default is port 8888.
   c. Select **Display recording toolbar on client machine**. This allows you to see and interact with the recording toolbar on the client machine.
   d. Set the VuGen Internet Explorer proxy details:

      In Internet Explorer, select **Tools > Internet Options > Connections**. Click **LAN settings** and enter the port and address of the client machine's Internet proxy.

      > **Note:** The **Use automatic configuration script** option is not supported.

   e. Select **Start Recording**.
3. On the client machine, delete the browser cache.
4. Configure the browser settings to use the VuGen's machine IP and port.

**Below are sample**

| Browser/OS | Path | Configuration |
|---|---|---|
| **Internet Explorer** | • **Internet Options > Connections > LAN Settings > Proxy server**<br>• **Control Panel and IE Tools > Options menu** | a. Select **Use a proxy server for your LAN**<br>b. Specify **Port**<br>c. Specify the VuGen IP in **Address** |
| **FireFox** | **Tools > Options > Network > Advanced > Connection > Settings...** | a. Select **Manual proxy configuration**<br>b. Specify **HTTP Proxy**<br>c. Specify **Port**<br>d. Check **Use this proxy server for all protocols** |

5. Record the business process.
   a. Navigate to your application.
   b. Perform the steps of your business process you want to record.
6. Generate the script.

   Select **Stop Recording** from the either the **Recording Toolbar** on the client machine or the **Floating Recording Toolbar** on the VuGen machine. VuGen generates the script .

> **Note:** It is common for business processes to use SSL communication even when not explicitly displaying a URL with **https**. In this case, a certificate may be required. See **Use Case 3** for details on obtaining the certificate.

## Use Case 3

Your application communicates using SSL.

1. Prepare to import the LoadRunner SSL certificate to the client machine.

> **Note:** As an application developer, you can set certain policies on the server certificate when using SSL. However, only if the LoadRunner certificate conforms to the policy, can the client trust the server and the SSL connection be set.

2. Download the certificate, by navigating to:

   **http://<computer name of VuGen machine>:port/proxyroot.cer**

   or

**http://<ip address of VuGen machine>:port/proxyroot.cer**

> **Note:** If you experience security restrictions, navigate to:
>
> **http://<computer name VuGen machine>:port/proxyroot.dat**
>
> or
>
> **http://<ip address of VuGen machine>:port/proxyroot.dat**
>
> After downloading the certificate, change the **.dat** extension back to **.cer** to import the certificate.

3. Import the SSL certificate. The following table provides examples of the path for various browsers.
   - For Internet Explorer, select **Internet Options > Content > Trusted Root Certificate Authorities > Import**.
   - For Firefox, select **Tools > Options > Advanced > Certificates tab > View Certificates > Authorities > Import**.

## Use Case 4

You want to do a proxy recording of a local application that uses the system proxy, where VuGen and the client application are on the same machine.

1. Create a new **Web - HTTP/HTML** script.
2. Set the recording option.

   Open the recording options (**Recording > Recording Options**) and select the **HTTP Properties > Advanced** node. Enable the **Use the LoadRunner Proxy to record a local application**.
3. Start recording.
   a. Open the Start Recording dialog box.
   b. In the **Recording mode** section, select **Record: Web Browser**.
4. Perform the business process.
   a. Navigate to your application.
   b. Perform the steps of your business process you want to record.

## After recording

VuGen automatically resets the proxy back to its original setting after the recording. If the recording did not end in the normal way, for example, if your application crashed during recording, you may need to manually set the proxy back to its original value. To do so, go to **Internet Options > Connections > LAN Settings > Proxy server**.

If you are recording a Java application or a browser other than Internet Explorer, and the application is not using the system proxy settings, you will need to manually set the proxy of the application.

## Import Actions to a Script

For Vuser types that support multiple actions, you can import actions into your script from another Vuser script.

You can import actions from Vusers of the same type only. Any parameters associated with the imported action will be merged with the script.

### To import actions into the current script:

1. Select **Design > Action > Import Action**, or right-click the Solution Explorer and select **Import Action**. The Import Action into VuGen Script dialog box opens.
2. Click **Browse**. A list of the script's actions is displayed in the **Actions to Import** section.
3. Select the actions you want to include and click **Import**. The imported action(s) are displayed in the **Solution Explorer**.

## Regenerate a Vuser Script

If you need to revert back to the originally recorded script, you can regenerate the script. This is ideal for debugging or fixing a corrupted script.

When you regenerate a script, VuGen removes all of the manually added enhancements to the recorded actions in the script. If you added parameters to your script, VuGen restores the original values. The parameter list, however, is not deleted; you can reinsert parameters that you created earlier. Note that regeneration only cleans up the recorded actions, but not those that were manually added.

> **Note:** If your script was imported from a .zip archive file, make sure that it was archived with *all* files. If it was saved only with the runtime files, you will not be able to regenerate the script to its original recorded state. For details, see "How to Work with .zip Files" on page 152.

### To regenerate a Vuser script:

1. Initialize the regeneration.

   Select **Record > Regenerate Script**. VuGen issues a warning indicating that all manual changes will be overwritten.
2. (Optional) Modify regeneration options.

   Click **Options** to open the **Regenerate Options** dialog box.

   - In a multiple protocol script, use the **General > Protocols** node to specify the protocols you want to record when the script is regenerated. For user interface details, see "General > Protocol Recording Options" on page 185.

- To change the Script options, select the **General > Script** node and select or clear the appropriate check box. For user interface details, see "General > Script Recording Options" on page 188.

Click **OK** to close the Regenerate Options dialog box.

3. Indicate whether to include imported actions.

   If your script contains actions imported from another script, the dialog box will contain an option to delete imported actions during the regeneration.

   If you chose to regenerate the script in a different language (using the **General > Script** options in the previous step), then non-recorded actions will automatically be deleted. Non-recorded actions include imported, renamed, or manually added actions.

> **Note:** If a Flex or Java over HTTP Vuser script encounters errors during the code generation phase, VuGen shows the errors in the Error pane. The Error pane displays details about each error, as well as recommended actions. Follow the recommended actions and regenerate the script.

## Start Recording Dialog Box

This dialog box enables you to record your business process.

| | |
|---|---|
| **To access** | • **Record > Record**<br>• [VuGen] **Start Recording** button |
| **Important information** | This dialog box is dynamic and changes according to the options you select and the protocol you are using.<br>• To see all the options, click the **More Options** button in the top right of the dialog box.<br>• To see only the basic options, click **Fewer Options**. |
| **Relevant tasks** | • "Record a Vuser Script" on page 158<br>• "Scripting Options Tab" on page 119<br>• "Record a Script via a Proxy" on page 235<br>• "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723 |

User interface elements are described below:

### All Protocols - recording (except Java)

| UI Element | Description |
|---|---|
| **Record into action** | The section of the script into which you want to record. You can choose one of the built-in action sections: **vuser_init** (for initialization steps), **Action** (for repeatable steps), or **vuser_end** (for sign off steps).<br><br>You can also add your own action. Type the action name in the **Record into action** field and click the **Add** button . The new action is added to the script. |
| **Record** | The mode used to record your business process.<br>• **Web Browser.** For example, Web and Oracle NCA protocols record Web applications.<br><br>**Note:** Recording intranet sites may not work with Microsoft Edge. For information on the workaround for this, see the troubleshooting entry for Recording on Microsoft Edge.<br><br>• **Windows Application.** For example, the Windows Socket protocol records Windows applications.<br>• **Remote Application via LoadRunner Proxy.** (For Internet protocols only) This option allows you to record traffic when VuGen cannot run on the client machine, such as Linux machines, Mac OS machines, and mobile devices. If you choose this mode, you can specify the following options:<br>  • **LoadRunner proxy listens on port:** The port on which the LoadRunner proxy will listen.<br>  • **LoadRunner uses network interface.** For Network Virtualization, enables you to select a network interface if there is more than one interface defined on the VuGen machine.<br>  • **Display recording toolbar on client machine:** Enables you to interact with the recording toolbar on the client machine.<br>• **Captured Traffic File Analysis.** For details about creating a script using a captured traffic file, see Captured Traffic File Analysis below. |

| UI Element | Description |
|---|---|
| **Application** | • For **Web Browser** recording: Select one of the browsers detected on the machine.<br>• For **Windows Application** recording: Specify the path of an executable file.<br><br>**Note:** To run a batch file (a file with a .bat extension), specify *cmd.exe* with its path as the **Application**, and the batch file as the **Program arguments**. |
| **URL address** | The starting URL address.<br>This option is displayed only when you select the **Web Browser** recording mode. |
| **Program arguments** | (**Windows Application** recording mode only) The command line arguments for the executable file specified in **Recorded application**. For example, if you specify `plus32.exe` (recorded application) with the command line options `peter@neptune`, it connects the user `Peter` to the server `Neptune` when starting `plus32.exe`. |
| **Start Recording** (For Internet protocols only) | You can record your business process either:<br>• **Immediately** - Recording starts as soon as you click the **Start Recording** button.<br>• **In delayed mode** - In the following instances, you may not want to record immediately:<br>  • You are recording multiple actions, in which case you only need to perform the startup in one action.<br>  • You want to navigate to a specific point in the application before starting to record.<br>  • You are recording into an existing script. |
| **Working directory** | For applications that require you to specify a working directory. |
| **Recording Options** | Opens the Recording Options dialog box. For user interface details, see "Recording Options" on page 164. |
| **Start Recording button** | Begins to record your business processes based on the option selected above: **Immediately** or **In delayed mode**. |

### Captured Traffic File Analysis

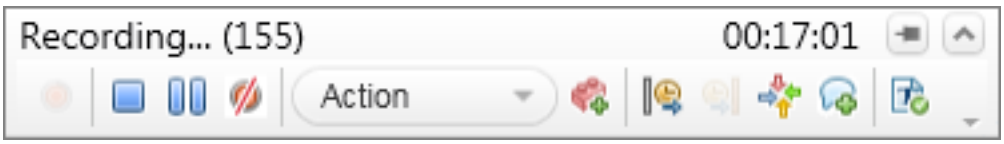
| UI Element | Description |
|---|---|
| **Record into action** | See All Protocols - recording above. |
| **Record** | Select **Captured Traffic File Analysis.** See All Protocols - recording above for details about the other options. |
| **Captured file** | Locate your **pcap**, **saz** (Fiddler), or **har** capture file.<br><br>For details about creating a capture file in a Windows, Linux, or mobile environment, using an external tool such as Wireshark, see "Create a PCAP File" on page 869. |
| **Client side filter** | The IP address of the *client* whose traffic you want to examine. VuGen typically detects the client side filter by analyzing the capture file.<br><br>**Tip:** Use the recording options to set a server side filter (**Recording > Recording Options > Network > Mapping and Filtering**). |
| **SSL Attributes** | A list of the SSL attributes for the servers being analyzed.<br><br>Use the **Add**, **Edit**, and **Remove** buttons to manage the entries.<br><br>The **Add** button opens the Add SSL Attribute dialog box, allowing you to add a server and specify its IP address, port, certificate file, and password if required.<br><br>This list is visible only after you select a **pcap** or **saz** capture file that contains SSL data and requires a certificate. This list is not available for **har** files—instead, configure the SSL through Fiddler. |
| **Start Recording button** | Begins to analyze your captured file. |

### Java Protocols

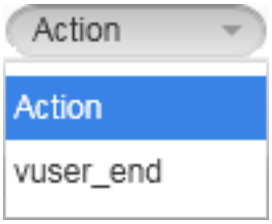
| UI Element | Description |
|---|---|
| **Record into action** | The section into which you want to record. |
| **Record:** | • **Java applet** to record a Java applet through Sun's applet viewer.<br>• **Java application** to record a Java application.<br>• **Internet Explorer** to record an applet within a browser.<br>• **Executable/Batch** to record an applet or application that is launched from within a batch file or the name of an executable file.<br>• **Listener** to instruct VuGen to wait for the batch file that initializes the configuration and runs an application before recording. This mode requires you to define the system variable _JAVA_OPTIONS as --**Xrunjdkhook** using jdk1.2.x and higher. (For JDK 1..x, define the environment variable **_classload_hook=JDKhook**. For JDK 1.6 set _JAVA_OPTIONS as **-agentlib:jdhook**.) |
| **URL address** | The URL to start recording (for Internet Explorer recordings) |
| **Parameters** | Any additional parameters that your application requires. |
| **Main Class** | The *complete* path of the Java class with the main method.<br>**Note:** This option is only present for **Java Application** type applications. |
| **Applet path** | This option is only present for Java applets. |
| **Internet Explorer path** | This option is only present for Internet Explorer type applications. |
| **Executable\Batch** | This option is only present for Executable\Batch type applications. |
| **Working directory** | A working directory is necessary only if your application must know the location of the working directory (for example, reading property files or writing log files). The default is the local LoadRunner/VuGen bin directory. |
| **Recording Options** | Opens the Recording Options dialog box. For user interface details, see "Recording Options" on page 164. |

## Floating Recording Toolbar

The floating recording toolbar enables you to control the recording of Vuser scripts, and provides easy access to common script commands.

| | |
|---|---|
| **UI example** |  |
| **To access** | The floating recording toolbar appears when script recording begins. |
| **Important information** | • The floating recording toolbar is dockable. For details, see "VuGen User Interface" on page 59.<br>• You can pin the toolbar with the button.<br>• You can expand or collapse the toolbar with the and buttons. |
| **Relevant tasks** | "Creating or Opening Vuser Scripts" on page 136 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Continue recording the script after recording has been paused. |
| | Stop recording the script. |
| | Pause recording. |
| | Cancel the recording. |
|  | Select an action to record into. |
| | Create a new action to record into. |
| | Insert a **Start Transaction** step into your script.<br>For details, see "Transaction Overview" on page 338. |

| UI Element | Description |
|---|---|
| | Insert an **End Transaction** step into your script.<br>For details, see "Transaction Overview" on page 338. |
| | Insert a **Rendezvous point** step into your script.<br>For details, see "Rendezvous Points" on page 343. |
| | Insert a comment into your script. |
| | Insert a **Text Check** step into your script (not available for all protocols).<br>For details, see "Text and Image Verification (Web Vuser Scripts) - Overview" on page 731. |
| | Displays:<br>• How many events have been recorded into your script.<br>• The time elapsed since recording began, excluding time the script was paused. |
| /  | Pin or unpin the recording toolbar. |
| /  | Display or hide the toolbar buttons. |
| | Hide the recording toolbar. The toolbar reappears when you refresh or navigate to the next page. Hiding the toolbar may be useful if the toolbar covers controls in the application being operated, thereby preventing access to the controls.<br>**Note:** The **Hide Toolbar** button appears for proxy recording only. |

## Files Generated During Recording

Assuming that the recorded script has been given the name **vuser** and is stored under **c:\tmp**, the following is a list of the more important files that are generated after recording:

| File Name | Details |
|---|---|

| | |
|---|---|
| **vuser.usr** | Contains information about the Vuser type, AUT, action files, and so on.<br><br>**Example:**<br><br>`[General]`<br>`Type=Oracle_NCA`<br>`DefaultCfg=default.cfg`<br>`BuildTarget=`<br>`ParamRightBrace=>`<br>`ParamLeftBrace=<`<br>`NewFunctionHeader=0`<br>`MajorVersion=5`<br>`MinorVersion=0`<br>`ParameterFile=nca_test3.prm`<br>`GlobalParameterFile=`<br>`[Transactions]`<br>`Connect=`<br>`[Actions]`<br>`vuser_init=init.c`<br>`Actions=run.c`<br>`vuser_end=end.c` |
| **vuser.bak** | A copy of Vuser.usr before the last save operation. |
| **default.cfg** | Contains a listing of all runtime settings as defined in the VuGen application (think time, iterations, log, web).<br><br>**Example:**<br><br>`[General]`<br>`XlBridgeTimeout=120`<br>`[ThinkTime]`<br>`Options=NOTHINK`<br>`Factor=1`<br>`LimitFlag=0`<br>`Limit=1`<br>`[Iterations]`<br>`NumOfIterations=1`<br>`IterationPace=IterationASAP`<br>`StartEvery=60`<br>`RandomMin=60`<br>`RandomMax=90`<br>`[Log]`<br>`LogOptions=LogBrief`<br>`MsgClassData=0`<br>`MsgClassParameters=0`<br>`MsgClassFull=0` |

| | |
|---|---|
| **vuser.asc** | The original recorded API calls. |
| **vuser.grd** | Contains the column headers for grids in database scripts. |
| **default.usp** | Contains the script's run logic, including how the actions sections run. |
| **init.c** | Exact copy of the Vuser_init function as seen in the VuGen main window. |
| **run.c** | Exact copy of the Action function as seen in the VuGen main window. |
| **end.c** | Exact copy of the Vuser_end function as seen in the VuGen main window. |
| **vdf.h** | A header file of C variable definitions used in the script. |
| **\Data** | The Data folder stores all of the recorded data used primarily as a backup. Once the data is in this folder, it is not touched or used. For example, Vuser.c is a copy of run.c. |

# Troubleshooting and Limitations for Recording with VuGen

## Proxy recording

- Recording by proxy (local or remote) works only for servers which use HTTP/S communication. It does not work for servers that use only Socket communication.
- If a client-side certificate is required during remote proxy recording, the dialog box requesting the certificate, opens on the VuGen machine, and not on the client machine.
- **Issue:** When recording a session in Chrome, the browser may appear to hang as it continually searches for external links.

  **Workaround:** Manually set the environment's proxy settings in Chrome—do not enable **Automatically detect settings**.

## Security Levels

**Issue:** "Trusted sites" appears in every recorded snapshot.

**Solution:** Open Internet Explorer at least once before recording a script in VuGen.

## Troubleshooting missing steps

**Issue:** Your script is missing steps you recorded.

You encounter the following warning in the **Output Pane > Code generation** tab:

```
Warning: One or more responses are missing or have missing packets. Therefore, a step
may appear to be missing in the script. This issue can be caused if the recording was
stopped before all the responses were received. If the script is generated from a
```

```
.pcap file, check if the file has missing packets.
```

This can be caused when you click **Stop Recording** before all the traffic has been received.

**Solution:** Record the script again. Make sure all pages and resources have been downloaded before clicking the **Stop Record** button.

## Recording on Internet Explorer 10

**Issue:** When recording on Internet Explorer (IE) 10, the browser uses cached pages, and may not record all of the steps.

**Solution:** Each time you begin recording, configure IE 10 to always refresh Web pages from the server. After you begin a recording session, in IE, click F12 to open the Developer Tools pane. In this pane, usually located at the bottom of the browser window, select **Cache > Always refresh from server**.

## Recording on Microsoft Edge

**Issue:** VuGen supports Microsoft Edge (run on Windows 10) to record Web protocols; however, by default, loopback calls are disabled in Windows 10. As a result, Edge is unable to access intranet sites through a proxy server, and consequently, VuGen fails to record intranet applications using Edge.

When you select **Microsoft Edge** as the Web browser in the "Start Recording Dialog Box" on page 240, VuGen will try to enable loopback calls:

- If you have administrator privileges on the machine, VuGen can enable loopback calls, and you can proceed with intranet recording. (The loopback calls will be disabled again at the end of recording.)
- If you do not have administrator privileges, a warning is displayed. If you do not want to record intranet sites, then you can ignore the warning and proceed. If you do try to record intranet sites, Edge will not be able to navigate the sites.

**Solution:** There are two workarounds that allow recording of intranet sites; however, both options require administrator privileges and therefore pose security issues:

- Run VuGen as administrator each time an Edge recording is performed. When you select **Microsoft Edge** as the Web browser, VuGen enables the loopback calls.
- Log in as administrator and enable the loopback call from the command line. Use the following command line setting:

```
checknetisolation loopbackexempt -a -p=S-1-15-2-3624051433-2125758914-
1423191267-1740899205-1073925389-3782572162-737981194
```

## Certificate warning message

When you open VuGen as a non-administrator user, during the recording process you may see a certificate pop-up warning message. The message is automatically closed and does not affect the recording.

## Multi-Protocol recording

If you record a script in the **Init** section and then re-record in the **Actions** sections, compilation may fail. This happens because VuGen creates new header files with each code generation, removing the old ones.

**Workaround:** Re-record the new session in the same section as the first recording.

## Overwriting of data

When recording a Web HTTP/HTML script using WebSockets, if you stop the recording and then resume the recording session, the new data overwrites the original data in the buffer. This is true even if you perform the second recording into a new action.

## Firefox as default browser

If Firefox is set as the default browser, the **Use the default HTTP proxy settings** option (Runtime Settings > Internet Protocol > Proxy) does not work, and a direct connection is used.

## FTP and Active SSL

FTP Active SSL mode is not supported for record or replay.

## Microsoft User Experience Virtualization Issue

When recording, if Microsoft User Experience Virtualization (UE-V) is enabled, it might block the LoadRunner recording (hooking) mechanism and cause recording to fail.

## HSTS Web Recording

If you try to record an HSTS (HTTP Strict Transport Security) enabled site, using an SSL level other than 2/3, you will be unable to navigate within the site.

**Workaround:** Set the SSL level to SSL 2/3 (**Record > Recording Options > Network > Port Mapping > New Entry / Edit Entry**).

## FTP Recording

An FTP recording may generate an empty script.

**Workarounds:** Perform one of the following:

- Configure the FTP server to include the string "FTP" in the welcome message.
- Open the **Network > Mapping and Filtering** recording option node. In the Port mapping area (upper section), click **New Entry**. In the Server Entry - Port Mapping dialog box, set the **Service ID** to **FTP** and specify the FTP server's port number.

### 64-bit Recording

In general, 64-bit applications ported from a 32-bit client version should work identically to the 32-bit client. There is a small risk that new clients will use the power of native 64-bit applications. For example, when using 64-bit long types for Identifiers in DB tables, the identifier value will be cut and the query will fail.

The following guidelines apply:

- The environment for 64-bit recording must be a Windows 7 x64 or Windows 8 x64, and a 64-bit Application Under Test (AUT).
- Recording on 64-bit operating system for 32 and 64-bit applications (running as a 64-bit application) is supported.
- You cannot record a page requiring a client certificate with 64-bit version of Internet Explorer.
- For the Java Over HTTP protocol: JVM 32-bit is required for replay.
- Oracle 2-Tier: Both 32-bit and 64-bit clients need to be installed (the 32-bit client is required for replay).
- For the .NET protocol: There are two available 64-bit types for .NET applications (AnyCPU and pure 64-bit). LoadRunner only supports AnyCPU. There is currently no solution for pure 64-bit applications.
  For replay, LoadRunner uses the same AnyCPU dlls that were used for Recording.

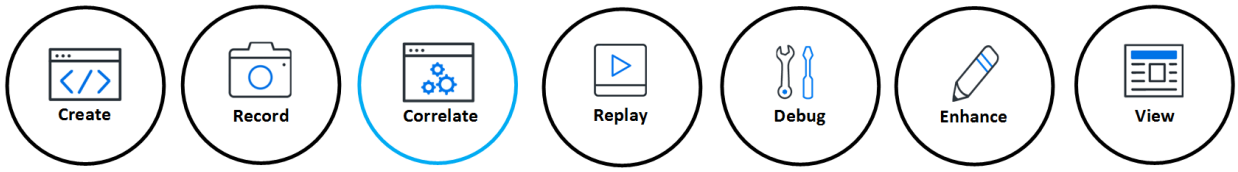
> **Note:** With LoadRunner 11.50 and later, .NET Framework 4 or later should be installed.

# Correlating

## Correlation Overview

Creating a Vuser script includes the steps shown below. This topic provides an overview of the third step, correlating a Vuser script.

Select an image to learn more.




Web pages sent from the server to the browser often contain dynamic data that the client must return in later requests. The process of locating, extracting and replacing recorded dynamic values with values valid at replay is called "correlation".

Common examples of dynamic data are:

- Session ID
- Timestamp
- Customer ID
- Authentication token

This information generally becomes invalid when the session is completed. At the next session, the same items are sent, but the values are different.

After recording the business process in LoadRunner, the script might contain dynamic values in the arguments of the generated API functions. When the script is replayed, these recorded values are sent to the server. However, the web server rejects them because they are not valid in the replay session. Generally, the test fails when this happens.

Example of a dynamic value as received from the server and as it appears in the recorded script:

**Response Body**

**Resulting script**

```
    web_add_auto_header("X-MicrosoftAjax",
        "Delta=true");

    web_add_auto_header("X-Requested-With",
        "XMLHttpRequest");

    web_add_auto_header("X-XSRF-Header",
        "*");

    web_submit_data("Login.aspx_2",
        "Action=http://mypcserver/LoadTest/General/Login.aspx",
        "Method=POST",
        "RecContentType=text/plain",
        "Referer=http://mypcserver/LoadTest/General/Login.aspx",
        "Mode=HTML",
        ITEMDATA,
        "Name=__VIEWSTATE", "Value="
        "/wEPDwUKMTc3MTA3OTE1MxBkZBYCZg9kFgQCAQ9kFgICAQ8WAh4EVGV4dAUyPG1ldGEgaHR0cC1lcXVpdj0iWC1VQS1Db21wYXRpYmxlIi
        "V2lkdGgcHglDb2xsYXBzZWRoHglNYXhIZWlnaHQCkE4eBkhlaWdodBsAAAAAABCLQAEAAAAeCE1pbldpZHRoAhQeDk9yaWdpbmFsSGVpZ2
        "Rpc2FibGVkQ2hlY2tCb3hXcmFwcGVyHxMCAmQWBAIDDw8WBh4TQXNzb2NpYXRlZENvbnRyb2xJRGUfEgUdbGJsRm9yQ2hrQm94IERpc2Fi
        "RkFVTFQeBVZhbHVlBQdERUZBVUxUHghTZWxlY3RlZGdkZA8UKwEBZhYBBXdUZWxlcmlrLldlYi5VSS5SYWRDb21ib0JveEl0ZW0sIFRlbG
        "kuLmFkQ29tYm9Cb3hJdGVtLCBUZWxlcmlrLldlYi5VSSwgVmVyc2lvbj0yMDE0LjEuNDAzLjQwLCBDdWx0dXJlPW5ldXRyYWwsIFB1Ymxp
        "cgUbY3RsNDAkUGFnZUNvbnRlbnQkdXBwZXJQYW5lBSRjdGwwMCRQYWdlQ29udGVudCRjYkF1dG9Mb2dpbiRjaGtCb3gFHGN0bDAwJFBhZ2
        "Name=__VIEWSTATEGENERATOR", "Value=22EB1FB4", ENDITEM,
        "Name=__EVENTVALIDATION", "Value=/wEdAAh6Ar/0pHDJD2/0DpEmoIcua3wFvlN0iY9epBu8OAOiFJYdl2hp8axRYEbe5nC4K%2BTf
        "Name=__ASYNCPOST", "Value=true", ENDITEM,
        "Name=ctl00$PageContent$btnLogin", "Value=Login", ENDITEM,
        LAST);

    lr_think_time(6);
```

To enable successful replay, the script must:

- Locate the dynamic data in the server responses, including headers.
- Extract and save that dynamic data.
- Replace the hard-coded dynamic data from the recording session with the data from the current replay session.

Dynamic data is located using boundaries definitions, regular expressions, attributes, XML queries, or JSON queries as appropriate. The data is extracted and the value is saved to a parameter.

The parameter is used instead of the recorded value in later requests to the server.

Some of the correlation is handled automatically by LoadRunner. You can configure the automatic correlations. For more information, see "Automatic Correlation" on page 256 and "Automatic Correlation Configuration" on page 257.

Dynamic data that is not correlated automatically requires your intervention. For more information, see "Manually Correlate Scripts" on page 260 and "Design Studio" on page 259.

If the script still fails on correlation problems, you can use the advanced techniques. For more information, see "Advanced Correlation Techniques" on page 269.

**Note:** Although parametrization and correlation both use LoadRunner parameters, the purpose is different.

- Parameterization is used to vary the requests based on a list of values you provide. Parameterization is not dependent on preceded responses from the server.

- Correlation uses data from the server in subsequent requests so that the server does not reject the request.

## Correlating Values

The Design Studio's Correlation tab enables you to correlate and manage dynamic values in your web-based Vuser scripts. To learn more about correlation concepts, see "Correlation Overview" on page 251.

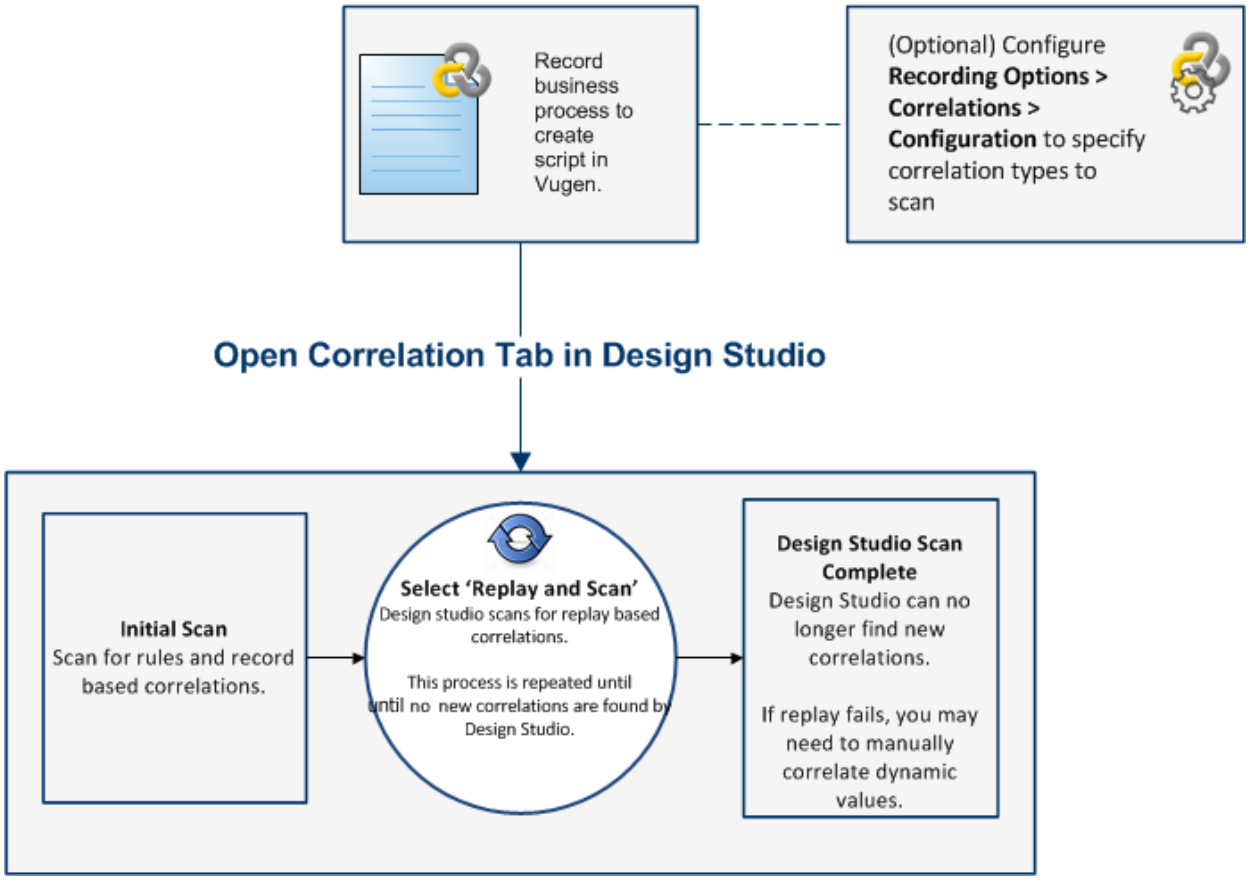
### Overview

With the **Correlation tab** you can:

- Scan for correlations using rules, record based, and replay based engines
- Correlate both raw and formatted data
- Add and edit rules
- Undo correlations
- Review details pertaining to a specific dynamic value in a snapshot

When you record a script using a web-based protocol, many of the values change dynamically each time a request is sent to the server. An example of a dynamic value is a `sessionID` which may include a date and time stamp of when the web session was opened. To learn more, see "Design Studio - Correlation Tab" on page 260.

The following flow chart illustrates the process for correlating values in your script using the Correlation tab:




As you can see from the flow chart, the Correlation tab scans for dynamic values using different processes.

## Correlation Types

Design Studio uses three processes to automatically find dynamic values that may need to be correlated.

| | |
|---|---|
| **Rules** | Design Studio first scans for dynamic values that are defined by rules, if the rules scan has been enabled. To learn more, see "Correlation Rules" on the next page. |
| **Record** | Design Studio scans for dynamic values after code generation. This method can find a significant percentage of dynamic values in your script. |
| **Replay** | Design Studio scans for dynamic values after replay. This method may need to be repeated several times. |

You can select which scan types the Correlation tab should use by configuring **Recording Options > Correlations > Configuration**. In general, it is recommended to enable all scan types.

The following table explains the expected behavior at various script states:

| Script State when opening the Correlation tab | Behavior in the Correlation tab<br>(All scan types enabled) |
|---|---|
| Script contains recorded data. | When Design Studio is opened, it will scan for rule and record based correlations.<br>You can then replay and scan for replay based correlations. Repeat this process until the Design Studio no longer finds new correlations. |
| Script contains recorded data and has been replayed. | When the Correlation tab is opened, it will scan for all correlation types.<br>You can then replay and scan for additional replay based correlations. Repeat this process until Design Studio no longer finds new correlations. |

### Correlation Rules

If you know the dynamic values that need to be correlated before recording, you can create correlation rules that will automatically identify those values while you record. If "**Automatically apply correlation rules**" is selected in the **Recording Options > Correlations > Configuration** node, values found based on rules will automatically be correlated. Additionally, there are some correlation rules that come pre-defined in VuGen for supported application servers. You can enable or disable rules in "Correlations > Rules Recording Options" on page 171.

### Snapshot Details and Occurrences

Design Studio provides details on each snapshot step that contains dynamic values. These details can help you determine which values to correlate in your script. In addition to the snapshot details, the Correlation tab, displays all occurrences of the dynamic value in your script. You can select specific occurrences to correlate or correlate all. For details, see "Design Studio - Correlation Tab" on page 260.

## Automatic Correlation

LoadRunner can detect dynamic values in the script and suggest their replacement with parameters. For automatic correlation, LoadRunner uses rule-based, recording-based, and replay-based correlations.

### Rule-based correlation

LoadRunner comes with a set of predefined, extendable rules for how to correlate dynamic values in well-known environments.

**Example:** An example of a rule in pseudo-code might be:

Find text that appears In the response body between string **__VIEWSTATE" value="** and the nearest **"** character and save it in parameter **"AspNet_ViewState"** to correlate the recorded value.

During code generation, LoadRunner scans server responses for values that match the Correlation Rules. When a matching value is found, LoadRunner adds a step to extract the value and save it in a parameter.

Then, the script code is scanned for the value and matches are replaced with a reference to the parameter.

For information about tuning the rules-based correlation, see "Automatic Correlation Configuration" below.

### Recording-based correlations

After code generation, the script is scanned for additional correlation candidates in the client request steps. If a candidate was received from the server before it is sent by the client, it is suggested as a correlation. The suggestion includes the exact description of where and how to extract the dynamic value from the server response and where to replace data with a parameter.

You can accept suggestions in the "Design Studio" on page 259.

### Replay-based correlations

Replay the script at least once. It is not a problem at this stage if some steps fail.

The Replay-based scan compares the server responses before the failures for mismatched values. If such mismatched values are used in the script, they are suggested as correlations.

Repeat this process until the last failure is detected and correlated.

### See also:

- "Correlations > Configuration Recording Options" on page 169

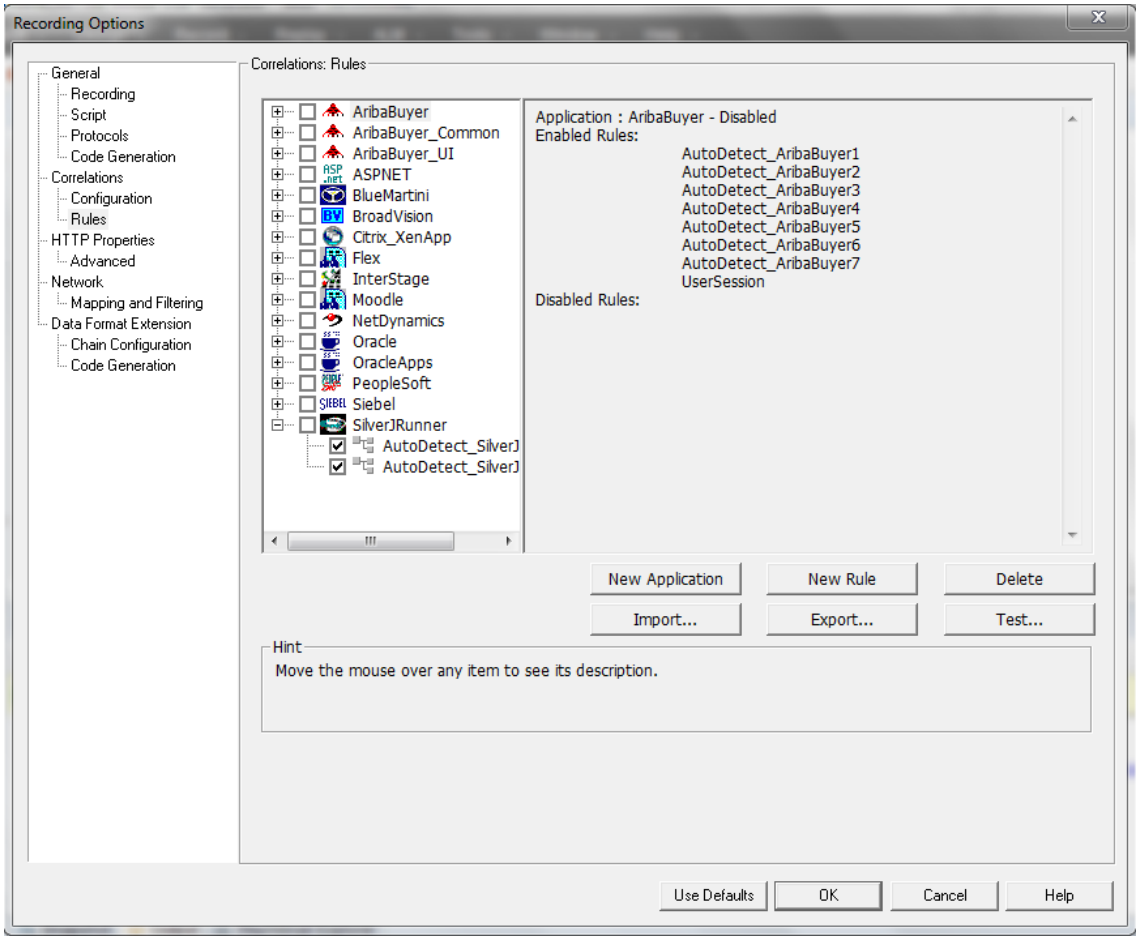
## Automatic Correlation Configuration

By default, all correlation methods are enabled for all LoadRunner users. It is possible to change the correlation settings in the Recording Options dialog.

Automatic correlations are enabled by checking the **Correlation Scan** check box under the **General > Code Generation** node.

To define preferred settings, go to the **Correlations --> Configuration** node.

Correlation rules can be found under the **Correlations --> Rules** node.




The rules are organized in logical groups (“applications”) that represent testing environments like ASP.NET or Citrix.

Correlation rules define where to search and where to replace the dynamic value, extraction method (boundary, regular expression or other) and provide meaningful parameter names to be used in the script for correlations.

Correlation rules can be modified and expanded with new rules. You can test correlation rules before applying them with the Token Substitution Testpad.

You can also exchange and share rules using the Export/Import functionality.

## Exclude Strings or Content Types from the Correlation Scan

By default, the correlation engine scans all plain and html text searching for correlations. Some of the candidates found may not be real correlations. To enhance correlation accuracy, you can configure VuGen to ignore certain text strings, regular expressions, or content types.

1. To exclude a specific text string
   a. Select **Record > Recording Options > Correlations > Configuration**.
   b. Click the View and Edit... button adjacent to **Excluded strings**.
   c. Click the button to open the **Add string to exclude** dialog box.

   For example, enter `getCachedId` to exclude the string `getCachedID` as a correlation candidate.
2. To exclude matches of a regular expression
   a. Select **Record > Recording Options > Correlations > Configuration**.

   Click the View and Edit... button adjacent to **Excluded strings**.
   b. Click the button to open the **Add string to exclude** dialog box.
   c. Enter a regular expression and check the **Regular Expression** box or select the button to view and select from a list box of regular expression character classes and complete the regular expression.

   For example, enter `^navurl:.*` to exclude strings such as `navurl:\\any_char, navurl:1234` as correlation candidates..
3. To exclude content types
   a. Select **Record > Recording Options > Correlations > Configuration**.

   Click the View and Edit... button next to **Excluded content types** to open the **Excluded Content Type List** dialog box. The list shows the content types that are automatically excluded.
   b. Click the button to add a new entry.
4. Delete an excluded item
   a. Highlight an item in the **Excluded String List** or **Excluded Content Type List** dialog boxes.
   b. Click the button.

## Design Studio

The Design Studio serves as a single entry point for all correlation functionality.

The Design Studio is displayed automatically when code generation has finished. It can also be reached manually by clicking on the "Design Studio" button in the toolbar or is accessible via the Design menu.

The Design Studio enables you to manage dynamic values detected by all correlation scans in the unified form and provides complete information about the location, extraction method and the expected replacements in the script.

You can examine the details of the “correlation candidate” and improve the configurations prepared by correlation engines if necessary (for example, correlation boundaries or parameter name).

With Design Studio, you can correlate dynamic values in the script, create correlation rules, undo correlations or discard some correlation suggestion as irrelevant (add these value to excluded strings).

For more information about the Design Studio, see "Design Studio - Correlation Tab" below.

## Manually Correlate Scripts

If the scan for correlation does not resolve all correlation-based errors in your script, you can attempt to manually correlate your script as follows:

1. Search for values that need correlation manually. There are a number of ways to manually search for values that need correlation. For details, see "Search for Values that Need Correlation" on page 269.
2. Correlate the value by doing one of the following:
   - Correlate from snapshots. Highlight the value to correlate, right-click, and select **Create Correlation**.

     When a value is correlated, VuGen adds the correlation parameter and saves the original value in a comment in the script.

     ```
     252 /* Correlation comment - Do not change!  Original value='1' Name ='CorrelationParameter' Type ='Manual' */
     253     lrc_save_rs_param(_Recordset_45,
     254         1,
     255         2,
     256         0,
     257         "CorrelationParameter");
     ```

   - Manually add correlation functions. Manually insert the relevant correlation functions into your script.

### See also:

- "Web Manual Correlations" on page 289
- "Web Manual Correlation in Code" on page 290
- "Winsock Manual Correlation" on page 294

## Design Studio - Correlation Tab

The Design Studio's Correlation tab enables you to scan for, correlate, and view information about dynamic values in your script.

| | |
|---|---|
| **To access** | Click the **Design Studio** button on the VuGen toolbar > **Correlation** tab<br><br>The **Design Studio** button is enabled when you have a recorded script in the Solution Explorer. |

| | |
|---|---|
| **Relevant tasks** | "Correlate Scripts Using Design Studio" on page 268 |
| **See also** | "Correlating Values" on page 254 |

## Correlation Tab - Main Area

| UI Element | Description |
|---|---|
| **Replay&Scan** | Design studio scans for dynamic values using all enabled types: rule, record, and replay. |
| **Correlate** | Replace a dynamic value in the script with a correlation parameter. |
| **Add as Rule** | Add dynamic value definition as a rule:<br>**Rule name.** Enables you to specify a rule name.<br>**Application Name.** Enables you to associate the rule to a specific application.<br>For details, see "Correlations > Rules Recording Options" on page 171. |
| **Undo Correlation** | Replace the correlation parameter with the original dynamic value. |
| **Discard** | Delete the selected dynamic values from the correlation grid. You can only use the discard action when the dynamic value has a status of **New**.<br>In addition, this action adds the text to the list of excluded strings. You can edit the list in **Recording Options > Correlation > Configuration > Excluded string list**. |
| **View** | Filter values found for correlation by the following types:<br>• All<br>• New<br>• Correlated |
| Correlations grid | Displays details about each dynamic value in the script. |
| Details | Displays details about the dynamic value in the snapshot/script selected in the grid. |

## Correlations grid

Displays details about each dynamic value in the script.

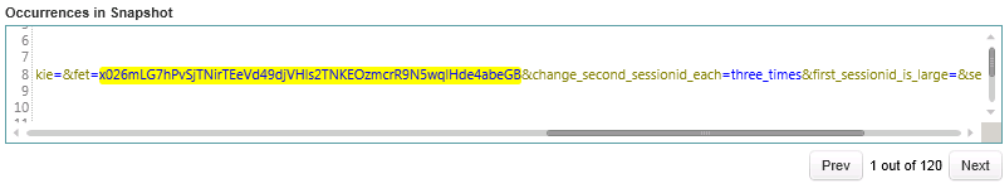
| | |
|---|---|
| **Type** | Displays which engine found the dynamic value for correlations:<br>• Record<br>• Rules<br>• Replay<br>• Manual |
| **Replace/Found** | Displays information about the number of dynamic values found with the same definitions. Since you can perform partial correlation, meaning you can replace specific occurrences, the information displayed depends on if you have correlated the value or not.<br>• Before correlation, this represents: Number of values that can be replaced / Number of values found<br>• After correlation, this represents: Number of values that have been replaced / Number of values found. |
| **Status** | Displays correlation status of the dynamic value from the script: **New** or **Correlated** |
| **Text in Response** | Displays the string of the dynamic value from the script. |
| **Correlation Parameter** | Displays the correlation parameter name of the dynamic value. |

## Details area

Displays details about the dynamic value in the snapshot/script selected in the grid:

**Original Snapshot Step Tab**

| | |
|---|---|
| **Step in Scrips** | **Name.** Displays the step name in the script where the dynamic value was found<br>**Line.** Displays the line of the script where the dynamic value was found.<br>**Action Name.** Displays the name of action from the script where the dynamic value was found.<br>**Description.** Displays a description of the step. |

| | |
|---|---|
| **Correlation Definition** | **Type.** Display API function that will be used to correlate the value.<br>• **Attribute based:** web_reg_save_param_attrib<br>• **Regular expression:** web_reg_save_param_regexp<br>• **Boundary based:** web_reg_save_param_ex<br>**Definition.** Displays the definition of the dynamic value.<br>• Attribute Based. Dynamic value correlation is defined by an **input text value.** Returns a warning if the text is not found.<br>• Regular Expression. Dynamic value correlation is defined by a **regular expression**. A regular expression is a special text string for describing a search pattern.<br>• Boundary Based. Dynamic value correlation is defined by **left and right boundary** text strings. |
| **Apply** | Enables you to select which definition to apply to the dynamic value. You can scroll through the definition of the dynamic value in **Occurrences in Snapshot** by clicking **Prev** or **Next** buttons. |
| **Occurrences in Snapshot** | **Record snapshot.** Displays all the occurrences of the dynamic value in the record snapshot once the script as been replayed. You can scroll to view each occurrence in the snapshot.<br>  <br>**Replay snapshot.** If the scan type of **Replay** has been selected, **Design Studio** displays all the occurrences of the dynamic value in the replay snapshot once the script as been replayed. You can scroll to view each occurrence in the snapshot.<br>**Note:** After the value has been correlated, the replay snapshot is blank. If you modify the **Correlation Definition**, the replay snapshot will be blank. |
| **Options** | Opens the **Recording Options** dialog box.<br>For details, see:<br>• "Correlations > Configuration Recording Options" on page 169<br>• "Correlations > Rules Recording Options" on page 171 |

**Occurrences in Script tab**

Displays the occurrences of the dynamic value in your script. You can correlate all the values or select individual values to correlate by selecting the check box adjacent to the occurrence.

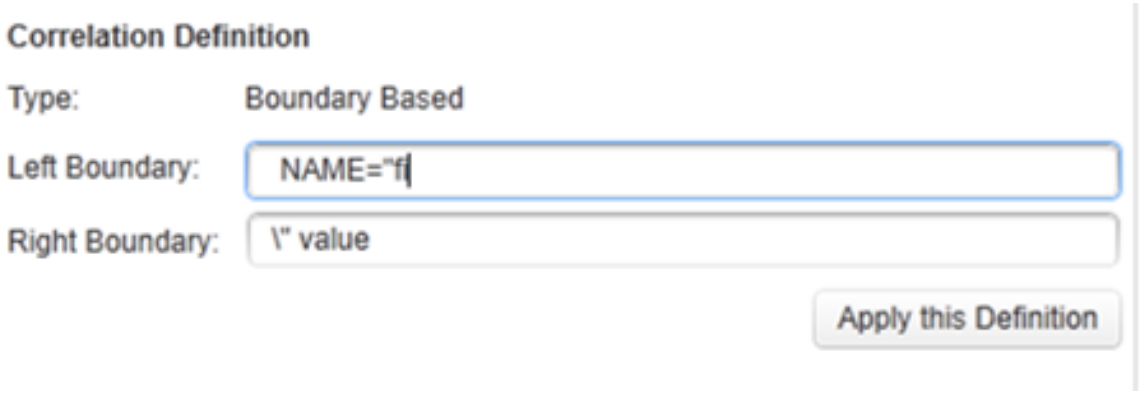
# Modify Correlation Definitions

You can modify correlation definitions to help eliminate dynamic values that do not require correlation. These tasks describe how to modify boundary based, regular expression, and XPath query correlation definitions for record or response correlation.

**Note:** These methods only apply to modifying correlations within your script. They do not affect the correlation rules.

## Modifying Boundary Based Correlation Definitions

1. Click the **Design Studio** button on the VuGen toolbar.
2. Select a dynamic value from the correlation grid and expand **Details**.
3. Edit the **Left Boundary** or **Right Boundary** under the **Correlation Definition** section. You can modify the definition by adding or deleting text.




4. Click **Apply**.

   The **Apply** button is enabled only if the modified boundary definition occurs in the snapshot and the script.

   **Note:** If you do not apply the definition before selecting another dynamic value in the grid, your changes will be lost. If you click **Replay&Scan** before correlating your value with the modified definition, your changes will be lost.

## Modifying Regular Expression Correlation Definitions

1. Click the **Design Studio** button on the VuGen toolbar.
2. Select a dynamic value from the correlation grid and expand **Details**.
3. Edit the **Regular Expression** under the **Correlation Definition** section.
4. Click **Apply**.

The **Apply** button is enabled only if the modified boundary definition occurs in the snapshot and the script.

> **Note:** If you do not apply the definition before selecting another dynamic value in the grid, your changes will be lost. If you click **Replay&Scan** before correlating your value with the modified definition, your changes will be lost.

## Modifying XPath Correlation Definitions

1. Click the **Design Studio** button on the VuGen toolbar.
2. Select a dynamic value from the correlation grid and expand **Details**.
3. Edit the XPath definition under the **Correlation Definition** section.
4. Click **Apply this Definition**.

   The **Apply this Definition** button will not be enabled unless the modified boundary definition occurs in the snapshot and the script.

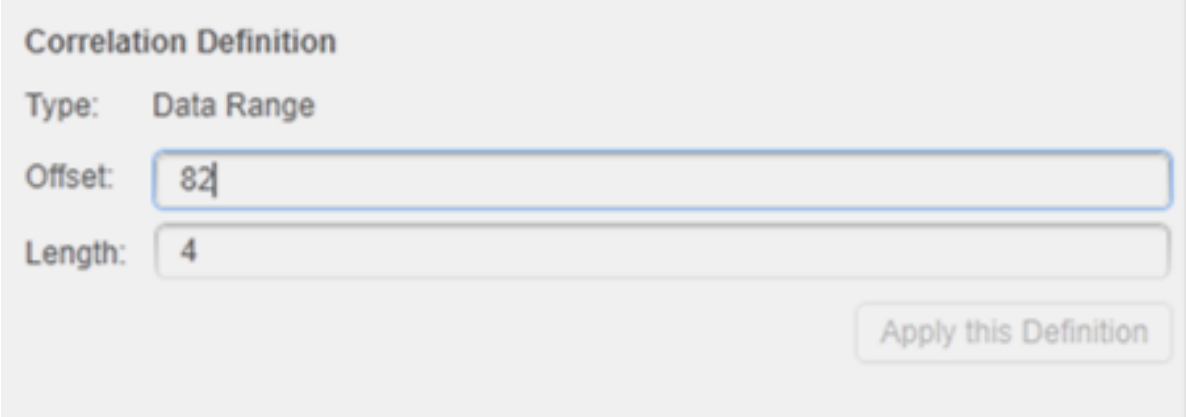
> **Note:** If you do not apply the definition before selecting another dynamic value in the grid, your changes will be lost. If you select **Replay & Scan** before correlating your value with the modified definition, your changes will be lost.

## Modifying Winsocket Correlation Definitions

1. Winsocket dynamic values are correlated from the snapshot. To access, select the relevant step in the **Step Navigator**, and view the step in the **Snaphot pane**. The Winsocket protocol has both a hex and text snapshot.

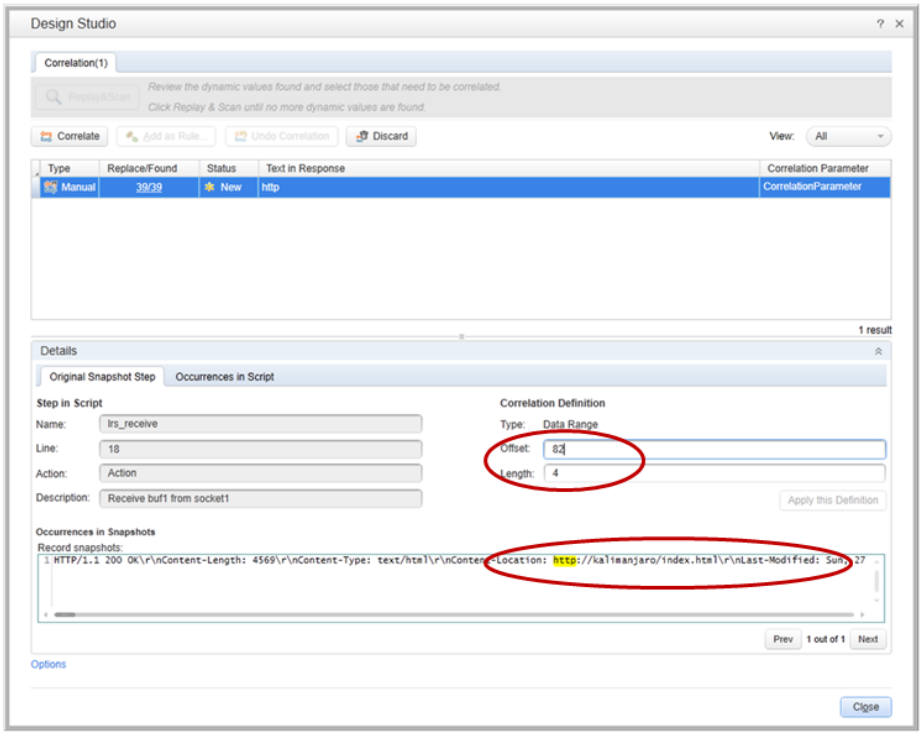
   Right click the value in the snapshot and select **Create correlation** or **Create boundary correlation**.

   This will open the **Design Studio**window.
2. Select a dynamic value from the correlation grid and expand **Details**.
3. If you selected **Create correlation**, edit the Data Range in the **Correlation Definition** section. If you selected **Create boundary correlation**, edit the left of right boundary.

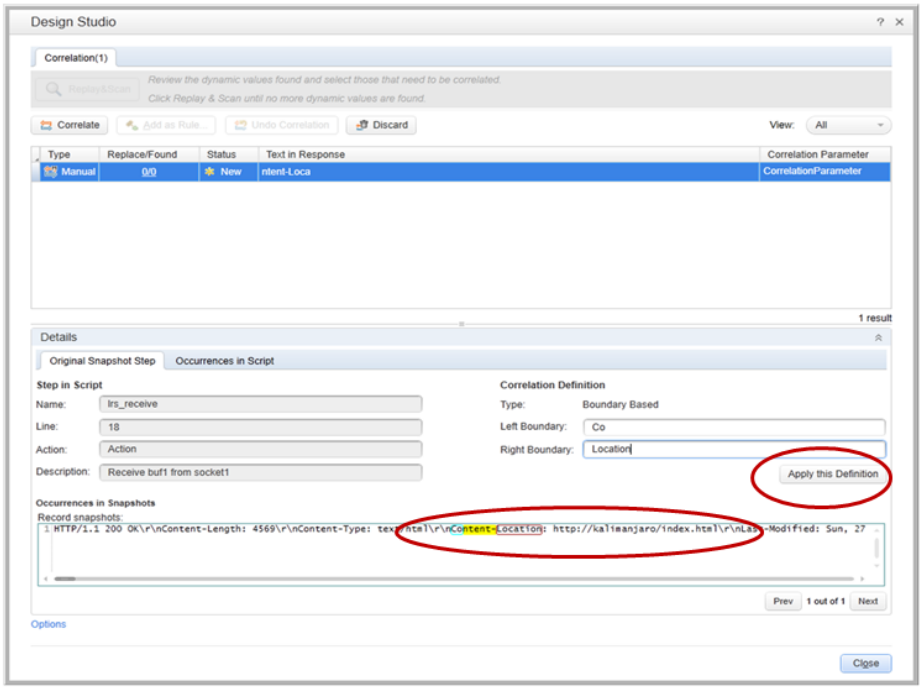


4. Click **Apply**.
   The **Apply** button is enabled only if the modified boundary definition occurs in the snapshot.
   View the following images that display both a Data Range definition and a Boundary definition.

Data Range definition correctly modified




Boundary base definition correctly modified



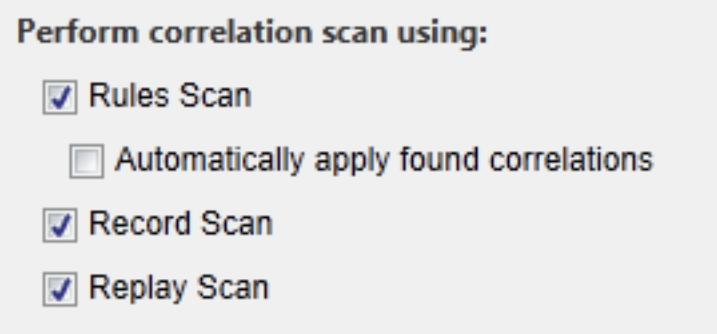
> **Note:** If you do not apply the definition before selecting another dynamic value in the grid, your changes will be lost. If you click **Replay&Scan** before correlating your value with the modified definition, your changes will be lost.

## Correlate Scripts Using Design Studio

1. Prerequisites:
   a. Record a script using one of the following protocols:
      - Web HTTP/HTML
      - Flex
      - RTMP/RTMPT
      - Citrix
      - SAP - Web
      - Oracle NCA

      > **Note:** You can correlate the Web HTTP/HTML steps only within your Oracle NCA script when the Web HTTP/HTML protocol is active. To activate, select **Recording Options > Protocol > Active Protocols > Web HTTP/HTML**.
      >
      > For additional information, see "Oracle NCA Correlation" on page 282

   b. Verify that **Record > Recording Options > Correlations > Configuration** has all scan types enabled.




2. In the Correlation tab, click the **Design Studio** button to perform an initial scan. This will open the Design Studio dialog box which will scan for response (or record based) correlations and apply correlation rules. The progress bar in the dialog box indicates if the initial scan was successful. For details on the Correlation tab, see "Correlating Values" on page 254.

> **Note:** If no dynamic values for correlation are found, a warning is displayed, advising you to

> check your recording options. Verify that **Record > Recording Options > Correlations > Configuration** has all scan types enabled.

3. Select which values to correlate by highlighting the value in the grid and clicking the **Correlate** button.

   When a value is correlated, VuGen adds a **web_reg_save_param_*** function, and saves the original value in a comment in the script.

   You can examine the details of the correlation by expanding the Details Chevron in the dialog box. For details, see "Correlating Values" on page 254.
4. Click the **Add as Rule** button to add a rule type . In addition, you can click the **Edit rule** button to view and edit the corresponding rule if the dynamic value was correlated by a rule. For details, see "Correlations > Rules Recording Options" on page 171.
5. Click the **Replay and Scan** button. The Correlation tab progress bar indicates if additional correlations have been found. Again, you can select which values you would like to correlate by highlighting the value in the grid and clicking the **Correlate** button. You may need to repeat this step several times.
6. When Design Studio no longer finds new correlations, the progress bar will display **Design Studio Scan Complete**.

> **Tip:** If Design Studio does not resolve all correlation-based errors, try to resolve them using manual correlation. For details, see "Manually Correlate Scripts" on page 260.

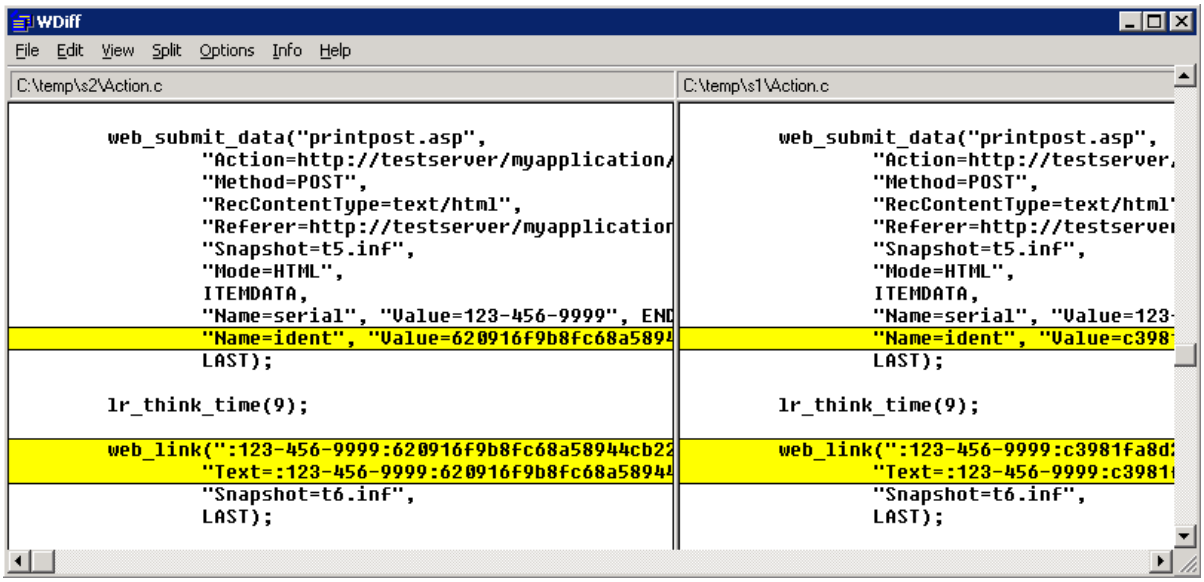
## Advanced Correlation Techniques

If your script continues to fail on correlation problems after you have finished the tasks describe above, you may need to use the advanced techniques described in this section.

### Search for Values that Need Correlation

#### Search by Comparing Scripts

1. Record a script and save it.
2. Create a new script and record the identical operations. Save the script.
3. Select **Tools** > **Compare with Vuser** to compare the scripts. For more details, see "Compare Scripts Side-by-Side" on page 138.
4. Differences in the script are highlighted. Review the differences to determine which ones may require correlation.




> **Note:** *WDiff* is the default utility, but you can specify a custom comparison tool. For more information, see "Compare Scripts Side-by-Side" on page 138.

### Replay Log Search

1. Scan the script in script view for strings that may need correlation such as hash strings, random strings, session ID's, and so on.
2. Search the generation log for the first time that the string appears (this is the response from the server).
3. Search the extended replay log for the same response. Check to see if this response contains a different string within the same boundaries as the original suspected string. If yes, this string requires correlation.

### Wdiff Correlation Utility

The Wdiff Utility lets you compare recorded Vuser scripts and replay results to determine which values need to be correlated.

> **Note:** *WDiff* is the default utility, but you can specify a custom comparison tool. For more information, see "Compare Scripts Side-by-Side" on page 138.

To use *WDiff* effectively, you record the identical operation twice, and compare the scripts (or data files for WinSock). WDiff displays differences in yellow. Note that not all differences indicate a value to correlate. For example, certain receive buffers that indicate the time of execution do not require correlation.

### Modifying Saved Parameters

After you save a value to a parameter, you may need to modify it before using it in your script. If you need to perform arithmetical operations on a parameter, you must change it from a string to an integer using the **atoi** or **atol** C functions. After you modify the value as an integer, you must convert it back to a string to use the new variable in your script.

In the following WinSock example, the data at offset 67 was saved to the parameter, **param1**. Using **atol**, VuGen converted the string to a long integer. After increasing the value of **param1** by one, VuGen converted it back to a string using sprintf and saved it as a new string, **new_param1**. The value of the parameter is displayed using **lr_output_message**. This new value may be used at a later point in the script.

```
lrs_receive("socket2", "buf47", LrsLastArg);lrs_save_param("socket2",
                    NULL, "param1", 67, 5);
lr_output_message ("param1: %s", lr_eval_string("<param1>"));
sprintf(new_param1, "value=%ld", atol(lr_eval_string("<param1>")) + 1);
lr_output_message("ID Number:"%s" lr_eval_string("new_param1"));
```

## Correlation Specifics for Protocols

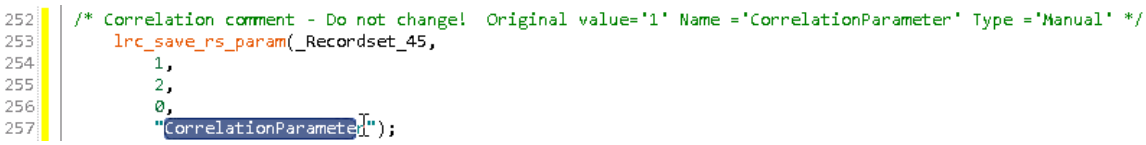
This section contains information that is specific to different protocols.

### COM Correlation

The following steps describe how to correlate COM Vuser scripts.

1. Select **View** > **Output** to display the output tabs at the bottom of the window. Check for errors in the Replay tab. Often, these errors can be corrected by correlation.
2. Select the relevant step in the Step Navigator, and view the step in the **Snaphot pane**.
3. Right click the value in the snapshot and select **Create correlation**. This will open the **Design Studio** window.
4. Select the value you would like to correlate by highlighting it in the grid and clicking the **Correlate** button.

   When a value is correlated, VuGen adds the correlation parameter and saves the original value in a comment in the script.




### C Vuser Scripts Correlation Functions

To correlate statements for protocols that do not have specific functions, you can use the C Vuser correlation functions. These functions can be used for all C-type Vusers, to save a string to a parameter

and retrieve it when required.

### lr_eval_string

Replaces all occurrences of a parameter with its current value.

In the following example, lr_eval_string replaces the parameter row_cnt with its current value. This value is sent to the Output window using lr_output_message.

```
lrd_stmt(Csr1, "select count(*) from employee", -1, 1 /*Deferred*/, ...);
lrd_bind_col(Csr1, 1, =;COUNT_D1, 0, 0);
lrd_exec(Csr1, 0, 0, 0, 0, 0);
lrd_save_col(Csr1, 1, 1, 0, "row_cnt");
lrd_fetch(Csr1, 1, 1, 0, PrintRow2, 0);
lr_output_message("value: %s" , lr_eval_string("The row count is: <row_cnt>"));
```

### lr_save_string

Saves a null-terminated string to a parameter.

To save a NULL terminated string to a parameter, use **lr_save_string**. To save a variable length string, use **lr_save_var** and specify the length of the string to save.

In the following example, lr_save_string assigns 777 to a parameter emp_id. This parameter is then used in another query or for further processing.

```
lrd_stmt(Csr1, "select id from employees where name='John'",...);
lrd_bind_col(Csr1,1,=;ID_D1,...);
lrd_exec(Csr1, ...);
lrd_fetch(Csr1, 1, ...);
/* GRID showing returned value "777" */
lr_save_string("777", "emp_id");
```

### lr_save_var

Saves a variable length string to a parameter.

### See also:

- Function Reference (**Help > Function Reference**).

## Database, RTMP and COM Correlation from a Snapshot

The following steps describe how to correlate scripts from a snapshot.

In order to correlate values from a snapshot, you must first select the **Enable correlation from snapshots** option in the **Scripting > Correlation** section of the "Options Dialog Box" on page 109.

**Note:** When you correlate a value from a snapshot, VuGen may create a boundary-based correlation, even though the recording correlation option is set to use regular expressions.

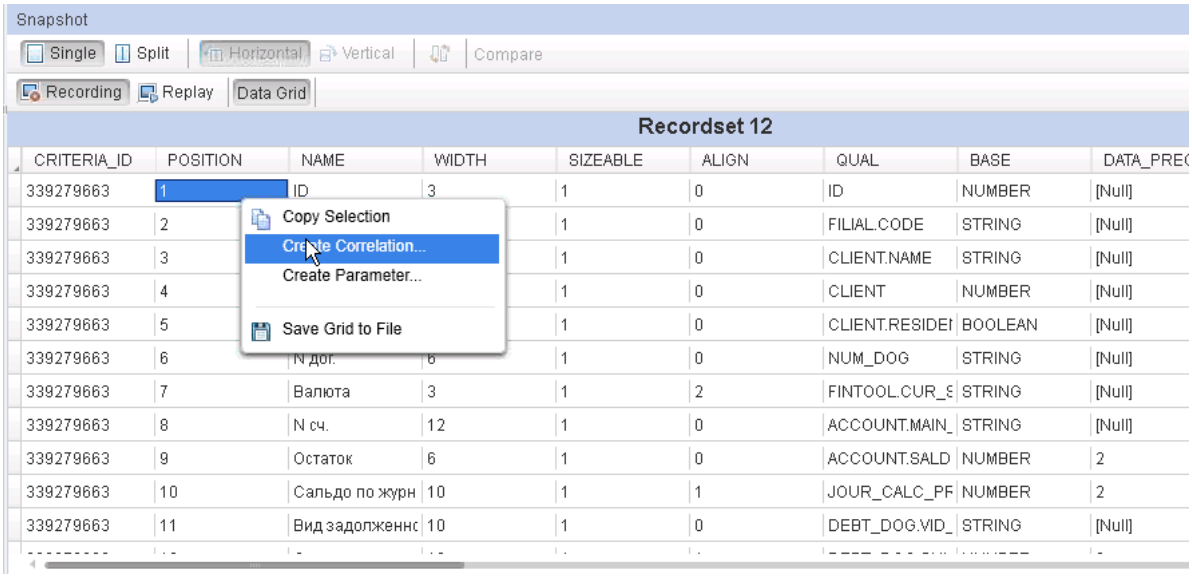
This task applies to the following protocols:

- Database Protocols
- RTMP
- COM Protocols

1. Open the **Output Pane**

   Select **View** > **Output** to display the output tabs at the bottom of the window. Check for errors in the Replay tab. Often, these errors can be corrected by correlation.

2. Select the relevant step in the **Step Navigator**, and view the step in the **Snaphot pane**. Right click the value in the snapshot and select **Create correlation**. This will open the **Design Studio** window.




3. You can select the value you would like to correlate by highlighting it in the grid and clicking the **Correlate** button.
4. When a value is correlated, VuGen adds the correlation parameter, and saves the original value in a comment in the script.

```
/* Correlation comment - Do not change!  Original value='1' Name ='CorrelationParameter' Type ='Manual' */
    lrc_save_rs_param(_Recordset_45,
        1,
        2,
        0,
        "CorrelationParameter");
```

## Flex (RTMP/AMF) Correlation

This task describes how to correlate Flex (RTMP/AMF) Vuser scripts. For details on how to use XPath correlation in Flex Vuser scripts, see "Flex XPath Correlation" on page 276.

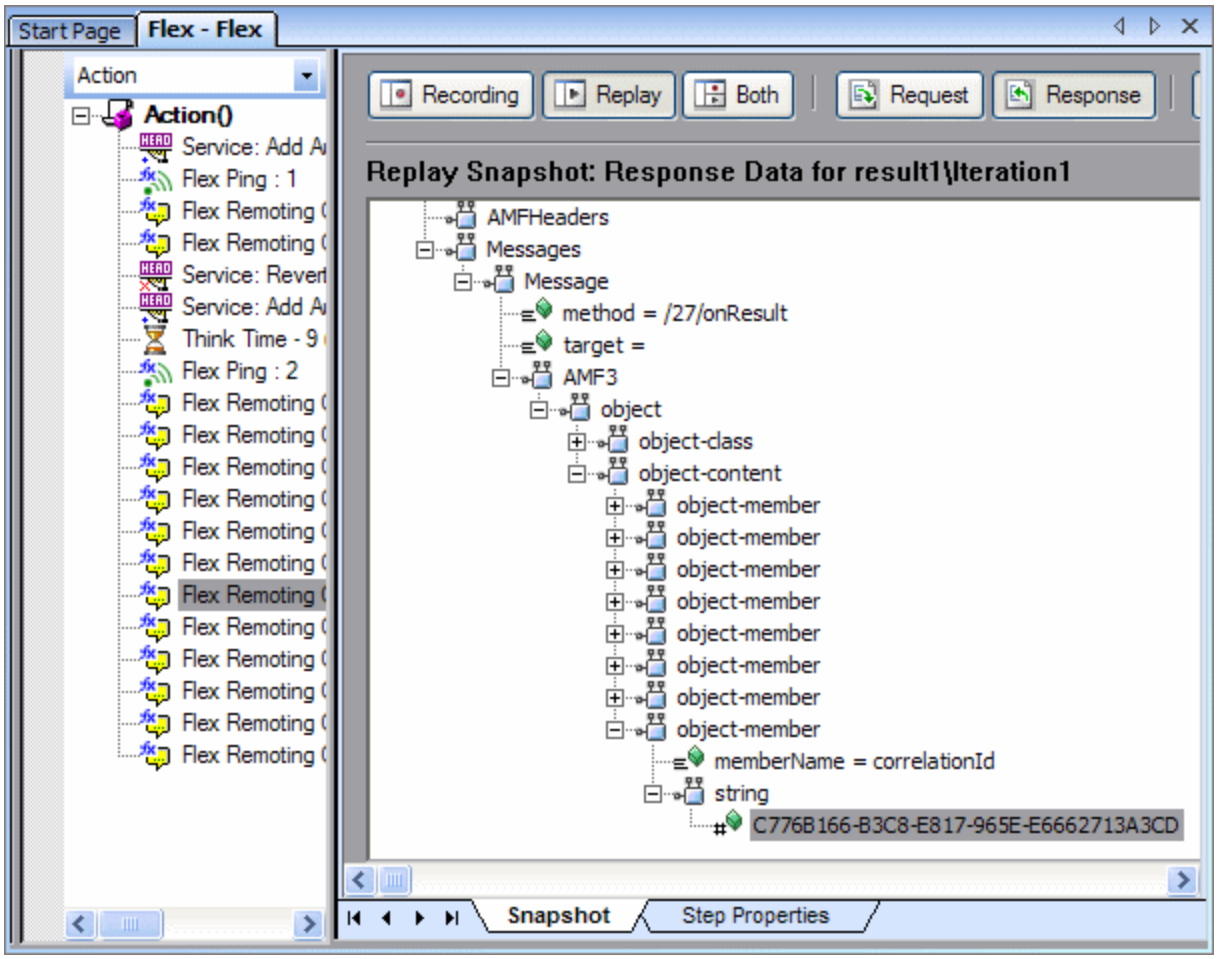
1. Locate the step in your script that failed due to dynamic values that need correlation.

   Use the Replay Log to assist you in finding the problematic step. These errors are not always obvious, and you may detect them only by carefully examining Vuser log files.

   ```
   Action.c(16): Error Server returned error for message #1 : "Incorrect session ID
   sent"/
   Action.c(16): There was an error during the Flex Call ("ConnStatus")
   ```

2. Identify the server response with the correct value in one of the previous steps.

   Double-Click the error in the Replay log to go to the step with the error. Examine the preceding steps in the Step Navigator and look for the value in the **Server Response** tab.



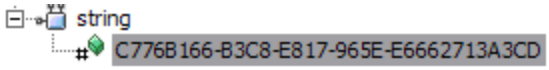

3. Save the entire server response to a parameter.

   Before you extract the value, the entire server response should be saved to a parameter as follows:

   - Right-click the step node (in the left Action pane) corresponding to the server response containing the value and select **Properties**.
   - In the Flex (or AMF) Call Properties dialog box, type a **Response parameter** name.
   - Click **OK** to save the new parameter name.

4. Save the original server response value to a parameter.

- In the XML tree of the Server Response, right-click the node above the value (for example, string), and select **Save value in parameter**.

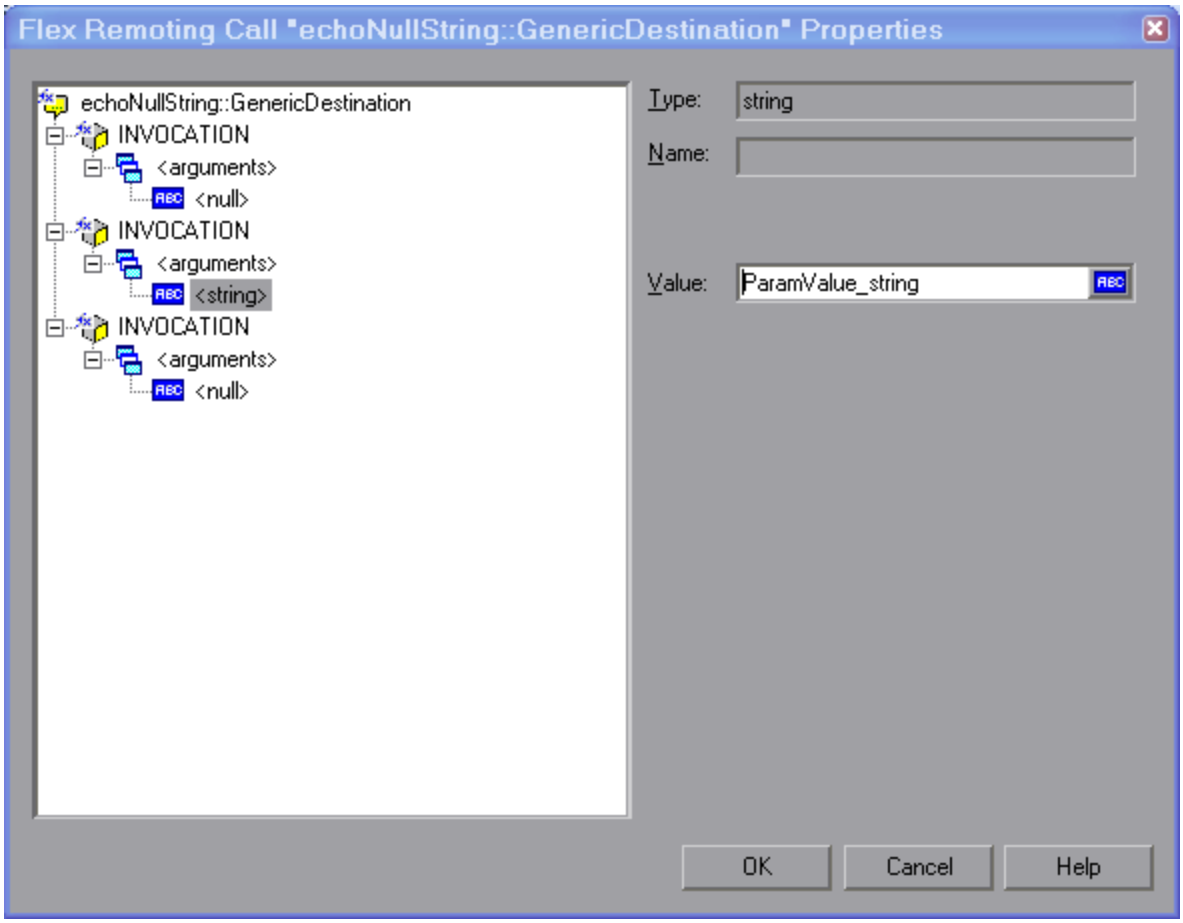



- In the XML Parameter Properties dialog, specify a parameter **Name**. You will use this name in subsequent steps.
- Click **OK**. The script now contains a new function, **lr_xml_get_values**.

5. Insert the parameter in the subsequent calls.

   In the VuGen Editor, beginning with the call that failed, replace the value in all subsequent calls to the object with the parameter that you defined:

   - Right-click the step node (in the Action pane) corresponding to the failed call and select **Properties**.
   - Locate the argument that required correlation.
   - In the **Value** box, type the parameter name in curly brackets, for example, **{ParamValue_string}**.




   Click **OK**.

6. Run the script.

   Make sure that VuGen properly substitutes the argument value with the parameter value that you saved.

## Flex XPath Correlation

This topic describes how to use XPath correlation in Flex Vuser scripts. You use the XML View inside the Snapshot pane to perform the correlation. Before you can successfully implement XPath correlation, you must first configure the recording options.

For details on how to use regular correlation in Flex Vuser scripts, see "Correlate Scripts Using Design Studio" on page 268.

1. Configure the recording options
   a. Select **Record > Recording Options**.
   b. Under **Flex**, click **Externalizable Objects**.
   c. Click **Serialize objects using**, and select **Custom Java Classes**.
   d. Click the **Add jar or zip file** button .
   e. On the LiveCycle installation discs, locate the following three files, and add them to the **Classpath Entries** list:
      i. **flex.jar**
      ii. **flex-messaging-common.jar**
      iii. **flex-messaging-core.jar**

   Ensure that the added files exist in the same location on all load generator machines.
2. Record the Vuser script.
3. In the Editor, click inside the **flex_amf_call** step that contains the data you want to correlate, or in the Step Navigator, double-click the **flex_amf_call** step that contains the data you want to correlate.
4. Click **View > Snapshot** or click the **Snapshot** button on the VuGen toolbar.
5. In the Snapshot pane, click the **Response Body** tab.
6. On the right-side of the Snapshot pane, click the **XML View** tab.
7. In the XML View, locate and select the entire string that contains the dynamic data that requires correlation.
8. Right-click inside the selection, and select **Create Correlation**. The Design Studio opens. For details on how to use Design Studio, see "Correlating Values" on page 254.

When the correlation is complete, VuGen adds a **web_reg_save_parm_xpath** step to the Vuser script.

## Java Custom Correlation

You can manipulate script objects like any Java object. One use of this is to write custom correlation code.

Automatic correlation is not supported in Java over HTTP, but you can use custom correlation as described in this example.

Given this business logic class:

```
public class RmiMessage implements Serializable {
    List<Integer> integers;

    public RmiMessage(List<Integer> myIntegers) {
        this.integers = myIntegers;
    }

    public void setIntegers(List<Integer> myIntegers) {
        this.integers = myIntegers;
    }
}
```

And given the following code in Action.java:

```
String _string9 = "com.comp.lr.testing.rmi.RmiMessage __CURRENT_OBJECT =
   {"
        + "java.util.List myIntegers = {"
             + "super = {"
                  + "super = {"
                  + "}"
                  + "int modCount = #0#"
             + "}"
             + "java.lang.Object a[] = {"
                  + "java.lang.Integer a[0] = {"
    + "super = {"
    + "}"
    + "int value = #4#"
                  + "}"
                  + "java.lang.Integer a[1] = {"
    + "super = {"
    + "}"
```

```
    + "int value = #21#"
                  + "}"
             + "}"
        + "}"
     + "}";
RmiMessage _message = (com.comp.lr.testing.rmi.RmiMessagelr.deserialize(_
   string9, 2);
_remote.exampleMethodCall(_message);
```

After the call to lr.deserialize, you have the business logic object, which can then be manipulated.

In this example, correlation is done by setting the integers list:

```
RmiMessage _message = (com.comp.lr.testing.rmi.RmiMessage) lr.deserialize
   (_string9, 2);
_message.setIntegers(Arrays.asList(4,5,7)); // call business logic methods
   here to adjust object to your needs
_remote.exampleMethodCall(_message);
```

## Java Scripts Correlation - Serialization

In RMI and some cases of CORBA, the client AUT creates a new instance of a Java object using the **java.io.serializable** interface. It passes this instance as a parameter for a server invocation. In the following segment, the instance **p** is created and passed as a parameter.

```
// AUT code:
java.awt.Point p =
    new java.awt.Point(3,7);
map.set_point(p);
```

The automatic correlation mechanism is ineffective here, since the object did not return from any previous call. In this case, VuGen activates the serialization mechanism and stores the object being passed as a parameter. It saves the information to a binary data file under the user folder. Additional parameters are saved as new binary data files, numbered sequentially. VuGen generates the following code:

```
public class Actions {
  // Public function: init
      public int init() throws Throwable {
```

```
            java.awt.Point p =
             (java.awt.Point)lr.deserialize(0, false);
            map.set_point(p);
        }
}
```

The integer passed to **lr.deserialize** represents the number of binary data files in the Vuser folder.

To parameterize the recorded value, use the Java setLocation method (for information, see https://docs.oracle.com/en/java/). The following example uses the **setLocation** method to set the value of the object, p.

```
public class Actions {
  // Public function: init
      public int init() throws Throwable {
          java.awt.Point p = (java.awt.Point)lr.deserialize(0, false);
          p.setLocation(2,9);
          map.set_point(p);
      }
}
```

In certain instances the public method of **setLocation** is not applicable. As an alternative, you can use the API of your class that incorporate get or set accessor methods. If you are working with AUT classes that do not have get/set methods or use private methods, or if you are unfamiliar with the classes' API, you can use VuGen's built-in serialization mechanism. This mechanism allows you to expand objects in their ASCII representation and manually parameterize the script. You enable this mechanism in the Recording Options dialog box. For details, see "Recording Properties > Serialization Options - Recording Options" on page 217.

VuGen generates an **lr.deserialize** method that deserializes the data or displays complex data structures as serial strings. Once the structure is broken down to its components, it is easier to parameterize. The **lr.deserialize** method receives two arguments, a string and an integer. The string is the parameter's value that is to be substituted during replay. The integer is the index number of binary file to load.

If you choose not to expand objects in your script by clearing the Unfold Serialized Objects check box, you can control the serialization mechanism by passing arguments to the lr.deserialize method. The first argument is an integer indicating the number of binary files to load. The second integer is a boolean value:

| | |
|---|---|
| **true** | Use VuGen's serialization mechanism. |
| **false** | Use the standard Java serialization mechanism. |

The following segment shows a generated script in which the serialization mechanism was enabled.

```
public class Actions {
  // Public function: init
```

```
      public int init() throws Throwable {
          _string = "java.awt.Point __CURRENT_OBJECT = {" +
                    "int x = "#5#" +
                    "int y = "#8#" +
          "}";
          java.awt.Point p = (java.awt.Point)lr.deserialize(_string,0);
          map.set_point(p);
      }
}
```

The string values are placed between delimiters. The default delimiter is "#". You can change the delimiter in the **Serialization** tab of the recording options. Delimiters are used to speed up the parsing of the string during replay.

When modifying the string, you must maintain the following rules:

- Order of lines may not be changed. The parser reads the values one-by-one—not the member names.
- Only values between two delimiters may be modified.
- Object references may not be modified. Object references are indicated only to maintain internal consistency.
- "_NULL_" can appear as a value, representing the Java null constant. You can replace it with string type values only.
- Objects may be deserialized anywhere in the script. For example, you can deserialize all objects in the **init** method and use the values in the **action** method.
- Maintain internal consistency for the objects. For example, if a member of a vector is **element count** and you add an element, you must modify the element count.

In the following segment, a vector contains two elements:

```
public class Actions {
  // Public function: init
      public int init() throws Throwable {
          _string = "java.util.Vector CURRENTOBJECT = {" +
                    "int capacityIncrement = "#0#" +
                    "int elementCount = #2#" +
                    "java/lang/Object elementData[] = {" +
                           "elementData[0] = #First Element#" +
                           "elementData[1] = #Second Element#" +
                           "elementData[2] = _NULL_" +
                            ....
                           "elementData[9] = _NULL_" +
                    "}" +
          "}";
          _vector = (java.util.Vector)lr.deserialize(_string,0);
          map.set_vector(_vector);
```

```
      }
}
```

In the following example, one of the vector's elements was changed—a "_NULL_" value was changed to "Third element". In coordination with the addition of the new element, the **elementCount** member was modified to **3**.

```
public class Actions {
  // Public function: init
      public int init() throws Throwable {
          _string = "java.util.Vector CURRENTOBJECT = {" +
                    "int capacityIncrement = "#0#" +
                    "int elementCount =  #3# " +
                    "java/lang/Object elementData[] = {" +
                           "elementData[0] = #First Element#" +
                           "elementData[1] = #Second Element#" +
                           "elementData[2] =  #Third Element#" +
                             ....
                           "elementData[9] = _NULL_" +
                    "}" +
          "}";
          _vector = (java.util.Vector)lr.deserialize(_string,0);
          map.set_vector(_vector);
      }
}
```

Due to the complexity of the serialization mechanism, which opens up the objects to ASCII representation, opening large objects while recording may increase the time required for script generation. To decrease this time, you can specify flags which will improve the performance of the serialization mechanism.

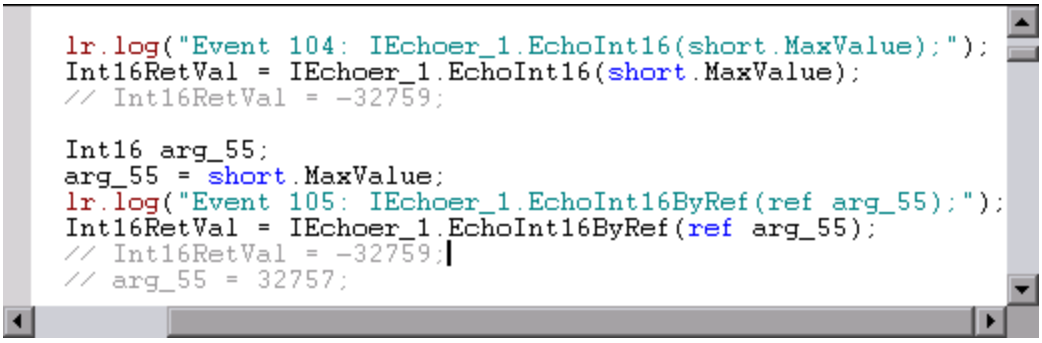
When adding **lr.deserialize** to your script, we recommend that you add it to the **init** method—not the **action** method. This will improve performance since VuGen will only deserialize the strings once. If it appears in the **action** method, VuGen will deserialize strings for every iteration.

## Microsoft .NET Correlation

This task describes how to correlate Microsoft .NET Vuser scripts.

For primitive values, you should generate the script with output parameter values and examine the output parameters for correlations.

1. Select **Recording > Recording Options**, and select the **General > Script** node.
2. Select the **Insert output parameter values** check box. Click **OK** to close the Recording Options dialog box.
3. Select **Record > Regenerate Script** to regenerate the script.
4. Search the commented output primitive values for correlations.




For more information about using correlation functions, see the Function Reference (**Help > Function Reference**).

## Oracle NCA Correlation

The following steps describe different items in Oracle NCA scripts that may need correlation.

### Correlate Statements for Load Balancing

VuGen supports load balancing for multiple application servers. You correlate the HTTP return values with the **nca_connect_server** parameters. The Vuser then connects to the relevant server during test execution, applying load balancing. The following steps describe how to correlate statements for load balancing.

1. **Record a multi-protocol script.**

   Record a multi-protocol script for Oracle NCA and Web Protocols. Perform the desired actions and save the script.

2. **Define parameters for host and host arguments.**

   Define two variables, **serverHost** and **serverArgs**, for parameterization:

   ```
   web_set_max_html_param_len("512");
   web_reg_save_param("serverHost", "NOTFOUND=ERROR",
       "LB=<PARAM name=\"serverHost\" value=\"","RB=\">", LAST);
   web_reg_save_param("serverArgs", "NOTFOUND=ERROR",
       "LB=<PARAM name=\"serverArgs\" value=\"","RB=\">", LAST);
   ```

3. **Assign values to serverHost and serverArgs:**

   ```
   web_url("step_name", "URL=http://server1.acme.com/test.htm", LAST);
   ```

4. **Modify the nca_connect_server statement from:**

   ```
   nca_connect_server("199.203.78.170", 9000"/*version=107*/,
   "module=e:\\appsnca...fndnam=apps ");
   ```

   to:

```
nca_connect_server("{ serverHost }", "9000"/*version=107*/, "
{serverArgs}");
```

The script should now look like this:

```
web_set_max_html_param_len("512");
web_reg_save_param("serverHost", "NOTFOUND=ERROR",
    "LB=<PARAM name=\"serverHost\" value=\"","RB=\">", LAST);
web_reg_save_param("serverArgs", "NOTFOUND=ERROR",
    "LB=<PARAM name=\"serverArgs\" value=\"","RB=\">", LAST);
    web_url("step_name", "URL=http://server1.acme/test.htm", LAST);
nca_connect_server("{serverHost}","9000"/*version=107*/,"{serverArgs}");
```

## Correlate the icx_ticket Variable

The icx_ticket variable, is part of the information sent in the **web_url** and **nca_connect_server** functions:

```
web_url("fnd_icx_launch.runforms",
"URL=http://ABC-123:8002/pls/VIS/fnd_icx_launch.runforms\?ICX_
TICKET=5843A55058947ED3=;RESP_APP=AR=;RESP_KEY=RECEIVABLES_MANAGER=;SECGRP_
KEY=STANDARD", LAST);
```

This **icx_ticket** value is different for each recording. It contains cookie information sent by the client. To correlate your recording, add **web_reg_save_param** before the first occurrence of the recorded **icx_ticket** value:

```
web_reg_save_param("icx_ticket", "LB=TICKET=", "RB==;RES", LAST);
...
web_url("fnd_icx_launch.runforms",
"URL=http://ABC-123:8002/pls/VIS/fnd_icx_launch.runforms\?ICX_TICKET={icx_ticket}=;RESP_
APP=AR=;RESP_KEY=RECEIVABLES_MANAGER=;SECGRP_KEY=STANDARD", LAST);
```

> **Note:** The left and right boundaries of **web_reg_save_param** may differ depending on your application setup.

### Correlate the JServSessionIdroot Values

The **JServSessionIdroot** value is a cookie that the application sets to store the session ID. In most cases, VuGen automatically correlates this value and inserts a **web_reg_save_param** function. If VuGen did not add this function automatically, you add it manually, replacing all of its occurrences with the parameter name.

To identify the value that you need to correlate, open the Execution log (**View > Output Window**) and locate the response body.

```
vuser_init.c(8):     Set-Cookie: JServSessionIdroot=my1sanw2n1.JS4; path=/\r\n
vuser_init.c(8):     Content-Length: 79\r\n
vuser_init.c(8):     Content-Type: text/plain\r\n
vuser_init.c(8):     \r\n
vuser_init.c(8):     81-byte response body for "http://ABC-
123/servlet/oracle.forms.servlet.ListenerServlet?ifcmd=getinfo=;
                          ifhost=mercury=;ifip=123.45.789.12" (RelFrameId=1)
vuser_init.c(8):
/servlet/oracle.forms.servlet.ListenerServlet?JServSessionIdroot=my1sanw2n1.JS4\r\n
```

To correlate this dynamic value, insert a **web_reg_save_param** function before the first occurrence and then replace the variable value with the parameter name throughout the script. In this example, the right and left boundaries are \r and \n, but you should check your specific environment to determine the exact boundaries in your environment.

```
web_reg_save_param("NCAJServSessionId","LB=\r\n\r\n","RB=\r","ORD=1",LAST);
web_url("f60servlet",
    "URL= http://ABC-
"123/servlet/oracle.forms.servlet.ListenerServlet?ifcmd=getinfo=;"
"ifhost=mercury=;ifip=123.45.789.12", LAST);
web_url("oracle.forms.servlet.ListenerSer",
    "URL=http://ABC-123{NCAJServSessionId}?ifcmd=getinfo=;"
"ifhost=mercury=;ifip=123.45.789.12", LAST);
```

## Siebel Correlation

The following steps describe how to correlate Siebel Web Vuser scripts.

### Correlation Library

To assist you with correlation, Siebel has released a correlation library file as part of the Siebel Application Server version 7.7. This library is available only through Siebel. The library file, **ssdtcorr.dll**, is located under the siebsrvr\bin folder for Windows and under siebsrvr/lib for Linux installations.

The library file, **ssdtcorr.dll**, must be available to all machines where a Load Generator or Controller reside. The following steps describe how to enable correlation with this library.

1. Copy the DLL file into the bin folder of the product installation.
2. Open a multi-protocol script using the **Siebel-Web** Vuser type.
3. Enable UTF-8 support in the **Recording Options > HTTP Properties > Advanced** node.
4. Open the recording option's Correlation node and click **Import**. Import the rules file, **WebSiebel77Correlation.cor**, from the \dat\webrulesdefaultsetting folder. If you are prompted with warnings, click **Override**.

   To revert to the default correlation, delete all of the Siebel rules and click **Use Defaults**.

When using the Siebel correlation library, verify that the SWE count rules (where the left boundary contains the **SWEC** string) are not disabled.

### Correlation Rules

VuGen's native built-in rules for the Siebel server detect the Siebel server variables and strings, automatically saving them for use at a later point within the script. The rules list the boundary criteria that are unique for Siebel server strings.

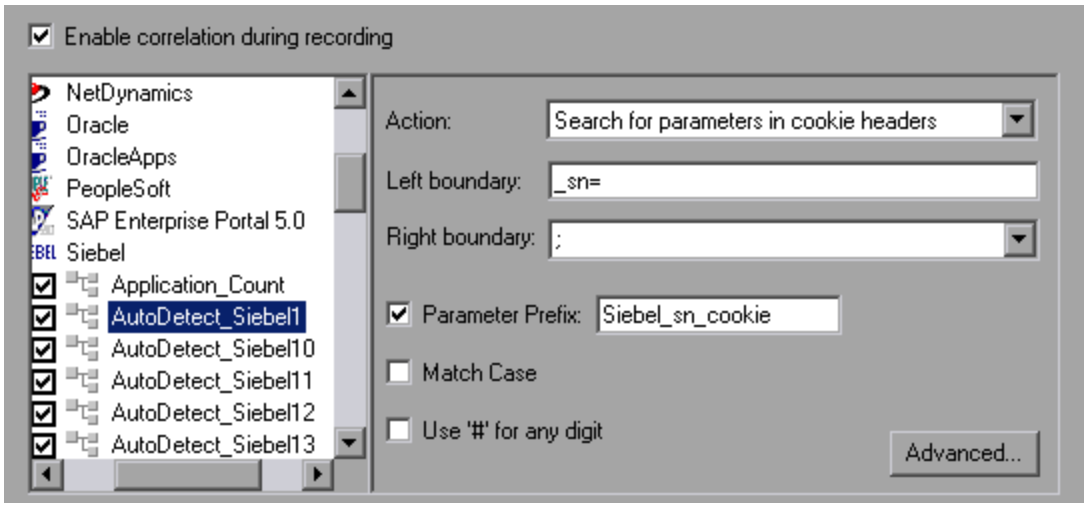
When VuGen detects a match using the boundary criteria, it saves the value between the boundaries to a parameter. The value can be a simple variable or a public function.

In normal situations, you do not need to disable any rules. In some cases, however, you may want to disable rules that do not apply. For example, disable Japanese content check rules when testing English-only applications.

Another reason to disable a rule is if the Controller explicitly requires an error condition to be generated. View the rule properties in the recording options and determine the conditions necessary for your application.

## Simple Variable Correlation

In the following example, the left boundary criteria is _sn=. For every instance of _sn= in the left boundary and ; in the right, VuGen creates a parameter with the **Siebel_sn_cookie** prefix.




In the following example, VuGen detected the _sn boundary. It saved the parameter to Siebel_sn_cookie6 and used it in the web_url function.

```
/* Registering parameter(s) from source
web_reg_save_param("Siebel_sn_cookie6",
"LB/IC=_sn=",
"RB/IC=;",
"Ord=1",
```

```
"Search=headers",
"RelFrameId=1",
LAST);
...
web_url("start.swe_3",
"URL=http://cannon.mylab.com/callcenter_
enu/start.swe?SWECmd=GotoPostedAction=;SWEDIC=true=;_sn={Siebel_sn_cookie6}=;SWEC=
{Siebel_SWECount}=;SWEFrame=top._sweclient=;SWECS=true",
"TargetFrame=",
"Resource=0",
"RecContentType=text/html",
"Referer=http://cannon.mylab.com/callcenter_enu/start.swe?SWECmd=GetCachedFrame=;_sn=
{Siebel_sn_cookie6}=;SWEC={Siebel_SWECount}=;SWEFrame=top._swe",
"Snapshot=t4.inf",
"Mode=HTML",
LAST);
```

## Function Correlation

In certain instances, the boundary match is a function. Functions generally use an array to store the runtime values. In order to correlate these values, VuGen parses the array and saves each argument to a separate parameter using the following format:

```
<parameter_name> = <recorded_value> (display_name)
```

The display name is the text that appears next to the value, in the Siebel Application.

VuGen inserts a comment block with all of the parameter definitions.

```
/* Registering parameter(s) from source task id 159
    // {Siebel_Star_Array_Op33_7} = ""
    // {Siebel_Star_Array_Op33_6} = "1-231"
    // {Siebel_Star_Array_Op33_2} = ""
    // {Siebel_Star_Array_Op33_8} = "Opportunity"
    // {Siebel_Star_Array_Op33_5} = "06/26/2003 19:55:23"
    // {Siebel_Star_Array_Op33_4} = "06/26/2003 19:55:23"
    // {Siebel_Star_Array_Op33_3} = ""
    // {Siebel_Star_Array_Op33_1} = "test camp"
    // {Siebel_Star_Array_Op33_9} = ""
    // {Siebel_Star_Array_Op33_rowid} = "1-6F"
    // */
```

In addition, when encountering a function, VuGen generates a new parameter for **web_reg_save_param**, **AutoCorrelationFunction**. VuGen also determines the prefix of the parameters and uses it as the parameter name. In the following example, the prefix is **Siebel_Star_Array_Op33**.

```
web_reg_save_param("Siebel_Star_Array_Op33",
```

```
    "LB/IC=`v`",
    "RB/IC=`",
    "Ord=1",
    "Search=Body",
    "RelFrameId=1",
    "AutoCorrelationFunction=flCorrelationCallbackParseStarArray",
    LAST);
```

VuGen uses the parameters at a later point within the script. In the following example, the parameter is called in **web_submit_data**.

**Example:** web_submit_data("start.swe_14", "Action=http://cannon.mylab.com/callcenter_enu/start.swe", "Method=POST", "RecContentType=text/html", "Referer=", "Snapshot=t15.inf", "Mode=HTML", ITEMDATA, "Name=SWECLK", "Value=1", ENDITEM, "Name=SWEField", "Value=s_2_1_13_0", ENDITEM, "Name=SWER", "Value=0", ENDITEM, "Name=SWESP", "Value=false", ENDITEM, "Name=s_2_2_29_0", "Value={**Siebel_Star_Array_Op33_1**}", ENDITEM, "Name=s_2_2_30_0", "Value={**Siebel_Star_Array_Op33_2**}", ENDITEM, "Name=s_2_2_36_0", "Value={**Siebel_Star_Array_Op33_3**}", ENDITEM, ...

During replay, Vusers do a callback to the public function, using the array elements that were saved as parameters.

**Note:** Correlation for the **SWEC** parameter is not done through the correlation rules. VuGen handles it automatically with a built-in detection mechanism. For more information, see below.

## SWEC Correlation

SWEC is a parameter used by Siebel servers representing the number of user clicks. The SWEC parameter usually appears as an argument of a URL or a POST statement. For example:

```
GET /callcenter_enu/start.swe?SWECmd=GetCachedFrame=;_sn=2-
mOTFXHWBAAGb5Xzv9Ls2Z45QvxGQnOnPVtX6vnfUU_=;SWEC=1=;SWEFrame=top._swe._sweapp
HTTP/1.1
```

or

```
POST /callcenter_enu/start.swe HTTP/1.1
...
\r\n\r\n
SWERPC=1=;SWEC=0=;_sn=2-mOTFXHWBAAGb5Xzv9Ls2Z45QvxGQnOnPVtX6vnfUU_
=;SWECmd=InvokeMethod...
```

VuGen handles the changes of the SWEC by incrementing a counter before each relevant step. VuGen stores the current value of the SWEC in a separate variable (Siebel_SWECount_var). Before each step, VuGen saves the counter's value to a VuGen parameter (Siebel_SWECount).

In the following example, web_submit_data uses the dynamic value of the SWEC parameter, Siebel_ SWECount.

```
Siebel_SWECount_var += 1;
lr_save_int(Siebel_SWECount_var, "Siebel_SWECount");
web_submit_data("start.swe_8",
    "Action=http://cannon.mylab.com/callcenter_enu/start.swe",
    "Method=POST",
    "TargetFrame=",
    "RecContentType=text/html",
    "Referer=",
    "Snapshot=t9.inf",
    "Mode=HTML",
    "EncodeAtSign=YES",
    ITEMDATA,
    "Name=SWERPC", "Value=1", ENDITEM,
    "Name=SWEC", "Value={Siebel_SWECount}", ENDITEM,
    "Name=SWECmd", "Value=InvokeMethod", ENDITEM,
    "Name=SWEService", "Value=SWE Command Manager", ENDITEM,
    "Name=SWEMethod", "Value=BatchCanInvoke", ENDITEM,
    "Name=SWEIPS",...
    LAST);
```

**Note:** The SWEC parameter may also appear in the referrer URL. However, its value in the referrer URL usually differs from its value in the requested URL. VuGen handles this automatically.

### Correlate SWECount Parameters

The SWECount parameter value is usually a small number consisting of one or two digits. It is often difficult to determine where to replace the recorded value with a parameter.

In the **web_submit_data** function, VuGen only replaces it in the SWEC field.

In URLs, VuGen only replaces the value when it appears after the strings "SWEC=" or "SWEC `".

The parameter name for all the SWECount correlations is the same.

### Correlate ROWID Parameters

In certain cases, the **rowid** is preceded by its length, encoded in hexadecimal format. Since this length can change, this value must be correlated.

For example, the following string is comprised of a length value and RowID, `xxx6_1-4ABCyyy`, where 6 is the length, and `1-4ABC` is the RowID.

If you define parameters to correlate the string as

```
xxx{rowid_Length}_{rowid}yyy
```

then using this enhanced correlation, VuGen generates the following function before the string:

```
web_save_param_length("rowid", LAST);
```

This function gets the value of **rowid**, and saves its length into the parameter **rowid_length** in hexadecimal format.

### Correlate SWET (timestamp) Parameters

The SWETS value in the script, is the number of milliseconds since midnight January 1st, 1970.

VuGen replaces all non-empty timestamps in the script, with the parameter {SiebelTimeStamp}. Before saving a value to this parameter, VuGen generates the following function:

```
web_save_timestamp_param("SiebelTimeStamp", LAST);
```

This function saves the current timestamp to the **SiebelTimeStamp** parameter.

## Web Manual Correlations

LoadRunner allows users to manually to add correlations that were missed by the automatic engine; or, alternatively, can be used by scripters that want to identify all correlations by themselves.

### Correlate selection

This method allows users to select the desired value (text string) in the script code (function arguments) and requests LoadRunner to check whether this value appears to be a correlation by activating the “Correlate selection” option from context menu (right click à correlate selection).
The logic used here is identical to the “Record-based correlations” mentioned above.

### Correlate from snapshot

The user should select a dynamic value from the Server response within a snapshot, and the engine will scan the entire script and replace all relevant occurrences.

### Double recording

This method is a design approach that directs the user to record the same business process twice, compare the scripts and locate any mismatches between the two as candidates for the correlation process that will be done manually.

As mentioned before, in order to correlate dynamic values, after locating the appropriate server response, we need to extract this data. The value can be extracted using different methods.

To do this, we use a set of API functions.

The main correlation APIs are:

| | |
|---|---|
| **web_reg_save_param_ex** | Registers a request to save dynamic data located between specified boundaries. |
| **web_reg_save_param_regexp** | Registers a request to save dynamic data that matches a regular expression. |
| **web_reg_save_param_xpath** | Registers a request to save dynamic data from XML response using XPath expression. |
| **web_reg_save_param_json** | Registers a request to save dynamic data from response that has been formatted as JSON. |

## Web Manual Correlation in Code

This task describes how to correlate web scripts manually by modifying the code.

1. Locate the string and its details

   Identify the statement that contains dynamic data and the patterns that characterize the locations of the data. These patterns may be boundaries or xpaths.

   a. **Identify Patterns using Boundaries**

   Use these guidelines to determine and set the boundaries of the dynamic data:

   - Analyze the location of the dynamic data within the HTTP response.
   - Identify the string that is immediately to the left of the dynamic data. This string defines the left boundary of the dynamic data.
   - Identify the string that is immediately to the right of the dynamic data. This string defines the right boundary of the dynamic data.
   - The right and left boundaries should be as unique as possible to better locate the strings.
   - **web_reg_save_param_ex** looks for the characters between (but not including) the specified boundaries and saves the information beginning one byte after the left boundary and ending one byte before the right boundary. **web_reg_save_param_ex** does not support embedded boundary characters.

     For example, if the input buffer is `{a{b{c}` and "{" is specified as a left boundary, and "}" as a right boundary, the first instance is c and there are no further instances—it found the right and left boundaries but it does not allow embedded boundaries, so "c" is the only valid match.

   By default, the maximum length of any boundary string is 256 characters. Include a **web_set_max_html_param_len** function in your script to increase the maximum permitted length. For example, the following function increases the maximum length to 1024 characters:

   These length restrictions do not apply where either the left or right boundaries are blank.

   b. **Identify Patterns using XPath or JSON Path expressions**

   Use the snapshot pane to manually search for the XPath of the desired string.

By default, the maximum length of any boundary string is 256 characters Include a **web_set_max_html_param_len** function in your script to increase the maximum permitted length. For example, the following function increases the maximum length to 1024 characters:

These length restrictions do not apply where either the left or right boundaries are blank.

2. Add **web_reg_save_param_*** function

   Add a **web_reg_save_param_*** function into the script before the statement that contains the dynamic data.

| | |
|---|---|
| **web_reg_save_param_ex** | This function searches server responses in web steps for the left boundary following by the string and the right boundary and saves the string to a parameter named in the function's argument. After finding the specified number of occurrences, **web_reg_save_param_ex** does not search any more responses. For more information, see the Function Reference (**Help > Function Reference**). |
| **web_reg_save_param_xpath** | This function searches server responses in web steps for a specified XPath. The string located in specified XPath is saved to a parameter named in the function's argument. For more information, see the Function Reference (**Help > Function Reference**). |
| **web_reg_save_param_json** | This function searches server responses in Web steps for a specified JSONPath. The located string is saved to a parameter named in the function's argument. For more information, see the Function Reference and Internet resources for JSONPath, XPath for JSON. |

3. Replace data with parameter

   Select **Edit** > **Replace** from the VuGen main window to display the Search and Replace dialog box. Search the entire script for the dynamic data and replace it with a parameter. Give the parameter any name and enclose it with braces: `{param_name}`. You can include a maximum of 64 parameters per script.

## Web-based Correlation Using Design Studio

This topic describes how to use the Correlation tab of the Design Studio to correlate Vuser scripts.

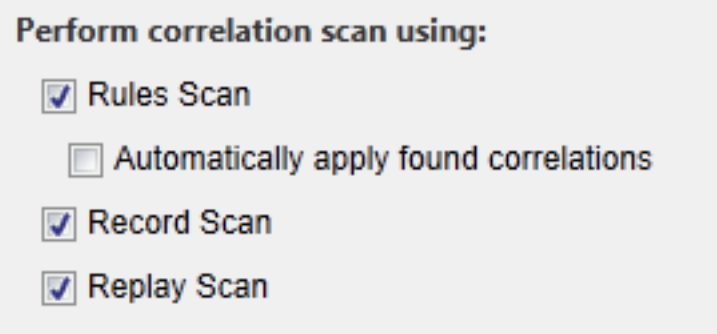
### Prerequisites

1. Record a script using one of the following protocols:
   - Web HTTP/HTML
   - Flex
   - RTMP/RTMPT
   - Citrix
   - SAP - Web

- Oracle NCA

> **Note:** You can only correlate the Web HTTP/HTML steps within your Oracle NCA script and the Web HTTP/HTML protocol must be active. To activate, select **Recording Options > Protocol > Active Protocols > Web HTTP/HTML**.
>
> For additional information on manual correlation, see "Oracle NCA Correlation" on page 282

2. Verify that **Record > Recording Options > Correlations > Configuration** has all scan types enabled.




## Using the Correlation tab

1. Click the **Design Studio** button to perform an initial scan. This will open the Design Studio dialog box which will scan for response (or record based) correlations and apply correlation rules. The progress bar in the dialog box indicates if the initial scan was successful.

   For details on the Correlation tab, see "Correlating Values" on page 254.

   > **Note:** If no dynamic values for correlation are found, a warning is displayed, advising you to check your recording options. Verify that **Record > Recording Options > Correlations > Configuration** has all scan types enabled.

2. Select which values to correlate by highlighting the value in the grid and clicking the **Correlate** button.

   When a value is correlated, VuGen adds a **web_reg_save_param_*** function, and saves the original value in a comment in the script.

   You can examine the details of the correlation by expanding the Details Chevron in the dialog box. For details, see "Correlating Values" on page 254.

   For details on the Correlation tab, see "Correlating Values" on page 254.

3. Click the **Add as Rule** button to add a rule type . In addition, you can click the **Edit rule** button to view and edit the corresponding rule if the dynamic value was correlated by a rule. For details, see "Correlations > Rules Recording Options" on page 171.

4. Click the **Replay and Scan** button. The Correlation tab progress bar indicates if additional correlations have been found. Again, you can select which values you would like to correlate by highlighting the value in the grid and clicking the **Correlate** button. You may need to repeat this step several times.
5. When Design Studio no longer finds new correlations, the progress bar will display **Design Studio Scan Complete**.

> **Tip:** If Design Studio does not resolve all correlation-based errors, try to resolve them using manual correlation. For details, see "Manually Correlate Scripts" on page 260.

## Winsock Correlation

VuGen's Design Studio provides a user interface for correlating Vuser scripts. Correlation is required when working with dynamic data. A common issue with Winsock Vuser scripts is dynamic ports—ports whose numbers are assigned dynamically. While certain applications always use the same port, others use the next available port. If you try to replay a script and the recorded port is no longer available, your script will fail to replay. To overcome this issue, you must perform correlation—save the actual runtime values and use them within the script.

VuGen uses **lrs_save_param** and **lrs_save_searched_string** functions correlate Winsock scripts. This means that it stores the data that is received for use in a later point within the script. Since correlation stores the received data, it applies only to Receive buffers and not to Send buffers. The recommended procedure is to select a string of dynamic data within the Receive buffer that you want to correlate. Use that same parameter in a subsequent Send buffer.

### Correlating a Winsock script

You use the Snapshot pane to begin correlating Winsock Vuser scripts. Both the Text and the Hex tabs in the Snapshot pane have the correlating functionality.

1. In the Snapshot pane, select the data that you want to correlate.
2. Right-click in the selection, and select **Create Correlation** or **Create Boundary Correlation**. The Design Studio opens and displays the Correlation tab.

   Note that you can click the number of occurrences in the **Replace/Found** column to enable you to choose the exact occurrences that you want to correlate.
3. Click the **Details** bar to display details about the correlation.
4. Make sure that the **Original Snapshot Step** tab is visible. Notice that the type is either **Data Range** or **Boundary Based**.
5. Click **Correlate** to perform the correlation of the Vuser script.
6. Click **Close** to close the Design Studio. Notice that VuGen has inserted the appropriate correlation functions and comments into the script.

For further details on how to use the Design Studio, see "Correlating Values" on page 254.

### Parameterization versus Correlation

This type of correlation should not be confused with simple parameterization. Simple parameterization (**Design > Parameters > Create New Parameter**) applies only to data within Send buffers. You set up a parameter and assign it several values. VuGen uses the different values in each of the script runs or iterations.

For details on how to manually correlate a Winsock Vuser script, see "Winsock Manual Correlation" below.

## Winsock Manual Correlation

This topic describes how to use the Editor to manually correlate values in a Winsock Vuser script. It is recommended that you first try to correlate values through the Design Studio. For details, see "Winsock Correlation" on the previous page.

In the following example, a user performed a telnet session. The user used a ps command to determine the process ID (PID), and killed an application based on that PID.

**Example:**

```
frodo:/u/user1>ps
   PID TTY      TIME CMD
14602 pts/18    0:00 clock
14569 pts/18    0:03 tcsh
frodo:/u/user1>kill 14602
[3]    Exit 1                clock
frodo:/u/user1>
```

During execution, the PID of the procedure is different, so killing the recorded PID will be ineffective. To overcome this problem, use **lrs_save_param_ex** to save the current PID to a parameter. Replace the constant PID value with the parameter.

**To manually correlate the value:**

1. View the buffer contents by selecting **data.ws** in the **Extra File** node of the script. Locate the data that you want to replace with the contents of the saved buffer. Use the right-click menu item **Replace with Parameter**, to replace all instances of the value with the parameter.
2. In the **data.ws** file, note the buffer in which the data was received, for example buf47.

**Example:**

```
recv buf47 98
    "\r"
    "\x00"
    "\r\n"
    "   PID TTY      TIME CMD\r\n"
    " 14602 pts/18   0:00 clock\r\n"
    " 14569 pts/18   0:02 tcsh\r\n"
```

```
    "frodo:/u/lab>"
.
.
.
send buf58
    "kill 14602"
```

3. Determine the offset (starting point) and length of the data string to save. In the above example, the offset of the **PID** is 11 and its length is 5 bytes. For additional information about displaying the data, see "Data Buffers" on page 866.
4. In the recorded section, determine the socket used by buf47. In this example, buf47 used socket1.

```
lrs_receive("socket1", "buf47", LrsLastArg);
```

5. Using the Steps Toolbox, insert an **lrs_save_param_ex** function in the recorded section, after the **lrs_receive** for the relevant buffer. In our example, the buffer is **buf47**. Use the parameter name that you used in Step 1.

```
lrs_save_param_ex("socket1", "user", buf47, 11, 5, ascii, param1);
```

6. Print the parameter to the output using **lr_output_message**.

```
lr_output_message ("param1: %s", lr_eval_string("<param1>"));
```

# Replaying

The **Replaying** section describes the various methods that are available to replay Vuser scripts.

## Replay Overview

Developing a Vuser script includes the steps shown below. This topic provides an overview of the fourth step, replaying a Vuser script.

After recording a Vuser script, you use VuGen to replay the script. This helps to test the basic functionality of the Vuser script and to uncover errors and issues that need to be addressed. For example, you may find the need for correlation, which is a typical issue that is revealed when you first replay a script.

When the replay is successful, you are ready to enhance the script by adding load-testing functionality to the script. Such functionality could include parameterization, transactions, and rendezvous points.

> **Note:** 64-bit replay is not supported for Java over HTTP protocol, and for Java Record Replay or Java Vuser protocols when using web packages.

### See also:

- "Correlation Overview" on page 251
- "Replay a Vuser Script" below
- "Debugging Overview" on page 328
- "Files Generated During Replay" on page 325
- "Bookmarks Overview" on page 323 to learn how to use bookmarks to navigate between sections of the script
- "Run a Vuser Script from a Command Prompt" on page 323
- "Run a Vuser Script from a Linux Command Line" on page 887

## Replay a Vuser Script

This task describes how to replay a Vuser script.

1. Configure the runtime settings and replay options
   a. Runtime settings control how your Vuser script is replayed. Access the desired runtime settings by double-clicking the **Runtime Settings** node in the **Solution Explorer**.

      For an overview of runtime settings, see "Runtime Settings" on page 301.

   b. Specify replay options by selecting **Tools > Options**. For details on options, see "Options Dialog Box" on page 109.
2. Replay the script

   To run a Vuser script until the end of the script or until the next breakpoint, perform one of the following:

   - Select **Replay > Run**.
   - Click the **Run** button on the Vuser toolbar.
   - Press **F5**.

   > **Note:** The status of the Vuser script execution appears in the lower left corner of VuGen. The script execution status may be **Ready**, **Running**, or **Paused**.

- To stop a script that is running, click the **Stop Replay** button on the VuGen toolbar.
- To pause a script that is running, click the **Pause** button on the VuGen toolbar.
- To continue running a script that is paused, click the **Continue** button on the VuGen toolbar.

3. View the logs for detailed information

   You can view detailed information about how your script behaved during the replay. This information appears in the Output window. For details, see "Output Pane" on page 100.

### See also:

- "Replay Overview" on page 295.

## Check Linux Compatibility

VuGen provides a tool to check the compatibility of your script to run on Linux-based load generators. You can use this tool to check Linux compatibility while developing the script in VuGen, and so avoid errors and issues later.

To use the tool, open the script in VuGen, then select **Replay > Test <script name> for Linux Compatibility**.

The tool first checks if the script protocol is supported on Linux; if it is, the tool runs script validation and displays one of the following results:

- **Success** - Linux compatibility test finished with no issues.
- **Warning** - the script might not run correctly on Linux.
- **Error** - the script is blocked, unable to run at all on Linux.

The warning and error messages include a list of found issues.

To see which protocols are supported to run on Linux, fully or partially, see the System Requirements (check for protocols that run on any supported operating system−−this includes Linux systems).

> **Note:** The BinaryXML DFE extension is not supported on Linux. However, the Linux compatibility test may not give a warning if a DFE API call that uses BinaryXml DFE extension was manually added to the script.

## Work with Snapshots

This topic describes how to use the basic Snapshot pane functionality.

### Show the Snapshot pane

Do one of the following:

- Select **View > Snapshot**.
- Click the **Snapshot** button on the VuGen toolbar.
- In the Editor, click inside a step that contains a reference to a snapshot.
- In the Step Navigator, double-click a step that contains a reference to a snapshot. Note that in the Step Navigator, each step that contains a snapshot displays a Snapshot icon . You can place your mouse cursor over the snapshot icon to see a thumbnail view of the snapshot.

  For more details on the Snapshot pane, see "Snapshot Pane" on page 85.

## Copy a snapshot to the clipboard

1. Display the snapshot in the Snapshot pane.
2. Right-click on the snapshot, and then select **Copy Image to the Clipboard**.

> **Note:** The "copy snapshot to the clipboard" functionality is available for only RDP, Citrix, and SAP Vuser scripts.

## Copy snapshot text to the clipboard

1. Display the snapshot in the Snapshot pane.
2. Select the text that you want to copy.
3. Right-click in the selected text, and select **Copy Selection**.

## Activate the snapshot-on-error functionality

1. Click **Replay > Runtime Settings**. The runtime settings dialog box opens.
2. Under **General**, click **Miscellaneous**.
3. Under **Error Handling**, select the **Generate snapshot on error** check box.

## Set the snapshot options

1. Click **Tools > Options**. The Options dialog box opens.
2. Click **Scripting**, and then click **Snapshot**. The snapshot options appear on the right of the dialog box.

## Troubleshooting Snapshots

If you encounter a step without a snapshot, follow these guidelines to determine why it is not available. Note that not all steps are associated with snapshots—only steps with screen operations or for Web, showing browser window content, have snapshots.

Several protocols allow you to disable the capturing of snapshots during recording using the Recording options.

If there is no **Record** snapshot displayed for the selected step, it may be due to one of the following:

- The script was recorded with a VuGen version 6.02 or earlier.
- Snapshots are not generated for certain types of steps.
- The imported actions do not contain snapshots.

If there is no **Replay** snapshot displayed for the selected step, it may be due to one of the following:

- The script was recorded with VuGen version 6.02 or earlier.
- The imported actions do not contain snapshots.
- The Vuser files are stored in a read-only folder, and VuGen could not save the replay snapshots.
- The step represents navigation to a resource.

### See also:

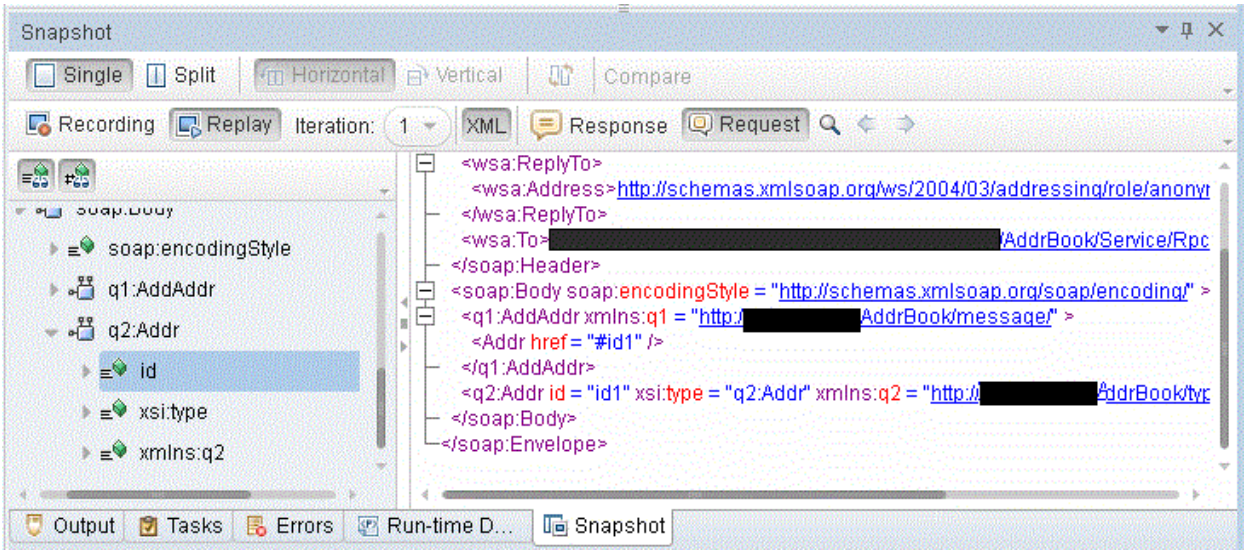
- "Snapshot Pane" on page 85

## Snapshots that Have an XML View

VuGen's Snapshot pane shows various snapshots that were recorded while a Vuser script was recorded or replayed. For specific Vuser protocols only, the Snapshot pane can show XML views of the snapshot. The XML or JSON data is displayed in the XML View tab.

The XML View tab only appears within the **Response** and **Request** Body tabs. The XML view includes its own set of controls and functionality.

The XML view is divided into two areas. On the left is the tree view of the snapshot data, and on the right is the text view.




The two XML views are synchronized. If you select an entry in the tree view, VuGen highlights the corresponding element, attribute, or value in the text view. Alternatively, if you double-click an element,

attribute, or value in the text view, VuGen opens the tree view as required, and highlights the corresponding entry.

The splitter controls ( ) between the tree view and text view enable you to set the proportion of the available space that is occupied by each of the views.

Controls to the left of the text view enable you to expand and collapse elements within the text view. Click the right-facing arrow to expand an element in the view, and click the down-facing arrow to collapse an element in the view. Note that you can place the mouse cursor inside an element in the text view and then press the <+> key to expand the element or the <-> key to collapse the element, as appropriate.

After recording a Vuser script, you can use the XML View in the Snapshot pane to add a text check to the script. For details, see "Add a Text Check From the XML View in the Snapshot Pane" below.

## Add a Text Check From the XML View in the Snapshot Pane

After recording a Vuser script, you can add a text check from the XML view in the Snapshot pane.

1. Click **View > Snapshot**, or click the **Show Snapshot Pane** button on the VuGen toolbar.
2. In the Snapshot pane, display a snapshot that contains the text that you want to verify.
3. On the right-side of the Snapshot pane, click the **XML View** tab.
4. In the Snapshot pane, click the **Response Body** tab.
5. In either the Tree view or the Grid view, locate and select the text string that you want to verify.
6. Right-click inside the selection, and select **Add Text Check Step**. The Find Text dialog box opens.
7. Modify the options in the Find Text dialog box. For details on the dialog box options, press F1 when in the dialog box to open the Function Reference.
8. Click **OK** to insert a **web_reg_find** step into the Vuser script.

### See also:

- "Snapshots that Have an XML View" on the previous page

## Running a Vuser as a Process or Thread

The Controller uses a driver program (such as *mdrv.exe* or *r3vuser.exe*) to run your Vusers.

If you run each Vuser as a **process**, then the same driver program is launched (and loaded) into the memory again and again for every instance of the Vuser. Loading the same driver program into memory uses up large amounts of RAM (random access memory) and other system resources. This limits the numbers of Vusers that can be run on any load generator.

Alternatively, if you run each Vuser as a **thread**, the Controller launches only one instance of the driver program (such as *mdrv.exe*), for every 50 Vusers (by default). This driver process/program launches several Vusers, each Vuser running as a thread. These threaded Vusers share segments of the memory

of the parent driver process. This eliminates the need for multiple re-loading of the driver program/process saves much memory space, thereby enabling more Vusers to be run on a single load generator.

To configure these options, open the runtime settings (F4) and select the **General > Miscellaneous** node.

## Runtime Settings

Runtime settings define the way a Vuser script runs, and can be configured before or after you record a script. These settings are stored in files located in the Vuser script folder. Runtime settings are applied to Vusers when you run a script using VuGen, the Controller, Performance Center, or Business Process Monitor.

Configuring runtime settings allows you to emulate different kinds of user activity. For example, you can emulate a user who responds immediately to output from the server, or a user who stops and thinks before each response. You can also configure the runtime settings to specify how many times the Vuser should repeat its set of actions.

Different combinations of runtime settings are listed depending on the protocol your are working with. For more information, see "Configuring Runtime Settings" below.

You can export runtime settings to a JSON file and import them into another script, instead of having to set them repeatedly for each script. For more information, see "Import and Export Runtime Settings" on page 319.

### Configuring Runtime Settings

This topic describes how to accessing and validating the available runtime settings.

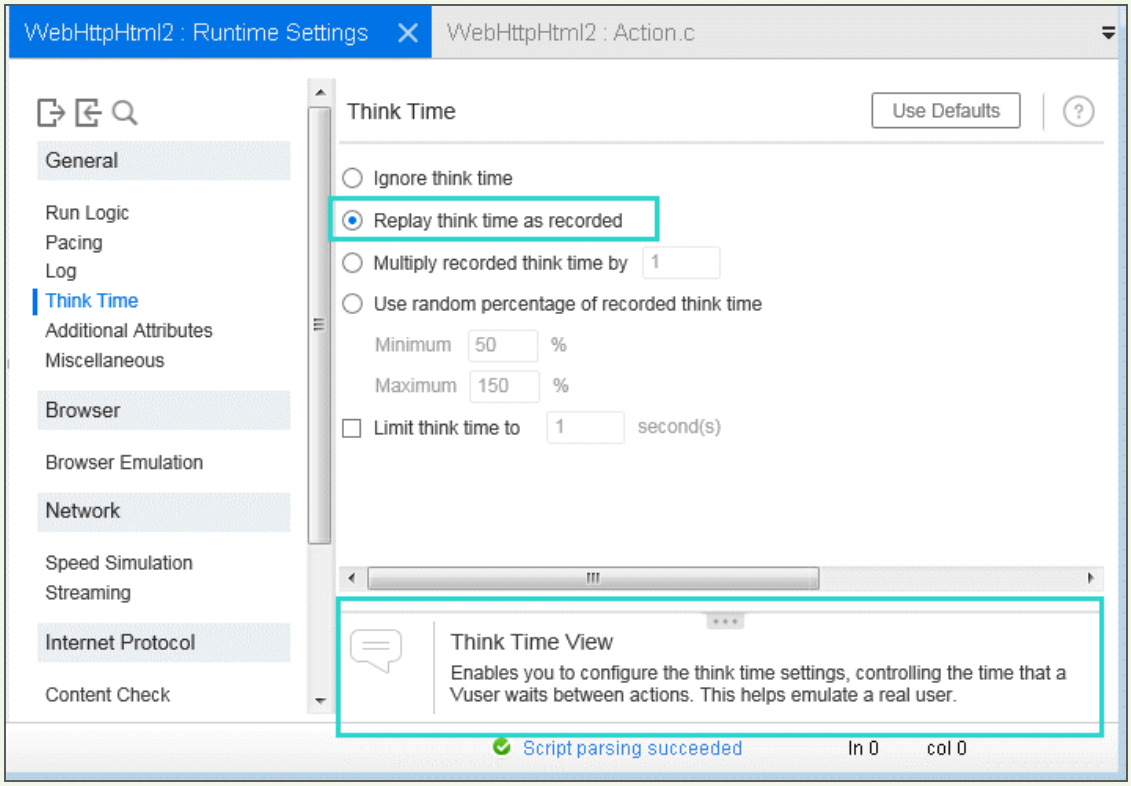
#### Access runtime settings

**To open the runtime settings do one of the following:**

- In the Solution Explorer, double-click **Runtime Settings**
- Select the menu item **Replay > Runtime Settings**
- Press the shortcut key, F4.

> **Note:**
>
> - Internet Explorer 10 (or later versions) must be installed to display the runtime settings.
> - For an overview of the runtime settings, see "Runtime setting descriptions" on page 303. Descriptions of the individual runtime settings are available within the runtime settings views,

by hovering the mouse over a runtime setting field name.




## Search the runtime settings

Use the Search feature to search for a runtime setting by its name.

To search for a runtime setting:

1. In the Runtime Settings pane, click the **Search** button.
2. In the search bar that opens, enter a search string.
   The search string is highlighted in each instance it was found.

   **Note:** Search results display only views containing settings with names that match the search criteria. Other views are filtered out.

3. Click the up and down arrows to browse through the results.
4. Click to clear the search.

Runtime setting descriptions

| View | Description |
|---|---|
| .NET Environment | Enables you to set the runtime settings for .NET Vuser scripts. |
| Additional Attributes | Enables you to provide additional arguments for a Vuser script. The Additional Attributes settings apply to all Vuser script types. You specify command line arguments that you can retrieve at a later point during the test run, using Command Line Parsing functions. Using this view, you can pass external parameters to prepared scripts. |
| Browser Emulation | Enables you to configure the browser related runtime settings. |
| Browser (TruClient - Web) | Enables you to configure settings for the TruClient browsers, for scripts that you run in load mode.<br>**Additional information:**<br>• Settings that you modify in this view only affect TruClient Vusers in load mode.<br>• These settings correspond to those in the **Browser Settings** tab in the TruClient General Settings dialog box. However, the settings that you modify in the TruClient General Settings dialog box affect interactive mode only. For more details, see the TruClient Help Center (select the relevant version).<br>• When you save your script in interactive mode, the settings that you modified in the **Browser Settings** tab are applied to these **Load** runtime settings. |
| Citrix Configuration | Enables you to set the Citrix configuration runtime settings. |
| Citrix Synchronization | Enables you to set the Citrix synchronization runtime settings. |
| Client Emulation | Enables you to set the Oracle NCA runtime settings. |

| View | Description |
|---|---|
| Content Check | Enables you to check websites for content during runtime. You can set Content Check runtime settings for Web - HTTP/HTML and other Internet protocols.<br><br>**Additional information:**<br><br>You use the Content Check settings to check the contents of a page for a specific string. This is useful for detecting non-standard errors. In normal operations, when your application server fails, the browser displays a generic HTTP error page indicating the nature of the error. The standard error pages are recognized by VuGen and treated as errors, causing the script to fail. Some application servers, however, issue their own error pages that are not detected by VuGen as error pages. The page is sent by the server and it contains a formatted text string, stating that an error occurred.<br><br>For example, suppose that your application issues a custom page when an error occurs, containing the text **ASP Error**. You instruct VuGen to look for this text on all returned pages. When VuGen detects this string, it fails the replay.<br><br>**Note:** VuGen searches the body of the pages—not the headers. |
| DFE Chain Configuration | Enables Data Format Extensions during replay. |
| Download Filters | Enables you to set the download filters for a script. |
| Flex Configuration | Enables you to set an external JVM (Java Virtual Machine) path and other runtime settings. |
| Flex Externalizable | Enables you to configure runtime setting for externalizable objects in Flex scripts. |
| Flex RTMP | Enables you to set the Flex RTMP runtime settings. |
| Java Classpath | Enables you to specify the location of additional classes that were not included in the system's classpath environment variable. You may need these classes to run Java applications and ensure proper replay. |
| Java VM | Enables you to set the Java VM runtime settings. |
| JMS Advanced | Enables you to set the JMS advanced runtime settings. |
| Log | Enables you to configure the amount and types of information that are recorded in the log. |

| View | Description |
|---|---|
| Log (TruClient) | Enables you to configure the amount and types of information that are reported to a log for TruClient scripts. |
| Miscellaneous | Enables you to set miscellaneous runtime settings.<br>Automatic transactions are not relevant for Application Performance Management.<br>**Tips:**<br>• It is not recommended to enable both the **Continue on Error** and **Generate Snapshot on Error** options in a load test environment. This configuration may adversely affect Vuser performance.<br>• If you require the Vusers to generate breakdown data for diagnostics (J2EE) during the test run, do not use automatic transactions. Instead, manually define the beginning and end of each transaction.<br>• For considerations on whether to run a Vuser as a process or thread, see "Running a Vuser as a Process or Thread" on page 300. |
| Mobile Device | Enables you to select mobile device properties when recording a TruClient - Mobile Web script. |
| MQTT Configuration | Enables you to configure various MQTT runtime settings, some of which are used as the default values for arguments in the script. |
| Other Settings (TruClient) | Enables you to configure snapshot generation and action on error for TruClient. |
| Pacing | Enables you to control the time between iterations. The pace tells the Vuser how long to wait between iterations of your actions. |
| Preferences | Enables you to set various Internet-related runtime settings.<br>For information about the Internet Preferences runtime settings, see "Preferences View - Internet Protocol" on page 309. |

| View | Description |
|---|---|
| Proxy | Enables you to set the proxy server connection settings.<br>If you select the option to use the default HTTP proxy settings:<br>• When running on Windows, the Internet Explorer proxy settings are used.<br>To use this option, your default browser cannot be Firefox. If it is, uninstall Firefox and then select another default browser.<br>• When running on Linux, the proxy values in the HTTP_PROXY or HTTPS_PROXY environment variables are used.<br>You must manually create these environment variables on the Linux computer where your script runs. |
| RDP Advanced | Enables you to set the RDP advanced runtime settings.<br>**Tip:** To conserve system resources on the remote desktop server. disable the options that are not essential for your test. |
| RDP Agent | Enables you to set the RDP Agent runtime settings.<br>**Tip:** For the **RDP agent log folder** option—if no folder is specified and the agent log destination was set to **File**, the log is saved in the temp folder of the user on the server. |
| RDP Configuration | Enables you to set the RDP configuration runtime settings. |
| RDP Synchronization | Enables you to set the RDP synchronization runtime settings. |
| Replay (TruClient) | Enables you to set the runtime settings for replay of TruClient scripts. |
| RTE | Enables you to set the RTE runtime settings.<br>You can use the **TE_typing_style** function to override the Delay settings for a portion of a Vuser script.<br>**Tip:** In the **Delay before typing** option, the delay settings determine how Vusers execute **TE_type** functions.<br>• **First key.** Specifies the time (in milliseconds) that a Vuser waits before entering the first character of a string.<br>• **Subsequent keys.** Specifies the time (in milliseconds) that a Vuser waits between submitting successive characters. |

| View | Description |
|---|---|
| Run Logic | Enables you to set the run logic runtime settings.<br><br>**Note:** When defining **random** run logic, the blocks or actions in a blocks are run with a **probability** of the percentages defined and not exactly as defined.<br><br>In the following example, during the 10 iterations, Block0 and Block1 will run with a **probability** of 20% and 80% respectively, and not exactly 2 and 8 times.<br> |
| SAP GUI > General | Enables you to set the SAP GUI runtime settings.<br><br>**Performance settings:**<br><br>• **Show SAP Client during replay.** This option shows an animation of the actions in the SAP client during replay. The benefit of this, is that you can closely follow the actions of the Vuser and see how the forms are filled out. This option, however, requires additional resources and may affect the performance of your load test.<br>• **Create snapshots during replay.** Captures both ActiveScreen snapshots and regular snapshots while a script runs.<br><br>ActiveScreen snapshots contain control ID information for all active objects. ActiveScreen snapshots differ from regular snapshots in that ActiveScreen snapshots allow you to see which objects were recognized by VuGen in the SAP GUI client. As you move your mouse across an ActiveScreen snapshot, VuGen highlights the detected objects. You can then add new steps to the script directly from within the snapshot. ActiveScreen snapshots also enable you to add steps interactively from within the snapshot for a specific object. For more information, see "Enhance SAP GUI Scripts" on page 687.<br><br>**Note:** Disabling replay snapshots may improve the script replay speed and save storage space. |

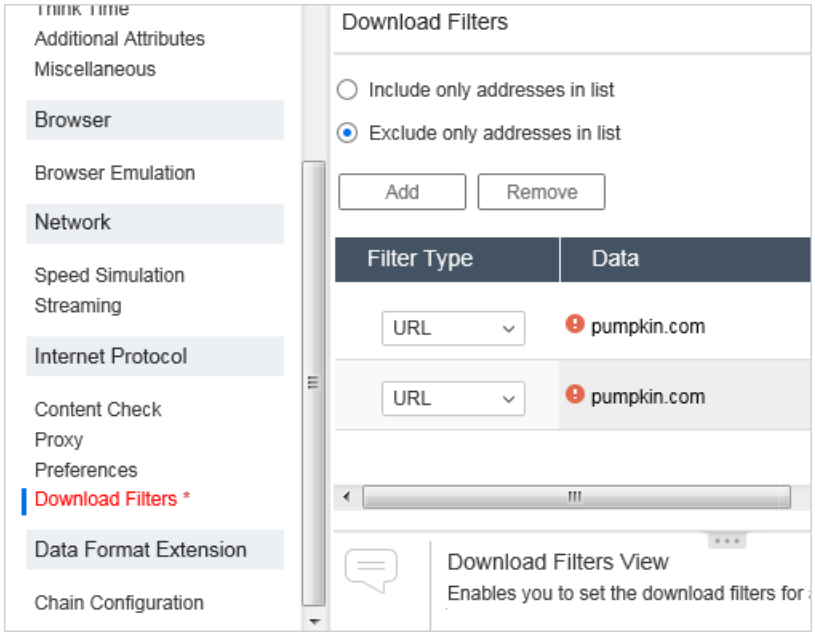
| View | Description |
|---|---|
| Server (TruClient Native Mobile) | Enables you to specify the server from which you want to collect data and the credentials. |
| Shared DLLs | Enables you to modify the list of shared DLLs after you record a Vuser script. If a DLL is included in the list of shared DLLs, when the Vuser script is run and requires a particular DLL, the Vuser will access the DLL in its shared location – the DLL will not be copied to the load generator. Adding a DLL to the list of shared DLLs therefore saves hard-drive space on the load generator when a Vuser runs. |
| Silverlight Services | Enables you to view the WSDL files associated with your script and modify their settings for the replay phase. |
| Speed Simulation | Enables you to configure bandwidth runtime settings. |
| Speed Simulation(TruClient) | Enables you to configure bandwidth for the TruClient Web and Mobile Web protocols. |
| Think Time | Enables you to configure the think time settings, controlling the time that a Vuser waits between actions. These settings are designed to help you emulate a real user. |

### Runtime setting value validation

The Solution Explorer indicates that one or more runtime settings have illegal values, with a warning icon in place of the standard Runtime Setting node icon .

If you enter a value above the maximum allowed value, VuGen automatically substitutes it with the maximum allowed value. If you enter a value below the minimum allowed value, VuGen automatically substitutes it with the minimum allowed value.

A red highlighted value indicates that one of the values in the view is invalid. The following example is invalid because the added filters are identical.




### Revert runtime settings to the default settings

1. In the Solution Explorer pane, select the runtime settings node of the script to be changed and navigate to the required setting.
2. Click the **Use Defaults** button.
3. Save the script. Only the defaults for the displayed node are changed. If you want to revert to the default settings for all the runtime settings, you must repeat the above steps for each runtime setting node.

## Preferences View - Internet Protocol

The Preferences view runtime settings (**Replay > Runtime Settings > Internet Protocol > Preferences**) enable you to set various Internet-related options.

This view is available only for specific protocols. When you open the runtime settings, only the relevant views are displayed.

For general information about runtime settings, see "Runtime Settings" on page 301.

### Checks

| UI Element | Description |
|---|---|
| **Enable image and text checks** | Allows the Vuser to perform verification checks during replay by executing the verification functions **web_find** or **web_image_check**. This option only applies to statements recorded in HTML-based mode. Vusers running with verification checks use more memory than Vusers who do not perform checks.<br><br>**Default value:** Disabled |

### Web Performance Graph Generation

| UI Element | Description |
|---|---|
| **Hits per Second**<br>**Pages per Second**<br>**Response Bytes per Second** | Instructs a Vuser to collect data for Web Performance graphs.<br>Select the types of graph data for the Vuser to collect to view the (Throughput) graphs during test execution using the online monitors and after test execution using the Analysis. You view the Component Breakdown graph after test execution using the Analysis.<br>**Note:** If you do not use the Web performance graphs, disable these options to conserve memory. |

### Advanced

| UI Element | Description |
|---|---|
| **Use WinInet replay instead of Sockets (Windows only)** | Instructs VuGen to use the WinInet replay engine instead of the standard Sockets replay.<br>VuGen has two HTTP replay engines: Sockets-based (default) or WinInet based.<br>The WinInet is the engine used by Internet Explorer and it supports all of the features incorporated into the IE browser. The limitations of the WinInet replay engine are that it is not scalable and does not support Linux. In addition, when working with threads, the WinInet engine does not accurately emulate the modem speed and number of connections.<br>VuGen's proprietary sockets-based replay is a lighter engine that is scalable for load testing. It is also accurate when working with threads. The limitation of the sockets-based engine is that it does not support SOCKS proxy. If you are recording in that type of environment, use the WinInet replay engine.<br>**Supported protocols:** Flex, Mobile Application - HTTP/HTML, Oracle - Web, Web - HTTP/HTML, SAP - Web, Siebel - Web<br>**Default value:** disabled (socket-based replay engine). |
| **Include File name and line in automatic transaction names.** | Creates unique transaction names for automatic transactions by adding file name and line number to the transaction name. |

| UI Element | Description |
|---|---|
| **List non-critical resource errors as warnings** | Returns warnings for actions that fail during test replay when the actions are performed on non-critical resources, thereby enabling the test to replay successfully.<br><br>For details on how a resource is classified as critical or non-critical, see "Defining Non-Critical Resources" on page 318.<br><br>If you want the failure of non-critical resources to be errors and thereby fail your test, you can disable this option. This option is enabled by default.<br><br>You can set a content type to be critical by adding it to the list of Non-Resources. For more information, see "Non-Resources Dialog Box" on page 199. |
| **Save snapshot resources locally** | Saves the snapshot resources to files on the local machine. |
| **Enable Dynatrace monitoring** | Enables LoadRunner to push data to the Dynatrace monitor on the AUT server during the scenario run.<br><br>When enabled, the x-dynaTrace header required by Dynatrace is added to each HTTP request.<br><br>**Note:**<br>• Step names that exceed 64 characters are trimmed to 64 characters.<br>• When a script runs on a Linux load generator, the transaction name is not displayed on the Dynatrace monitor. |

HTTP

| UI Element | Description |
|---|---|
| **HTTP version** | Specifies which version HTTP to use: version 1.0 or 1.1. This information is included in the HTTP request header whenever a Vuser sends a request to a Web server.<br><br>HTTP 1.1 supports the following features:<br>• Persistent Connections—see "Keep-Alive HTTP connections" below.<br>• HTML compression—see Accept Server-Side Compression below.<br>• Virtual Hosting—multiple domain names sharing the same IP address. |
| **Enable HTTP/2 features** | Enables you enable or disable HTTP/2 features. |

| UI Element | Description |
|---|---|
| **Keep-Alive HTTP connections** | Keep-alive is a term used for an HTTP extension that allows persistent or continuous connections. These long-lived HTTP sessions allow multiple requests to be sent over the same TCP connection. This improves the performance of the Web server and clients.<br><br>The keep-alive option works only with Web servers that support keep-alive connections. This setting specifies that all Vusers that run the Vuser script have keep-alive HTTP connections enabled.<br><br>**Default value:** Enabled |
| **Include Accept-Language request header** | Provides a comma-separated list of accepted languages. For example, **en-us**, **fr**, and so forth. For more details, see "Page Request Header Language" on page 900. |
| **Mark HTTP errors as warnings** | Issues a warning instead of an error upon failing to download resources due to an HTTP error. |
| **HTTP-request connect timeout (sec)** | The time, in seconds, that a Vuser will wait for the connection of a specific HTTP request within a step before aborting. Timeouts provide an opportunity for the server to stabilize and respond to the user.<br><br>**Maximum value:** 32000 seconds |
| **HTTP-request receive timeout (sec)** | The time, in seconds, that a Vuser will wait to receive the response of a specific HTTP request within a step before aborting. Timeouts provide an opportunity for the server to stabilize and respond to the user.<br><br>**Maximum value:** 32000 seconds |
| **HTTP Keep-Alive timeout (sec)** | A time limit within which some activity must be performed on an HTTP connection. If this timeout is reached, the connections is closed during replay. |
| **Request zlib headers** | Sends request data to the server with the **zlib** compression library headers. By default, requests sent to the server include the **zlib** headers. This option lets you emulate non-browser applications that do not include **zlib** headers in their requests.<br><br>**Default value:** Enabled |

| UI Element | Description |
|---|---|
| **Accept server-side compression** | Indicate to the server that the replay can accept compressed data. The available options are: **None** (no compression), **gzip** (accept gzip compression), **gzip, deflate** (accept gzip or deflate compression), and **deflate** (accept deflate compression). Note that by accepting compressed data, you may significantly increase the CPU consumption.<br><br>Default value: Accept **gzip** and **deflate** compression.<br><br>To manually add compression, enter the following function at the beginning of the script:<br><br>`web_add_auto_header("Accept-Encoding", "gzip");`<br><br>To verify that the server sent compressed data, search for the string Content - Encoding: gzip in the section of the server's responses of the replay log. The log also shows the data size before and after decompression. |
| **Delete unreferenced cache entries** | Delete cache entries that have not been referenced within the specified number of iterations. Set to zero (0) to never delete cache entries. |

General

| UI Element | Description |
|---|---|
| **Create snapshots during replay** | Create snapshots during replay.<br><br>**Note:** Disabling replay snapshots will improve the replay speed. However, snapshot-dependent features such as DFE and correlations, will not be able to use data captured during the replay. This may cause unstable behavior. |
| **DNS caching** | Instructs the Vuser to save a host's IP addresses to a cache after resolving its value from the Domain Name Server. This saves time in subsequent calls to the same server. In situations where the IP address changes, as with certain load balancing techniques, be sure to disable this option to prevent Vuser from using the value in the cache.<br><br>**Default value:** Enabled |
| **Convert to/from UTF-8** | Converts received HTML pages and submitted data from and to UTF-8. You enable UTF-8 support in the recording options. For more information, see "Recording Options" on page 164.<br><br>**Default value:** No |

| UI Element | Description |
|---|---|
| **Charset to use for converting HTML** | The character set to use to convert received HTMLs and submitted data from/to the set charset.<br>This option is ignored if you enabled the previous option, \'Convert to/from UTF-8\'. |
| **Mark step timeouts caused by resources as a warning** | Issues a warning instead of an error when a timeout occurs due to a resource that did not load within the timeout interval. For non-resources, VuGen issues an error.<br>**Default value:** Disabled |
| **Parse HTML content-type** | When expecting HTML, parse the response only when it is the specified content-type: **HTML**, **text\html**, **TEXT** any text, or **ANY**, any content-type. Note that text/xml is not parsed as HTML.<br>**Default value:** TEXT |
| **Step download timeout (sec)** | The time that the Vuser will wait before aborting a step in the script. This option can be used to emulate a user behavior of not waiting for more than x seconds for a page.<br>**Maximum value:** 32000 seconds<br>The timeout settings are primarily for advanced users who have determined that acceptable timeout values should be different for their environment. The default settings should be sufficient in most cases. If the server does not respond in a reasonable amount of time, check for other connection-related issues, rather than setting a very long timeout which could cause the scripts to wait unnecessarily. |
| **Network buffer size** | Sets the maximum size of the buffer used to receive the HTTP response.<br>If the size of the data is larger than the specified size, the server will send the data in chunks, increasing the overhead of the system.<br>When running multiple Vusers from the Controller, every Vuser uses its own network buffer.<br>This setting is primarily for advanced users who have determined that the network buffer size may affect their script's performance.<br>The default is 12K bytes.<br>The maximum size is 0x7FFF FFFF. |
| **Print NTLM information** | Print information about the NTLM handshake to the standard log. |
| **Print SSL information** | Print information about the SSL handshake to the standard log. |

| UI Element | Description |
|---|---|
| **SSL version** | The version of SSL used by your application. |
| **Maximum number of failure-matches to list as errors** | Limit the number of content-check failures that are issued as errors, where a failure is indicated by the appearance of a string (Fail=Found). This applies to match criteria using a left and right boundary. All subsequent matches are listed as informational messages.<br>**Default value:** 10 matches |
| **Maximum redirection depth** | The maximum number of allowed redirections.<br>**Default value:** 10 |
| **Maximum number of 'META Refresh' on a single page** | The maximum number of times that a META refresh can be performed per page.<br>**Default value:** 2 |
| **Consider ContentCheck values as UTF-8 encoded** | Store the values in the ContentCheck XML file in UTF-8.<br>**Default value:** Disabled |
| **Limit the Tree view request body to** | Limit the number of request body bytes displayed in Tree-View. Set to zero (0) for no limit. |
| **IP version** | The IP version to be used: IPv4, IPv6 or automatic selection. The default value is IPv4. |
| **web_sync retry interval** | The time to wait (in milliseconds) between testing the condition that yields false and the next retry.<br>**Default value:** 1000 |
| **web_sync retry timeout** | The maximum time (in milliseconds) during which retries are allowed. If the computed timeout exceeds the step timeout (as determined by the 'Step download timeout' setting), the latter is used. |
| **WebSocket callback interval** | The time interval in milliseconds, before repeating a call to a WebSocket callback handler. This must be a non-zero value. |

| UI Element | Description |
|---|---|
| **Prefetch and prerender callback timer interval** | The time interval in milliseconds, before repeating a call to Prefetch and Prerender callback handlers. This must be a non-zero value. |

Authentication

| UI Element | Description |
|---|---|
| **Add a fixed delay upon authentication** | Automatically adds think time to the Vuser script for emulating a user entering authentication information (username and password). This think time will be included in the transaction time.<br>**Default value:** 0 |
| **Disable NTLM2 session security** | Use full NTLM 2 handshake security instead of the more basic NTLM 2 session security response.<br>**Default value:** No |
| **Use the native Windows NTLM implementation** | Use the Microsoft Security API for NTLM authentication instead of the indigenous one.<br>**Default value:** No |
| **Override credentials in a Windows native NTML implementation** | Use the credentials provided by the user at logon. |
| **Enable integrated authentication** | Enable Kerberos-based authentication. When the server proposes authentication schemes, use **Negotiate** in preference to other schemes.<br>**Default value:** No |
| **Induce heavy KDC load** | Do not reuse credentials obtained in previous iterations. Enabling this setting will increase the load on the KDC (Key Distribution Server). To lower the load on the server, set this option to **Yes** in order to reuse the credentials obtained in previous iterations.<br>This option is only relevant when Kerberos authentication is used.<br>**Default value:** No |

| UI Element | Description |
|---|---|
| **Use canonical name in SPN** | Use the canonical name instead of the original hostname retrieved from the URL, to generate SPN (Service Principal Name).<br>**Default value:** Yes |
| **Append non-default port to SPN** | Append the port number to the SPN, if the specified port is a non-standard one (neither 80 nor 443).<br>**Default value:** No |
| **Enable retrieving keys from nCipher HSM** | Enables Vusers to retrieve private keys from the nCipher HSM (Hardware Security Module). This option loads and initializes the CHIL engine necessary to retrieve these keys.<br>**Default value:** Yes |

Logging

| UI Element | Description |
|---|---|
| **Print buffer line length** | Line length for printing request/response header/body and/or JavaScript source, disabling wrapping. |
| **Print buffer escape for binary zeros only** | • **Yes.** Escape only binary zeros when printing request/response headers/body and/or JavaScript source.<br>• **No**. Escape any unprintable/control characters. |
| **Limit the maximum response size written to the log** | Limits the size of the log containing the response data. |

JavaScript

| UI Element | Description |
|---|---|
| **Enable JavaScript debugging mode** | **Only visible for Web-based Vuser Scripts generated in the JavaScript language.** Enables debugging for the replay of Vuser scripts. This only applies to the replay in VuGen—not the Controller. Enabling this option may impact replay performance. |
| **Enable running JavaScript code** | **Only visible for Vuser Scripts generated in the C language.** Enables the replay of Web JavaScript steps, such as **web_js_run()** and **web_js_reset()**. This option creates a JavaScript runtime engine even if there are no JavaScript steps in the script. |

| UI Element | Description |
|---|---|
| **JavaScript Engine runtime size** | **Only visible for Vuser Scripts generated in the C language.**The memory size in kilobytes, to allocate for the JavaScript engine runtime. One runtime engine will be created for all Vusers in a process.<br><br>**Default:** 51200 KB |
| **JavaScript Engine stack size per thread** | **Only visible for Vuser Scripts generated in the C language.**The memory size in kilobytes, to allocate for each Vuser thread in the JavaScript engine.<br><br>**Default:** 32 KB |

### See also:

- "Runtime Settings" on page 301
- "Configuring Runtime Settings" on page 301

## Defining Non-Critical Resources

You can define resources used in your test to be non-critical, so that the test can replay successfully even when actions performed on these resources fail. For example, an image or a Java applet that failed to download may not be critical to your business process.

### Prevent unnecessary failures due to errors with non-critical resources

- In the Runtime Settings Preferences view, the field List non-critical resource errors as warnings must be enabled (default value). See "Preferences View - Internet Protocol" on page 309.
- You classify a resource as non-critical, by defining the value of the Resource attribute in the function, as described below.

### Resource classification

The classification of a resource as critical or non-critical depends on the value of the **Resource** attribute in the function, as described below.

If **Resource** is specified:

- Resource = 0: The resource is always critical.
- Resource = 1: The resource is always non-critical.

If **Resource** is not specified:

- If Method is not Get: The resource is always critical.
- If Method is Get:

| RecContentType is omitted | RecContentType is text/html | RecContentType is not text/html |
|---|---|---|
| Critical | Critical | Not critical |

### Example

The resource in the following function is non-critical because **Resource** is 1:

```
web_url("webcode.exe",
      "URL=https://Example/asp/monitora.asp",
      "Resource=1",
      "RecContentType=gif",
      "Referer=",
      "Snapshot=t1.inf",
      LAST);
```

## Import and Export Runtime Settings

This topic describes how to import and export runtime settings. This functionality is available for all Vuser protocols.

### Export runtime settings

1. Open a script and double-click **Runtime Settings** in the Solution Explorer.
2. In the Runtime Settings view, click the **Export All** button.
3. In the Save As dialog box, choose a location for the JSON file that will store the runtime settings and click **Save**.

### Import runtime settings

1. Open a script and double-click **Runtime Settings** in the Solution Explorer.
2. In the Runtime Settings view, click the **Import** button.
3. In the Open dialog box, select a JSON file containing the runtime settings that you exported earlier and click **Open**.
4. Save the script.

### See also:

- "Runtime Settings" on page 301

## Configure Runtime Settings Manually

To configure Vuser runtime settings manually , you modify the *default.cfg* and *default.usp* files created with the script. The *default.cfg* file contains the setting for the General, Think Time, and Log options. The *default.usp* file contains the setting for the Run Logic and Pacing.

These runtime settings correspond to VuGen's runtime settings (see "Configuring Runtime Settings" on page 301).

### General Options

There is one General option for Linux Vuser scripts:

**ContinueOnError** instructs the Vuser to continue when an error occurs. To activate the option, specify 1. To disable the option, specify 0.

In the following example, the Vuser will continue on an error.

```
[General]
ContinueOnError=1
```

### Think Time Options

You can set the think time options to control how the Vuser uses think time during script execution. You set the parameters Options, Factor, LimitFlag, and Limit parameters according to the following chart.

| Option | Options | Factor | LimitFlag | Limit |
|---|---|---|---|---|
| Ignore think time | NOTHINK | N/A | N/A | N/A |
| Use recorded think time | RECORDED | 1.000 | N/A | N/A |
| Multiply the recorded think time by... | MULTIPLY | number | N/A | N/A |
| Use random percentage of recorded think time | RANDOM | range | lowest percentage | upper percentage |
| Limit the recorded think time to... | RECORDED/ MULTIPLY | number (for MULTIPLY) | 1 | value in seconds |

To limit the think time used during execution, set the `LimitFlag` variable to 1 and specify the think time `Limit`, in seconds.

In the following example, the settings tell the Vuser to multiply the recorded think time by a random percentage, ranging from 50 to 150.

```
[ThinkTime]
```

```
Options=RANDOM
Factor=1
LimitFlag=0
Limit=0
ThinkTimeRandomLow=50
ThinkTimeRandomHigh=150
```

### Log Options

You can set the log options to create a brief or detailed log file for the script's execution.

```
[Log]
LogOptions=LogBrief
MsgClassData=0
MsgClassParameters=0
MsgClassFull=0
```

You set the parameters LogOptions, MsGClassData, MsgClassParameters, and MsgClassFull variables according to the following chart:

| Logging Type | LogOptions | MsgClassData | MsgClassParameters | MsgClassFull |
|---|---|---|---|---|
| Disable Logging | LogDisabled | N/A | N/A | N/A |
| Standard Log | LogBrief | N/A | N/A | N/A |
| Parameter Substitution (only) | LogExtended | 0 | 1 | 0 |
| Data Returned by Server (only) | LogExtended | 1 | 0 | 0 |
| Advanced Trace (only) | LogExtended | 0 | 0 | 1 |
| All | LogExtended | 1 | 1 | 1 |

In the following example, the settings tell the Vuser to log all data returned by the server and the parameters used for substitution.

```
[Log]
LogOptions=LogExtended
MsgClassData=1
MsgClassParameters=1
```

```
MsgClassFull=0
```

### Iterations and Run Logic

You can set the Iteration options to perform multiple iterations and control the pacing between the iterations. You can also manually set the order of the actions and their weight. To modify the run logic and iteration properties of a script, you must edit the *default.usp* file.

To instruct the Vuser to perform multiple iterations of the Actions section, set RunLogicNumOfIterations to the appropriate value.

To control the pacing between the iterations, set the RunLogicPaceType variable and its related values, according to the following chart:

| Pacing | RunLogicPaceType | Related Variables |
|---|---|---|
| As soon as possible | ASAP | N/A |
| Wait between Iterations for a set time | Const | RunLogicPaceConstTime |
| Wait between iterations a random time | Random | RunLogicRandomPaceMin, RunLogicRandomPaceMax |
| Wait after each iteration a set time | ConstAfter | RunLogicPaceConstAfterTime |
| Wait after each iteration a random time | After | RunLogicAfterPaceMin, RunLogicAfterPaceMax |

In the following example, the settings tell the Vuser to perform four iterations, while waiting a random number of seconds between iterations. The range of the random number is from 60 to 90 seconds.

```
[RunLogicRunRoot]
MercIniTreeFather=""
MercIniTreeSectionName="RunLogicRunRoot"
RunLogicRunMode="Random"
RunLogicActionOrder="Action,Action2,Action3"
RunLogicPaceType="Random"
RunLogicRandomPaceMax="90.000"
RunLogicPaceConstTime="40.000"
RunLogicObjectKind="Group"
RunLogicAfterPaceMin="50.000"
Name="Run"
RunLogicNumOfIterations="4"
RunLogicActionType="VuserRun"
RunLogicAfterPaceMax="70.000"
RunLogicRandomPaceMin="60.000"
```

```
MercIniTreeSons="Action,Action2,Action3"
RunLogicPaceConstAfterTime="30.000"
```

## Bookmarks Overview

When you edit a Vuser script, you can use bookmarks to navigate between sections of the script. When you add a bookmark, a bookmark icon is added to the left of the selected line in your script.

The Bookmarks pane displays a list of all bookmarks that exist in the Vuser script. Using the Bookmarks pane, you can:

- Navigate to the location of the bookmark in your script.
- Navigate between consecutive bookmarks in the pane.
- Delete an individual bookmark.
- Clear all bookmarks.

### See also:

- "Use Bookmarks" on the next page

## Run a Vuser Script from a Command Prompt

This task describes how to replay a Vuser script on a machine from a command prompt or from the Windows Run dialog box—without the VuGen user interface.

To send command line parameters to a Vuser from within VuGen, add the attributes and their values in the Runtime Settings dialog box. For details, see the **General > Additional Attributes** view.

When you run a script from the command line, VuGen replays it in its basic form. The command line replay will not capture anything related to the UI, since it is absent in this type of run. This excludes the capturing of snapshots and similar runtime settings.

To run a script from a command line or the Run dialog box:

1. Open a **Command Prompt** window, or select **Start** > **Run** to open the Run dialog box.
2. Type **mdrv** followed by the script name, using the following syntax:

   ```
   <installation_dir>/bin/mdrv.exe -usr <script_name>
   ```

   where **script_name** is the full path to the *.usr* script file, for example, **c:\temp\mytest\mytest.usr.**
3. Add other command line options and arguments.
4. Click **Enter**. The **mdrv** program runs a single instance of the script without the user interface. The output files provide the runtime information.

> **Note:** The Linux command line utility, *run_db_vuser*, does not yet support many of the standard

> Windows command line options. For details, see "Run a Vuser Script from a Linux Command Line" on page 887.

The following examples provide common usages of a command line expression:

- You can specify the load generator, as well as indicate the number of times to run the script as indicated by the following example:

```
script1  -host pc4  -loop 5
```

- Specify a location for the output files. For example:

```
-out c:\tmp\vuser
```

- Specify arguments to pass to your script by using the following format:

```
-usr script_name  -arg_name arg_value  -arg_name arg_value
```

  For example, in the following command:

```
<installation_dir>/bin/mdrv.exe -usr myfile.usr -username Lisa
```

  you can use **lr_get_attrib_string("username")** in the script to retreive the value **Lisa**

You can retrieve the command line values by parsing the command line during replay, using the parsing functions, such as **lr_get_attrib_double**. For details, see the Function Reference (**Help > Function Reference**).

> **Tip:** To further customize your run, set the runtime settings for your script in the configuration files. For details, see "Configure Runtime Settings Manually" on page 320.

## Use Bookmarks

When working in the Editor, VuGen lets you place bookmarks at various locations within your script. You can navigate between the bookmarks to analyze and debug your code. The following steps describe how to work with bookmarks. Most of the bookmark functionality is available from VuGen's Bookmarks pane. To access the Bookmarks pane, click **View > Bookmarks**.

### Create a Bookmark

In the Editor, place the cursor at the desired location and press Ctrl + F2. VuGen places a bookmark icon in the left margin of the script.

### Remove a Bookmark

To remove a bookmark, perform one of the following:

- In the Editor, click in the line that contains the bookmark and press Ctrl + F2.
- In the Bookmark pane, select the bookmark that you want to delete and click the **Delete Bookmark** button .

VuGen removes the bookmark icon from the left margin.

### Navigate between Bookmarks

Click **View > Bookmarks** to display the Bookmarks pane.

- To move to the next bookmark, click the **Next Bookmark** button or press **F2**.
- To return to the previous bookmark, click the **Previous Bookmark** button or press **Shift + F2**.

You can navigate between bookmarks in the current action only. To navigate to a bookmark in another action, select that action in the left pane and then press F2.

### Navigate to a Specific Bookmark in a Vuser Script

In the Bookmarks pane, double-click the specific bookmark to which you want to navigate. The cursor flashes in the Editor at the start of the line containing the bookmark.

## Files Generated During Replay

This section describes what occurs when a Vuser script is replayed, and describes the files that are created.

1. The **options.txt** file is created. This file includes command line parameters to the preprocessor.

   **Example of options.txt file**

   ```
   -DCCI
   -D_IDA_XL
   -DWINNT
   -Ic:\tmp\Vuser                  (name and location of Vuser include files)
   -IE:\LRUN45B2\include           (name and location of include files)
   -ec:\tmp\Vuser\logfile.log      (name and location of output logfile)
    c:\tmp\Vuser\VUSER.c           (name and location of file to be processed)
   ```

2. The file **Vuser.c** is created. This file contains 'includes' to all the relevant .c and .h files.

   **Example of Vuser.c file**

```
#include "E:\LRUN45B2\include\lrun.h"
#include "c:\tmp\web\init.c"
#include "c:\tmp\web\run.c"
#include "c:\tmp\web\end.c"
```

3. The c preprocessor **cpp.exe** is invoked in order to 'fill in' any macro definitions, precompiler directives, and so on, from the development files.

   The following command line is used:

   ```
   cpp -foptions.txt
   ```

4. The file **pre_cci.c** is created which is also a C file (**pre_cci.c** is defined in the **options.txt** file). The file **logfile.log** (also defined in **options.txt**) is created containing any output of this process. This file should be empty if there are no problems with the preprocessing stage. If the file is not empty then it is almost certain that the next stage of compilation will fail due to a fatal error.
5. The **cci.exe** C compiler is now invoked to create a platform-dependent pseudo-binary file (.ci) to be used by the Vuser driver program that will interpret it at runtime. The cci takes the **pre_cci.c** file as input.
6. The file **pre_cci.ci** is created as follows:

   ```
   cci -errout c:\tmp\Vuser\logfile.log -c pre_cci.
   ```

7. The file **logfile.log** is the log file containing output of the compilation.
8. The file **pre_cci.ci** is now renamed to **Vuser.ci**.

   Since the compilation can contain both warnings and errors, and since the driver does not know the results of this process, the driver first checks if there are entries in the **logfile.log** file. If there are, it then checks if the file **Vuser.ci** has been built. If the file size is not zero, it means that the cci has succeeded to compile - if not, then the compilation has failed and an error message will be given.
9. The relevant driver is now run, taking both the **.usr** file and the **Vuser.ci** file as input. For example:

   ```
   mdrv.exe -usr c:\tmp\Vuser\.usr -out c:\tmp\Vuser -file
   c:\tmp\Vuser\Vuser.ci
   ```

   The **.usr** file is needed since it tells the driver program which database is being used. This determines which libraries need to be loaded for the run.
10. If there is an existing replay log file, **output.txt**, (see the following entry), the log file is copied to **output.bak**.
11. The **output.txt** file is created (in the path defined by the 'out' variable). This file contains the output messages that were generated during the script replay. These are the same messages that appear in the Replay view of VuGen's Output pane.

# NV Insights Report

This section describes how to view an NV Insights Report in VuGen after running a Vuser script.

> **Note:** The report is available for the following Vuser protocols only: Web HTTP/HTML, TruClient Web, Flex, SAP Web, and Siebel Web.

The NV Insights Report assists in pinpointing factors that negatively impact an application's performance across a network. The data in an NV Insights Report is derived from packet list data; the report displays the resulting data in an informative report that provides insight into an application's operation. The NV Insights Report consists of several sub-reports, each displaying different aspects of the network data captured during the replay of the Vuser script.

The report includes performance optimization recommendations and an HTTP analysis and resources breakdown, as well as load times, component download analysis, response time breakdown, and details of errors received.

> **Note:** VuGen saves an HTML version of the NV Insights Report for the most recent replay in the following location: <script's folder>\result1\NVReport\EmbeddedReport.html

For details on the NV Insights Report, see the Network Virtualization for LoadRunner & Performance Center Help.

## Opening the NV Insights Report

1. Make sure the NV Insights Report is enabled.

   > **Note:** To enable/disable the NV Insights Report, in VuGen click **Tools > Options > Scripting > Replay**, and select/clear the **Display NV Insights Report** check box. This option is enabled by default when the NV for Load Generator and VuGen component is installed on the VuGen machine. For details, see "Display NV Insights Report" on page 122.
   >
   > For details on how to install the required Network Virtualization component, see the Network Virtualization for LoadRunner & Performance Center Help.

2. After a script has finished running, open the NV Insights Report by clicking the **Open NV Insights Report** link in VuGen's Replay Summary page (displayed after script replay).

> **Note:** Generating the NV Insights Report may take time. Click **Cancel** (in VuGen's Replay Summary page) to abort report generation.

### Limitations

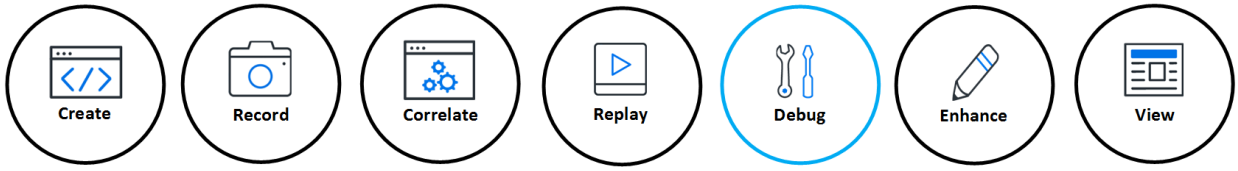
- The NV Insights Report is not supported for WinINet replay mode (set for a Web Vuser in **Replay > Runtime Settings > Internet Protocol > Preferences > Advanced**). In some instances, it is possible to modify a script that requires WinINet to run, to enable the script to run without WinINet. Contact customer support for more information.
- The NV Insights Report cannot be created if several instances of VuGen are running at the same time.
- The NV Insights Report supports only IPv4 network traffic—not IPv6.
- The NV engine permits the following characters in transaction names: a-z|A-Z|0-9|.|,|#|_|-|'
  Any other characters in transaction names are replaced by the character <-> in the NV Insights Report. (The original transaction name will still appear in the LoadRunner Analysis reports.)
- In order to distinguish transactions with the same name, during the merge process the NV engine drops all characters in the transaction name that appear after three consecutive underscores ("___"). Do not use three consecutive underscores in transaction names.

# Debugging

The **Debugging** section describes the various methods that are available to debug Vuser scripts.

## Debugging Overview

Developing a Vuser script includes the steps shown below. This topic provides an overview of the fifth step, debugging a Vuser script.




After creating a Vuser script, replay the script to verify that the script runs without errors. Using VuGen's debug features, you can identify and resolve errors in your scripts. You can access most of these script debugging features from the VuGen toolbar.

### Running a Vuser script

To run a Vuser script until the end of the script or until the next breakpoint, perform one of the following:

- Select **Replay > Run**.
- Click the **Run** button on the Vuser toolbar.
- Press **F5**.

> **Note:** The status of the Vuser script execution appears in the lower left corner of VuGen. The script execution status may be **Ready**, **Running**, or **Paused**.

- To stop a script that is running, click the **Stop Replay** button on the VuGen toolbar.
- To pause a script that is running, click the **Pause** button on the VuGen toolbar.
- To continue running a script that is paused, click the **Continue** button on the VuGen toolbar.

## The Run Step by Step Command

The **Run Step by Step** command runs the script one line at a time. This enables you to follow the script execution. The **Run Step by Step** command starts the script replay, and then and pauses it on the first line of the script, usually in the vuser_init() action.

To run the script step by step, perform one of the following:

- Select **Replay > Run Step by Step**.
- Click the **Run Step by Step** button on the VuGen toolbar.
- Press **F10**.

> **Note:** The **Run Step by Step** button is available only while a script is being replayed.

## Breakpoints

Breakpoints pause script execution at specified points in the script. This enables you to examine the effects of the script on your application at pre-determined points during script execution.

- For concept details on breakpoints, see "Working with Breakpoints" on page 332.
- For task details, see "Debug Scripts with Breakpoints" on page 335.

## Bookmarks

When working in Script view, VuGen lets you place bookmarks at various locations within your script. You can navigate between the bookmarks to help analyze and debug your code.

- For task details, see "Use Bookmarks" on page 324.

## Watching Variables

The Watch pane enables you to monitor variables and expressions while a script runs. You can monitor variables and expressions only when execution of a Vuser script is in the Paused state. To display the

Watch pane, click **View > Debug > Watch**. For details on using the Watch pane, see "Watching Expressions and Variables" on page 333.

### Go To Commands

- To navigate around a script using breakpoints, you can use the **Go To Source** command. For details, see "Debug Scripts with Breakpoints" on page 335.
- To navigate around a script using bookmarks, you can use the **Next Bookmark** and **Previous Bookmark** commands. For details, see "Use Bookmarks" on page 324.

If you want to examine the Replay log messages for a specific step or function, right-click the step in the Editor and select **Go To Step in Replay Log.** VuGen places the cursor at the start of the corresponding step in the Output pane's Replay log.

### Output Pane

The Output pane displays messages that were generated during the replay of your script. For details, see "Output Pane" on page 100.

### Correlation

To enable some recorded Vuser scripts to replay correctly, it may be necessary to implement correlation. Correlation is used when a recorded script includes a dynamic value (such as a session ID) and therefore cannot be successfully replayed. To resolve this, you convert the dynamic value into a variable—thereby enabling your script to replay successfully. For details, see "Correlation Overview" on page 251.

## Error Handling

You can specify how a Vuser handles errors during script execution. By default, when a Vuser detects an error, the Vuser stops executing the script. You can instruct a Vuser to continue with the next iteration when an error occurs using one of the following methods:

- Using runtime settings. You can specify the **Continue on Error** runtime setting. The **Continue on Error** runtime setting applies to the entire Vuser script. You can use the **lr_continue_on_error** function to override the **Continue on Error** runtime setting for a portion of a script.
- Using the **lr_continue_on_error** function. The **lr_continue_on_error** function enables you to control error handling for a specific segment of a Vuser script. To mark the segment, enclose it with `lr_continue_on_error(1);` and `lr_continue_on_error(0);` statements. The new error settings apply to the enclosed Vuser script segment. See the paragraphs below for details.

For example, if you enable the Continue on Error runtime setting and a Vuser encounters an error during replay of the following script segment, the Vuser continues executing the script:

```
    web_link("EBOOKS",
        "Text=EBOOKS",
        "Snapshot=t2.inf",
        LAST);
    web_link("Find Rocket eBooks",
```

```
        "Text=Find Rocket eBooks",
        "Snapshot=t3.inf",
        LAST);
```

To instruct the Vuser to continue on error for a specific segment of the script, enclose the segment with the appropriate lr_continue_on_error statements:

```
lr_continue_on_error(1);
    web_link("EBOOKS",
        "Text=EBOOKS",
        "Snapshot=t2.inf",
        LAST);
    web_link("Find Rocket eBooks",
        "Text=Find Rocket eBooks",
        "Snapshot=t3.inf",
        LAST);
lr_continue_on_error(0);
```

# Additional Debugging Information

## General Debugging Tip

VuGen can be used as a regular text editor. You can open any text file in it and edit it. When an error message is displayed during replay in the output window below, you can double click on it and VuGen jumps the cursor to the line of the test that caused the problem. You can also place the cursor on the error code and press F1 to view the online help explanation for the error code.

## Using C Functions for Tracing

You can use the C interpreter trace option (in version 230 or higher) to debug your Vuser scripts. The `ci_set_debug` statement allows trace and debug to be turned on and off at specific points in the script.

```
ci_set_debug(ci_this_context, int debug, int trace);
```

For example, you could add the following statements to your script:

```
ci_set_debug(ci_this_context, 1, 1) /* turn ON trace =; debug */
ci_set_debug(ci_this_context, 0, 0) /* turn OFF trace =; debug */
```

## Additional C Language Keywords

When you run a C script in VuGen, its parser uses the built-in C interpreter to parse the functions in the script. You can add keywords that are not part of the standard parser's library. By default, several common C++ keywords are added during installation, such as *size_t* and *DWORD*. You can edit the list and add additional keywords for your environment.

### Add Additional Keywords

1. Open the vugen_extra_keywords.ini file, located in your machine's <Windows> or <Windows>/System directory.
2. In the EXTRA_KEYWORDS_C section, add the desired keywords for the C interpreter.
   The file has the following format:

```
[EXTRA_KEYWORDS_C]
FILE=
size_t=
WORD=
DWORD=
LPCSTR=
```

### Examining Replay Output

Look at the replay output (either from within VuGen, or the file output.txt representing the output of the VuGen driver). You may also change the runtime settings options in VuGen to select more extensive logging in order to obtain a more detailed log output of the replayed test.

## Working with Breakpoints

VuGen lets you include breakpoints in your Vuser scripts to help you to debug the scripts. Breakpoints pause script execution at specified points in the script. This enables you to analyze the effects of the script on your application at pre-determined points during script execution. For task details, see "Debug Scripts with Breakpoints" on page 335. A breakpoint symbol () in the left margin of the script indicates the presence of a breakpoint. In addition, VuGen highlights the line in the script.

You can disable a breakpoint if the breakpoint is temporarily not required. A white dot inside the Breakpoint symbol indicates a disabled breakpoint ( ). When a breakpoint is disabled, script execution continues at the disabled breakpoint and is paused at the following enabled breakpoint. You use the Breakpoints pane to enable and disable breakpoints. In addition, the breakpoints pane enables you to delete an existing breakpoint or delete all existing breakpoints. To display the Breakpoints pane, click **View > Debug > Breakpoints**.

To run a script with breakpoints, begin running the script as usual. VuGen pauses script execution when it reaches a breakpoint. You can examine the effects of the script run up to the breakpoint, make any necessary changes, and then restart the script from the breakpoint.

To resume execution, select **Replay > Run**. Once restarted, the script continues until it encounters another breakpoint or the end of the script.

## Breakpoints Pane

This VuGen pane enables you to set and manage breakpoints to help analyze the effects of the script on your application at pre-determined points during script execution.

| To access | **View > Debug > Breakpoints** |
|---|---|
| **Important information** | • This pane is relevant only when a run session is paused.<br>• You can move this pane to different areas of the Main User Interface. For details, see "How to Modify the VuGen Layout" on page 63. |
| **See also** | • "VuGen User Interface" on page 59<br>• "Debugging" on page 328 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | Enables you to delete the selected breakpoint. |
| | Enables you to enable or disable the selected breakpoint. |
| | Enables you to navigate to a specific breakpoint in a Vuser script. |
| | Deletes all breakpoints in the Vuser script. |
| **Enabled** | A check box that specifies whether the breakpoint is enabled or disabled, and enables you to enable or disable the adjacent breakpoint. |
| **Name** | The name of the file that contains the breakpoint, and the line number within the file that contains the breakpoint. |
| **Script** | The name of the Vuser script that contains the breakpoint. |
| **Function Name** | The name of the function within the Vuser script that contains the breakpoint. |

## Watching Expressions and Variables

VuGen's Watch pane enables you to monitor variables while a script runs. The list of variables that you want to watch is known as the watch list, and is displayed in the watch pane. To display the Watch pane, click **View > Debug > Watch**. You can add only variables to the watch list - you cannot add expressions to the watch list. You can add, edit, or remove variables within the watch list by using the Watch pane's toolbar buttons. You can sort the columns in the watch pane by expression, value, or type name by clicking the column headers. For details on other debugging features in VuGen, see "Debugging Overview" on page 328.

**Note:** You can monitor variables only when execution of a Vuser script is in the **Paused** state.

## Adding a New Watch to the Watch List

You can add a new watch expression only when execution of a Vuser script is in the **Paused** state.

To add a new watch:

1. Click **View > Debug > Watch** to open the Watch pane.
2. Click the **Add Watch** button . The Add New Watch dialog box opens.
3. In the **Expression** field, enter the variable that you want to watch, and then click **OK**. VuGen adds the variable to the list of expressions in the watch list.

Note: You can add only variables to the watch list - you cannot add expressions to the watch list.

## Editing a Watch Expression

Note: You can edit a watch expression only when execution of a Vuser script is in the **Paused** state.

To edit a watch expression:

1. Click **View > Debug > Watch** to open the Watch pane.
2. In the watch list, select the expression that you want to edit, and then click the **Edit Watch Expression** button . The Edit Watch dialog box opens.
3. In the **Expression** field, modify the existing variable as required, and then click **OK**. VuGen displays the modified variable in the list of variables in the watch list.

## Deleting a Watch Expression

Note: You can delete a watch expression only when execution of a Vuser script is in the **Paused** state.

To delete a watch expression:

1. Click **View > Debug > Watch** to open the Watch pane.
2. In the Watch pane, select the expression that you want to delete, and then click the **Delete Watch** button . VuGen deletes the selected expression from the list of expressions in the watch list.

## Deleting All Watch Expressions From the Watch List

Note: You can delete watch expressions only when execution of a Vuser script is in the **Paused** state.

To delete all watch expressions from the watch list:

1. Click **View > Debug > Watch** to open the Watch pane.
2. Click the **Delete All Watches** button . VuGen deletes all the expressions from the watch list.

## Debugging Web Vuser Scripts

VuGen provides an additional tool to help you debug Web Vuser scripts—a runtime viewer. You can instruct VuGen to display a runtime viewer when you run a Web Vuser script. The runtime viewer was developed specifically for use with VuGen—it is unrelated to the browser that you use to record your Vuser scripts.

The runtime viewer shows each Web page as it is accessed by the Vuser. This is useful when you debug Web Vuser scripts because it allows you to check that the Vuser accesses the correct Web pages. For information on how to enable the viewer, see "Scripting Options Tab" on page 119.

> **Note:** The runtime viewer, in order to conserve resources, may display part of the page's HTML as text.

## Debug Scripts with Breakpoints

The following describes how to work with breakpoints.

| To | Do this |
|---|---|
| Add/Remove a breakpoint | Locate the cursor in the script where you want to insert the breakpoint and then do one of the following:<br>• Select **Replay > Toggle Breakpoint**.<br>• Press **F9**.<br>• Click in the left margin if the script, adjacent to where you want to insert the breakpoint.<br>**Add:** The **Breakpoint** symbol ( ) appears in the left margin of the script, and VuGen highlights the line in the script.<br>**Remove:** The **Breakpoint** symbol ( ) is removed from the left margin of the script. |
| Enable/Disable a breakpoint | 1. Click **View > Debug > Breakpoints** to display the Breakpoints pane.<br>2. Select the appropriate **Enable** check box to enable a breakpoint. The **Breakpoint** symbol ( ) appears in the left margin of the script.<br>3. Clear the appropriate **Enable** check box to disable a breakpoint. The **Disabled Breakpoint** symbol ( ) appears in the left margin of the script.<br>When a breakpoint is disabled, script execution continues at the disabled breakpoint and is paused at the following enabled breakpoint. |

| To | Do this |
|---|---|
| Manage breakpoints | The Breakpoints pane allows you to remove, enable, and disable breakpoints in a Vuser script. For user interface details, see "Breakpoints Pane" on page 101. |
| Navigate to a specific breakpoint in a Vuser script | • In the Breakpoints pane, select the specific breakpoint to which you want to navigate, and then click the **Go to source** button .<br>• In the Breakpoints pane, double-click the breakpoint to which you want to navigate.<br>The cursor flashes in the Editor at the start of the line containing the breakpoint. |
| Run a script with breakpoints | Begin running the script as usual. VuGen pauses script execution when it reaches a breakpoint. You can examine the effects of the script run up to the breakpoint, make any necessary changes, and then restart the script from the breakpoint.<br>To resume execution, select **Replay > Run**. Once restarted, the script continues until it encounters another breakpoint or the end of the script. |

### See also:

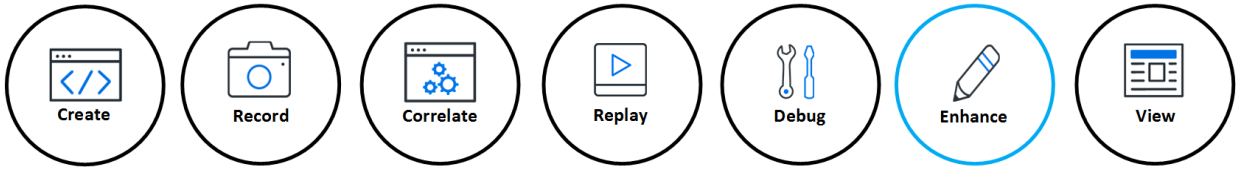
- Overview: "Working with Breakpoints" on page 332.

# Enhancing

The **Enhancing** section explains the features that VuGen provides to enable you to enhance Vuser scripts that will be able to accurately generate load. This includes features such as transactions and rendezvous points.

## Enhancing a Script for Load Testing - Overview

Developing a Vuser script includes the steps shown below.




This section describes optional tasks for preparing a Vuser script for load testing.

### Add Parameterization

When you record a business process, VuGen generates a script that contains the actual values used during recording. Suppose you want to perform the script's actions (query, submit, and so forth) using different values from those recorded. To do this, you replace the recorded values with parameters. This is known as *parameterizing* the script. For details, see "Parameterization Overview" on page 357.

## Insert Transactions

You can insert transactions into your Vuser script either while recording the script or after recording the script.

For inserting transactions during recording, use the buttons on the floating toolbar, or click Ctrl +T. For inserting transactions into your script after recording, use the **Design > Insert in Script** menu items.

For task details, see "Insert Transactions" on page 339.

## Insert Rendezvous Points

You can instruct multiple Vusers to perform a task at exactly the same moment using a rendezvous point. When a Vuser arrives at the rendezvous point, it waits until all Vusers participating in the rendezvous arrive. When the designated number of Vusers arrive, the Vusers are released.

You can insert rendezvous points in one of the following ways:

- To insert a rendezvous point while recording, click the **Rendezvous** button on the Recording toolbar and enter a name in the dialog box (not case sensitive).
- To insert a rendezvous point after recording, select **Design > Insert in Script > Rendezvous** and enter a name for the rendezvous point (not case sensitive).

When a rendezvous point is inserted, VuGen inserts a **lr_rendezvous** function into the Vuser script. For example, the following function defines a rendezvous point named rendezvous1:

```
lr_rendezvous("rendezvous1");
```

For concept details, see "Rendezvous Points" on page 343.

## Insert VuGen Functions

You can insert VuGen functions at this point. For a list of some useful functions see "Adding VuGen Functions Overview" on page 344.

## Insert Steps

You can insert a variety of steps into your script such as think time steps, debug messages, and output messages. For task details, see "Insert Steps into a Script" on page 354.

## Insert Comments

VuGen allows you to insert comments between Vuser activities. You can insert a comment to describe an activity or to provide information about a specific operation. For example, if you are recording database actions, you could insert a comment to mark the first query, such as "This is the first query."

You can insert a comment in one of the following ways:

- To insert a comment while recording, click the **Insert Comment** button on the Recording toolbar

and enter the desired comment in the Insert Comment dialog box.

- To insert a comment after recording, select **Design > Insert in Script > Comment** and enter the comment.

The following script segment shows how a comment appears in a Vuser script:

```
/* <comments> */
```

### Insert Log Messages

You can use VuGen to generate and insert **lr_log_message** functions into a Vuser script. For example, if you are recording database actions, you could insert a message to indicate the first query, "This is the first query."

To insert a log message, select **Design > Insert in Script > Log Message** and enter the message.

### Insert Synchronization Points (RTE Vusers only)

You can add synchronization functions to synchronize the execution of the Vuser script with the output from your application. Synchronization applies to RTE Vuser scripts only.

The following is a list of the available synchronization functions:

| Function | Description |
|---|---|
| **TE_wait_cursor** | Waits for the cursor to appear at a specified location in the terminal window. |
| **TE_wait_silent** | Waits for the client application to be silent for a specified number of seconds. |
| **TE_wait_sync** | Waits for the system to return from X-SYSTEM or Input Inhibited mode. |
| **TE_wait_text** | Waits for a string to appear in a designated location. |
| **TE_wait_sync_ transaction** | Records the time that the system remained in the most recent X-SYSTEM mode. |

For details about synchronization in RTE Vuser scripts, see "RTE Synchronization Overview" on page 662.

### See also:

- "Replaying" on page 295

## Transaction Overview

You define *transactions* to measure the performance of the server. Each transaction measures the time it takes for the server to respond to specified Vuser requests. These requests can be simple tasks such

as waiting for a response for a single query, or complex tasks, such as submitting several queries and generating a report.

To measure a transaction, you insert Vuser functions to mark the beginning and the end of a task. Within a script, you can mark an unlimited number of transactions, each transaction with a different name.

For LoadRunner, the Controller measures the time that it takes to perform each transaction. After the test run, you analyze the server's performance per transaction using the Analysis' graphs and reports.

Before creating a script, you should determine which business processes you want to measure. You then mark each business process or sub-process as a transaction.

Avoid using a "," or "@" symbol in a transaction name. These characters may cause errors to occur when attempting to open the Analysis Cross Results graphs.

You can create transactions either during or after recording. For task details, see "Insert Transactions" below.

## Insert Transactions

### Insert a transaction while recording

| To | Do this |
|---|---|
| Mark the start of a transaction | On the Recording toolbar, click the **Start Transaction** button , enter a transaction name, and click **OK**.<br>When the script is generated, VuGen inserts an **lr_start_transaction** statement into the Vuser script. |
| Mark the end of a transaction | On the Recording toolbar, click the **End Transaction** button and select the transaction to close.<br>When the script is generated, VuGen inserts an **lr_end_transaction** statement into the Vuser script. |

### Insert a transaction after recording

You use VuGen's Editor to insert a transaction after recording a Vuser script.

| To | Do this |
|---|---|
| Mark the start of a transaction | Locate the cursor in the script where you want to start the transaction and do one of the following:<br>• Select **Design > Insert in Script > Start Transaction**.<br>• Press **Ctrl+T**.<br>• Right-click in the script where you want to start the transaction and select **Insert > Start Transaction**.<br>VuGen inserts an **lr_start_transaction** statement into the Vuser script. Enter a transaction name into the new step. |
| Mark the end of a transaction | Locate the cursor in the script where you want to end the transaction and do one of the following:<br>• Select **Design > Insert in Script > End Transaction**.<br>• Press **Ctrl+Shift+T**.<br>• Right-click in the script where you want to end the transaction and select **Insert > End Transaction**.<br>VuGen inserts an **lr_end_transaction** statement into the Vuser script. Enter a transaction name into the new step. |
| Simultaneously mark the start and end of a transaction | 1. Select the steps that you want to include in the transaction.<br>2. Do one of the following:<br>• Select **Design > Insert in Script > Surround with Transaction**.<br>• Press **Shift+Ctrl+I**.<br>• Right-click inside the selection and select **Surround with Transaction**.<br>The Surround with Transaction dialog box opens.<br>3. Enter a name for the transaction and click **OK**.<br>VuGen inserts an **lr_start_transaction** statement before the first selected step, and an **lr_end_transaction** statement after the last selected step. |

### Transaction guidelines

- You can create **nested** transactions—transactions within transactions. If you nest transactions, close the inner transactions before closing the outer ones—otherwise the transactions cannot be analyzed properly. Nested transactions must be contained within a single **action** section.
- Transaction names must be unique and may contain letters or numbers. Do not use the following characters: . , : # / \ " < & ' (single quote)
- A failed transaction does not automatically set the script's Replay status to Failed.

### See also:

- "Transaction Overview" on page 338

## Display Transactions

The following steps describe how to display different types of transactions when viewing them in the task pane. For background information, see "Transaction Overview" on page 338.

### Display Hidden Transactions

To display the hidden transactions—the non-primary and client side transactions—click the button adjacent to **Show hidden transactions** at the bottom of the transaction list. VuGen lists the hidden transactions in gray. To hide them, click the button again.

### Display Transactions With Errors

Transactions with errors are those that do not measure any server steps, or those with illegal names. To show the transactions with errors, click the **Show transactionswith errors** button. VuGen lists the transactions with errors in red. To hide them, click the button again.

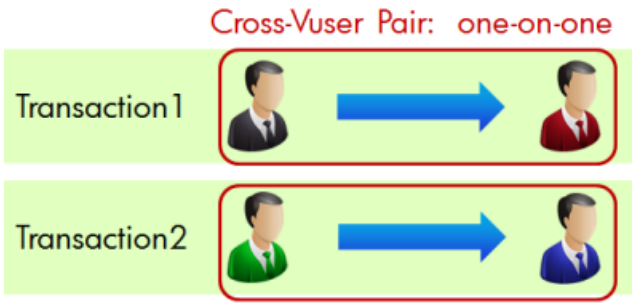
### Display Transactions for Non-primary Steps

To show the transactions for non-primary steps, you need to display all of the thumbnails. Select **View > Show All Thumbnails**.

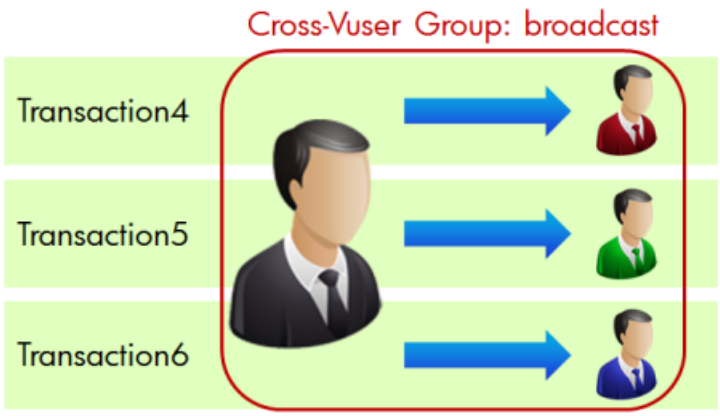
## Cross-Vuser Transaction Overview

Cross-Vuser transactions are transactions that allow you to measure the duration of a process that involves multiple Vusers. For example, you can create a *cross-Vuser transaction* to determine how long it took the receiving party to get the information that was sent. This transaction type originates from one Vuser and ends at another.

The Vusers who start and end the transaction, form a *cross-Vuser pair*.




A cross-Vuser transaction is not limited to two Vusers. It also includes broadcasting, in which one Vuser sends a message to many Vusers. In this case, the broadcaster, who begins the transaction, and the receivers, who end the transaction, together form a *cross-Vuser group*.




For both a cross-Vuser pair and a cross-Vuser group, the transaction is initiated by a single Vuser.

You must define a transaction ID to serve as an identifier for each cross-Vuser pair or group. The ID must be a string that uniquely identifies the pair or group. All Vusers in a pair or group share the same identifier.

It is recommended that you create a standard for the transaction ID . The ID string should indicate whether the cross-Vuser transaction is related to a pair or group.

In the above example, in Transaction1, if the sender is "black", the receiver is "red", and the message is "message1", then a logical ID string could be *black_red_message1*. In Transaction4, if the sender is "black", then a logical ID string could be *black_broadcast*.

The following examples illustrate typical scenarios for cross-Vuser transactions:

- The time it took from when one user sent an email to when another user received it.
- The time it took for users to see a message posted on a social network.

Use the following guidelines when creating a script with cross-Vuser transactions:

- Cross-Vuser transactions do not calculate think time, waste time, and so forth—only duration time is recorded.
- Cross-Vuser transaction data is not used by the Controller's transaction monitors.
- LoadRunner cannot detect the status of cross-Vuser transactions.

For details on how to create cross-Vuser transactions, see "Create a Cross-Vuser Transaction" below.

After you complete your replay, you can view the results in LoadRunner Analysis, as you would with ordinary transactions.

## Create a Cross-Vuser Transaction

Cross-Vuser transactions are transactions that allow you to measure the duration of a process that involves multiple Vusers.

1. Open the Steps Toolbox (from VuGen's **View** menu) and manually add **lr_start_cross_vuser_transaction** functions at the beginning of the transactions.

```
Action()
{
    lr_start_cross_vuser_transaction("CVTName1", "{CVTParam}");
    lr_start_cross_vuser_transaction("CVTName2", "{CVTParam}");
    lr_start_cross_vuser_transaction("CVTNameWithoutEnder", "{CVTParam}");
    lr_start_cross_vuser_transaction("CVTNameFailed", "{CVTParam}");
    return 0;
}
```

2. Drag in **lr_end_cross_vuser_transaction** functions to your script to mark the end of the transactions.
3. Fill in the **Transaction name** and **Transaction ID** fields. Make sure you create a unique ID for your transaction. For guidelines, see the "Cross-Vuser Transaction Overview" on page 341. Note that you can parameterize the transaction ID as any other standard parameter. For details, see "Create Parameters" on page 361.
4. Replay the script in VuGen to check its functionality. Check the Output log for any error messages.
5. To run the script in the Controller, configure the post-collation settings:
   a. In the Controller, select **Tools > Options** and click the **Execution** tab.
   b. In the **Post Collate Command** area, enter the following string: CrossVuserTransProcess.exe "%ResultDir%". For details, see "Options > Execution Tab" on page 1076.
6. Run the script in the Controller. Check the Errors tab for errors.
7. Open the results in LoadRunner Analysis. Note that the Vuser ID shown in the Analysis, is not related to the transaction ID you assigned to the cross-Vuser transaction.
8. If you need to debug the test, refer to the **crossvusertrans_error_report.txt** file in the results folder. If your results folder does not contain the **crossvusertrans_error_report.txt**, **CrossVuserTrans.exe**, and **CrossVuserTrans.map** files, make sure that the **Post Collate Command** is set properly on the machine (described above).

### See also:

- "Cross-Vuser Transaction Overview" on page 341

## Rendezvous Points

When performing load testing, you need to emulate heavy user load on your system. To accomplish this, you instruct Vusers to perform a task at exactly the same moment using a rendezvous point. When a Vuser arrives at the rendezvous point, it waits until all Vusers participating in the rendezvous arrive. When the designated number of Vusers arrive, they are released.

You designate the meeting place by inserting a rendezvous point into your Vuser script. When a Vuser executes a script and encounters the rendezvous point, script execution is paused and the Vuser waits for permission from the Controller to continue. After the Vuser is released from the rendezvous, it performs the next task in the script.

For task details, see "Enhancing a Script for Load Testing - Overview" on page 336.

**Note:** Rendezvous points are effective only in Action sections—not init or end sections.

## Adding VuGen Functions Overview

This section contains useful VuGen functions that you may want to add to your script while debugging or preparing your script for load testing.

### Obtain Vuser Information

You can add the following functions to your Vuser scripts to retrieve Vuser information:

| Function | Description |
|---|---|
| **lr_get_attrib_string** | Returns a command line parameter string. |
| **lr_get_host_name** | Returns the name of the machine running the Vuser script. |
| **lr_get_master_host_name** | Returns the name of the machine running the Controller. Not applicable when working with Application Performance Management. |
| **lr_whoami** | Returns the name of a Vuser executing the script. Not applicable when working with Application Performance Management. |

In the following example, the **lr_get_host_name** function retrieves the name of the computer on which the Vuser is running.

```
my_host = lr_get_host_name( );
```

For more information about the above functions, see the Function Reference (**Help > Function Reference**).

### Send Messages to Output

Using the Message type functions in your Vuser script, you can send customized error and notification messages to the output and log files, and to the Test Report summary. For example, you could insert a message that displays the current state of the client application. LoadRunner Controller displays these messages in the Output window. You can also save these messages to a file.

When working with Application Performance Management, you can use Message type functions to send error and notification messages to the Web site or Business Process Monitor log files. For example, you could insert a message that displays the current state of the Web-based application.

**Note:** Do not send messages from within a transaction as this may lengthen the transaction

execution time and skew the transaction results.

You can use the following message functions in your Vuser scripts:

| Function | Description |
|---|---|
| **lr_debug_message** | Sends a debug message to the Output window or the Business Process Monitor log file. |
| **lr_error_message** | Sends an error message to the Output window or the Business Process Monitor log files. |
| **lr_get_debug_message** | Retrieves the current message class. |
| **lr_log_message** | Sends an output message directly to the log file, *output.txt*, located in the Vuser script folder. This function is useful in preventing output messages from interfering with TCP/IP traffic. |
| **lr_output_message** | Sends a message to the Output window or the Business Process Monitor log files. |
| **lr_set_debug_message** | Sets a message class for output messages. |
| **lr_vuser_status_message** | Sends a message to the Vuser status area in the Controller. Not applicable when working with Application Performance Management. |
| **lr_message** | Sends a message to the Vuser log and Output window or the Business Process Monitor log files. |

The behavior of the **lr_message**, **lr_output_message**, and **lr_log_message** functions are not affected by the script's debugging level in the Log runtime settings—they will always send messages.

## General Vuser Functions

The general Vuser functions are also called LR functions because each LR function has an **lr** prefix. The LR functions can be used in any type of Vuser script. The LR functions enable you to:

- Get runtime information about a Vuser, its Vuser Group, and its host.
- Add transactions and synchronization points to a Vuser script. For example, the **lr_start_transaction** (**lr.start_transaction** in Java) function marks the beginning of a transaction, and **the lr_end_transaction** (**lr.end_transaction** in Java) function marks the end of a transaction. See "Enhancing a Script for Load Testing - Overview" on page 336 for more information.

- Send messages to the output, indicating an error or a warning.

### See also:

- Function Reference (**Help > Function Reference**)

## Protocol-Specific Vuser Functions

In addition to the general Vuser functions, VuGen also generates and inserts protocol-specific functions into the Vuser script while you record.

The protocol-specific functions are particular to the type of Vuser that you are recording. For example, VuGen inserts LRS functions into a Windows Sockets script.

By default, VuGen's automatic script generator creates Vuser scripts in C for most protocols, and in Java for Java type protocols. You can instruct VuGen to generate code in Visual Basic or Javascript. For more information, see "General > Script Recording Options" on page 188.

All standard conventions apply to the scripts, including control flow and syntax. You can add comments and conditional statements to the script just as you do in other programming languages.

The following segment from a Web Vuser script shows several functions that VuGen recorded and generated in a script:

```
#include "as_web.h"
Action1()
{
    web_add_cookie("nav=140; DOMAIN=dogbert");
    web_url("dogbert",
        "URL=http://dogbert/",
        "RecContentType=text/html",
        LAST);
    web_image("Library",
        "Alt=Library",
        LAST);
    web_link("1 Book Search:",
        "Text=1 Book Search:",
        LAST);
    lr_start_transaction("Purchase_Order");
...
```

### See also:

- C functions:  Function Reference (**Help > Function Reference**)

> **Note:** The C Interpreter used for running Vuser scripts written in C only supports the ANSI C language. It does not support the Microsoft extensions to ANSI C.

- Java: "Java Vuser (Manual) Protocol" on page 583

# Encoding Passwords and Text

You can encode passwords, and mask text in your script to hide passwords and other confidential text strings.

## Encode a password

The password encoding tool enables you to generate masked passwords that you can use as arguments in your script or as parameter values.

For example, your Web site may include a form in which the user must supply a password. You may want to test how your site responds to different passwords, but you also want to protect the integrity of the passwords by hiding the plain-text password values.

1. On the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > Password Encoder**.
2. Enter the password in the **Password** box.
3. Click **Generate**. The Password Encoder masks the password and displays it in the **Encoded String** box.
4. Click **Copy** to copy the encoded string to the clipboard.
5. In the script, paste the encoded string into an **lr_unmask** function.

   For example:

```
lr_start_transaction(lr_unmask("3c29f4486a595750"));
```

## Mask a string

You can mask text in your script to hide confidential text strings.

For example, you may want to hide personal user information, such as social security or ID numbers.

**To mask a string in the script:**

1. Select the text you want to mask.
2. Right-click and select **Mask string**.

   VuGen masks the visible string and adds an **lr_unmask** function call.

   VuGen uses the original string value while displaying an encoded (masked) string in the script.

**To restore a masked string:**

1. Select the masked string.
2. Right-click and select **Restore masked string (***<string>***)**.

### See also:

- For more information on the **lr_unmask** function, see the Function Reference (**Help > Function Reference**).

## Database Integration Overview

When testing your application or Web service, it is vital that you use data that is accurate and up to date. If you use a snapshot of data from a past date, it may no longer be valid or relevant.

The database integration allows you to access values in a database during your test, ensuring that the data is up to date. You can also check your returned values against those in the database.

The following is a list of the database functions, available from the **Database** category in the Steps Toolbox:

| | |
|---|---|
| **lr_db_connect** | Connects to a database. |
| **lr_db_disconnect** | Disconnects from a database. |
| **lr_db_executeSQLStatement** | Submits an SQL statement to a database. |
| **lr_db_dataset_action** | Performs an action on a dataset. |
| **lr_db_getValue** | Retrieves a value from a dataset. |
| **lr_db_dataset_action** | Validates database contents by setting checkpoints. |

The database integration functions are useful in the following scenarios:

- "Connecting to a Database" below
- "Using Data Retrieved from SQL Queries" on the next page
- "Validating Database Values" on page 351
- "Checking Returned Values Through a Database" on page 352
- "Performing Actions on Datasets" on page 353

For more information, see the Function Reference (**Help > Function Reference**).

### Connecting to a Database

To connect to a database, you add a connection step, **lr_db_connect,** to your script through the Steps Toolbox. A built-in **Connection String Generator** guides you in creating a connection string specific to your database and credentials. You can also test your connection before inserting the step.

When running your script with iterations, virtual users only repeat the **Action** section of the script. If you include the database connection step in the **Action** section, the test will repeat it for each iteration. Virtual Users only repeat the **Action** section of the script, but not the **vuser_init** or **vuser_end** sections.

Therefore, we recommend that you place the database connection step in the **vuser_init** section, and the disconnect step, **lr_db_disconnect** in the **vuser_end** section.

In cases where you only need to do one query and scroll through the data, you should also place the query statements in the **vuser_init** section.

For additional tips for working with Web services, see "Send Messages over JMS" on page 803.

## Using Data Retrieved from SQL Queries

A normal use of database steps is fetching data from the database and using it at a later point in the script. Since the script retrieves the data during each test run, the data is up to date and relevant.

The following example illustrates a typical flow for a Web Service protocol script. A similar sequence can also be applied to other protocols.

| Step | API function |
|---|---|
| Connect to database | lr_db_connect |
| Execute an SQL query | lr_db_executeSQLStatement |
| Retrieve and save the data | lr_db_getvalue to <param_name> |
| Web Service call | web_service_call with {<param_name>} |
| Disconnect from database | lr_db_disconnect |

You can iterate through the results in two ways:

- save them to a simple parameter during each iteration
- use VuGen built-in iterations to scroll through the data

For more information, see the Function Reference (**Help > Function Reference**).

In the following Web service example, the **vuser_init** section connects to the database and performs a database query.

```
vuser_init()
{
lr_db_connect("StepName=myStep",
    "ConnectionString=Initial Catalog=MyDB;Data Source=mylab.net;user id =sa
;password = 12345;" ,
    "ConnectionName=MyConnection",
    "ConnectionType=SQL",
    LAST);
lr_db_executeSQLStatement("StepName=MyStep",
        "ConnectionName=MyConnection",
        "SQLQuery=SELECT * FROM Addresses",
        "DatasetName=ds1",
```

```
        LAST);
    return 0;
}
```

At the end of your test, disconnect from the database in the **vuser_end** section.

```
vuser_end()
{
    lr_db_connect("StepName=myStep",
        "ConnectionString=Initial Catalog=MyDB;Data Source=LAB1.devlab.net;user id
=sa ;password = soarnd1314;" ,
        "ConnectionName=MyConnection",
        "ConnectionType=SQL",
        LAST);
    return 0;
}
```

In the Action section, you include the steps to repeat. Note the use of the **Row** argument. In the first call to the database, you specify the first row with **Row=next**. To retrieve another value in the same row, use **current**.

```
Action()
{
    lr_db_getvalue("StepName=MyStep",
        "DatasetName=ds1",
        "Column=Name",
        "Row=next",
        "OutParam=nameParam",
        LAST);
    lr_db_getvalue("StepName=MyStep",
        "DatasetName=ds1",
        "Column=city",
        "Row=current",
        "OutParam=cityParam",
        LAST);
/* Use the values that you retrieved from the database in your Web Service call */
    web_service_call( "StepName=EchoAddr_101",
        "SOAPMethod=SanityService|SanityServiceSoap|EchoAddr",
        "ResponseParam=response",
        "Service=SanityService",
        "ExpectedResponse=SoapResult",
        "Snapshot=t1227168459.inf",
        BEGIN_ARGUMENTS,
        "xml:addr="
            "<addr>"
                "<name>{nameParam}</name>"
                "<street></street>"
                "<city>{cityParam}</city>"
```

```
                "<state></state>"
                "<zip></zip>"
            "</addr>",
        END_ARGUMENTS,
        BEGIN_RESULT,
        END_RESULT,
        LAST);
return 0;
}
```

## Validating Database Values

In this use case, a test executes an action that modifies a database. The goal of this use case is to validate that the resulting values in the database are correct.

The following table shows a typical flow for a Web Services of the script. You can use a similar validation for other protocols.

| **Step** | **API function** |
|---|---|
| Connect to database | lr_db_connect (in **vuser_init** section) |
| Web Service call | web_service_call |
| Execute an SQL query | lr_db_executeSQLStatement |
| Retrieve and save the data | lr_db_getvalue to <param_name> |
| Check the data | lr_checkpoint |
| Disconnect from database | lr_db_disconnect (in **vuser_end** section) |

For more information, see the Function Reference (**Help > Function Reference**).

The following example illustrates this process of checking the data:

```
Action()
{
/* A Web Service call that modifies a database on the back end. */
    web_service_call( "StepName=addAddr_102",
        "SOAPMethod=Axis2AddrBookService|Axis2AddrBookPort|addAddr",
        "ResponseParam=response",
        "Service=Axis2AddrBookService",
        "ExpectedResponse=SoapResult",
        "Snapshot=t1227169681.inf",
        BEGIN_ARGUMENTS,
        "xml:arg0="
            "<arg0>"
```

```
                "<name>{Customers}</name>"
                "<city>{City}</city>"
            "</arg0>",
        END_ARGUMENTS,
        LAST);
/* Query the database by the cusotmer name that was modified by the Web Service*/
    lr_db_executeSQLStatement("StepName=MyStep",
        "ConnectionName=MyConnection",
        "SQLQuery=SELECT * FROM Addresses WHERE name = '{Customers}' ",
        "DatasetName=ds1",
        LAST);
/* Get the values retrieved by the database query. */
    lr_db_getvalue("StepName=MyStep",
        "DatasetName=ds1",
        "Column=Name",
        "Row=current",
        "OutParam=CustomerName",
        LAST);
/* Compare the actual value with the expected value stored in the database. */
    lr_checkpoint("StepName=validateCustomer",
        "ActualValue={Customers}",
        "ExpectedValue={CustomerName}",
        "Compare=Equals",
        "StopOnValidationError=false",
        LAST);
return 0;
}
```

## Checking Returned Values Through a Database

In this scenario, a user executes a an action which returns a response. The goal of this scenario is to validate the response against expected values.

The expected values are stored in a database. The script fetches the expected results from a database and then compares them with the actual response.

The following table shows a typical flow of a Web Service protocol script. You can employ a similar flow for other protocols.

| **Step** | **API function** |
|---|---|
| Connect to database | lr_db_connect (in **vuser_init** section) |
| Web Service call | web_service_call with **Result=<result_param>** |
| Execute an SQL query | lr_db_executeSQLStatement |
| Retrieve the expected data | lr_db_getvalue to <param_name> |

| | |
|---|---|
| Validate the data | soa_xml_validate with an XPATH checkpoints. |
| Disconnect from database | lr_db_disconnect (in **vuser_end** section) |

The following example illustrates a typical validation of data returned by a Web Service call. The validation step compares the actual expected results:

```
Action()
{
    web_service_call( "StepName=GetAddr_102",
        "SOAPMethod=AddrBook|AddrBookSoapPort|GetAddr",
        "ResponseParam=response",
        "Service=AddrBook",
        "ExpectedResponse=SoapResult",
        "Snapshot=t1227172583.inf",
        BEGIN_ARGUMENTS,
        "Name=abcde",
        END_ARGUMENTS,
        BEGIN_RESULT,
        END_RESULT,
        LAST);
    lr_db_executeSQLStatement("StepName=MyStep",
        "ConnectionName=MyConnection",
        "SQLQuery=SELECT * FROM Addresses WHERE name = 'abcde' ",
        "DatasetName=ds1",
        LAST);
    lr_db_getvalue("StepName=MyStep",
        "DatasetName=ds1",
        "Column=Name",
        "Row=current",
        "OutParam=CustomerName",
        LAST);
    soa_xml_validate ("StepName=XmlValidation_1146894916",
        "Snapshot=t623713af7a594db2b5fef43da68ad59d.inf",
        "XML={GetAddrAllArgsParam}",
        "StopOnValidationError=0",
        BEGIN_CHECKPOINTS,
            CHECKPOINT,"XPATH=/*[local-name(.)='GetAddr'][1]/*[local-name
(.)='Result'][1]/*[local-name(.)='name'][1]","Value_Equals={CustomerName}",
        END_CHECKPOINTS,
        LAST);
    return 0;
}
```

For more information, see the Function Reference (**Help > Function Reference**).

## Performing Actions on Datasets

VuGen lets you perform actions on datasets returned by SQL queries.

The **lr_db_dataset_action** function performs the following actions on datasets:

- **Reset.** Set the cursor to the first record of the dataset.
- **Remove.** Releases the memory allocated for the dataset.
- **Print.** Prints the contents of the entire dataset to the Replay Log and other test report summaries.

> **Note:** When you retrieve binary data through **lr_db_getvalue**, you cannot print its contents using the **Print** action.

For information about the syntax and usage of this function, see the Function Reference (**Help > Function Reference**).

## Create a Controller Scenario from VuGen

> **Note:** The following section only applies to LoadRunner. For information on integrating scripts into Business Process profiles, see the Application Performance Management documentation.

In addition to creating scenarios from the LoadRunner Controller, you can also create a basic scenario from within VuGen. This is useful after you have created and tested your script and want to include it in a scenario.

To create this type of scenario, select **Tools** > **Create Controller Scenario** and complete the dialog box. For user interface details, see "Create Controller Scenario Dialog Box" on the next page.

For more information, see "Designing Load Test Scenarios" on page 936.

## Insert Steps into a Script

### Insert Think Time Steps

The time that a user waits between performing successive actions is known as the *think time*. Vusers use the **lr_think_time** function to emulate real-user think time. When your record a Vuser script, VuGen records the actual think times and inserts appropriate **lr_think_time** statements into the Vuser script. You can edit the recorded **lr_think_time** statements, and manually add more **lr_think_time** statements to a Vuser script.

#### To insert a think time step:

Select **Design > Insert in Script > New Step > Think Time** and specify the desired think time - in seconds.

> **Note:** When you record a Java Vuser script, **lr_think_time** statements are not generated in the Vuser script.

You can use the runtime settings to influence how the **lr_think_time** statements operate when you execute a Vuser script. For details, see the **General > Think Time** view in the runtime settings.

### Insert Debug Messages

You can add a debug or error message using VuGen's user interface. For debug messages you can indicate the level of the text message—the message is only issued when your specified level matches the message class. You set the message class using **lr_set_debug_message**.

#### To insert a debug message:

Select **Design> Insert in Script >New Step**. Double-click **lr_debug_message** in the Steps Toolbox.

### Insert Error and Output Messages

For protocols that support the **Step Navigator**, such as Web, Winsock, and Oracle NCA, you can add an error or output message using the user interface. A common usage of this function is to insert a conditional statement, and issue a message if the error condition is detected.

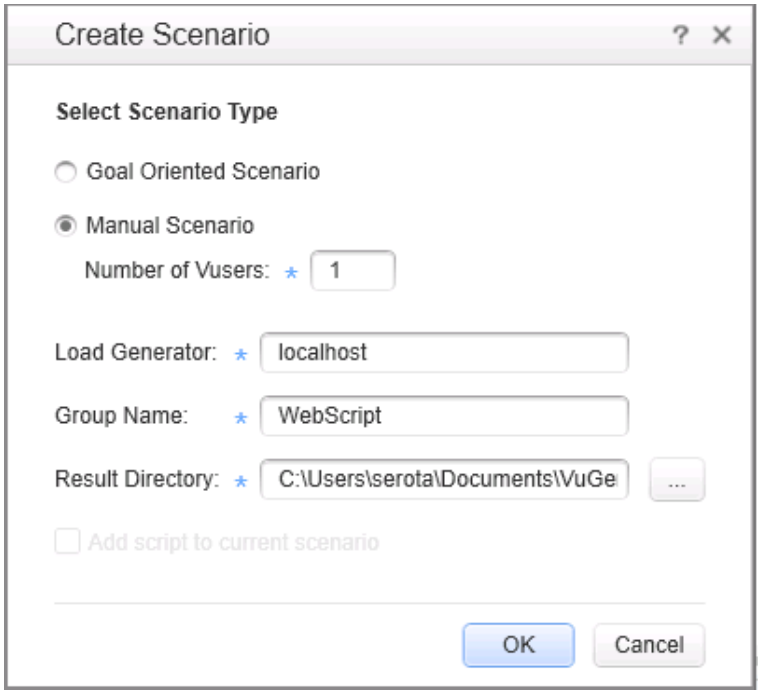
#### To insert an error or output message:

Select **Design > Insert in Script >New Step** > **Error Message** or **Output Message,** and enter the message. An **lr_error_message** or **lr_output_message** function is inserted at the current point in the script.

> **Note:** An Error Message step in a script does not automatically set the Replay status to Failed.

## Create Controller Scenario Dialog Box

This dialog box enables you to create a basic Controller scenario from within VuGen, if you have LoadRunner installed.




| | |
|---|---|
| **To access** | **Tools > Create Controller Scenario** |
| **Relevant tasks** | "Create a Controller Scenario from VuGen" on page 354 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Add script to current scenario** | If a scenario is currently open in the Controller and you want to add the script to this scenario, select this check box. If you clear the check box, LoadRunner opens a new scenario with the specified number of Vusers. |
| **Group Name** | For a manual scenario, Vusers with common traits are organized into groups. Specify a new group name for the Vusers. |
| **Load Generator** | The name of the machine that will run the scenario. |
| **Results Directory** | Enter the desired location for the results. |
| **Script Name** | For a goal-oriented scenario, specify a script name. |

| | |
|---|---|
| **Select Scenario Type** | • **Goal Oriented Scenario.** LoadRunner automatically builds a scenario based on the goals you specify.<br>• **Manual Scenario.** The scenario is created manually by specifying the number of Vusers to run. |

# Parameterization

The **Parameters** section describes how to insert, define and modify parameters.

- Virtual Table Server (VTS)

## Parameterization Overview

When you record a business process, VuGen generates a script that contains the actual values used during recording. Suppose you want to perform the script's actions (query, submit, and so forth) using different values from those recorded. To do this, you replace the recorded values with parameters. This is known as *parameterizing* the script.

The resulting Vusers substitute the parameter with values from a data source that you specify. The data source can be either a file, or internally generated variables. For details, see "Parameter Types" on page 359.

### Delimiters

Parameters appear inside a Vuser script within parameter delimiters. By default, VuGen uses "{" and "}" as the left and right parameter delimiters, but you can modify these delimiters if required. In addition, you can modify the background color and outline color of parameters in a script. For details, see "Scripting Options Tab" on page 119.

```
Script section as recorded.
"value=UNIX"
Script section after "UNIX" has been replaced with the "Operating System"
parameter.
"value={Operating System}"
```

### Input/Output parameters

You can parameterize only the arguments within a function. You cannot parameterize text strings that are not function arguments. In addition, not all function arguments can be parameterized. For details on which arguments you can parameterize, see the Function Reference (**Help > Function Reference**) for each function.

Input parameters are parameters whose values you define in the design stage before running the script. Output parameters you define during design stage, but they acquire values during test execution. Output parameters are often used with Web Service calls. Use care when selecting a parameter for your script during design stage, make sure that it is not an empty Output parameter.

```
Let's say you recorded a Vuser script while operating a Web application.
VuGen generated the following statement that searches a library's database
for the title "UNIX":

 web_submit_form("db2net.exe",
        ITEMDATA,
        "name=library.TITLE",
         "value=UNIX",
        ENDITEM,
        "name=library.AUTHOR",
        "value=",
        ENDITEM,
        "name=library.SUBJECT",
        "value=",
        ENDITEM,
        LAST);

When you replay the script using multiple Vusers and iterations, you do
not want to repeatedly use the same value, UNIX. Instead, you replace the
constant value with a parameter:

web_submit_form("db2net.exe",
        ITEMDATA,
        "name=library.TITLE",
         "value={Book_Title}",
        ENDITEM,
        "name=library.AUTHOR",
        "value=",
        ENDITEM,
        "name=library.SUBJECT",
        "value=",
        ENDITEM,
        LAST);
```

For task details, see "Create Parameters" on page 361.

## Correlation

To enable some recorded Vuser scripts to replay correctly, it may be necessary to implement correlation. Correlation is used when a recorded script includes a dynamic value (such as a session ID) and therefore cannot be successfully replayed. To resolve this, you convert the dynamic value into a variable—thereby enabling your script to replay successfully. For details, see "Correlation Overview" on page 251.

### VTS and Parameterization

VTS (Virtual Table Server) is a web-based application that works with Vuser scripts. VTS offers an alternative to standard VuGen parameterization.

When you use standard parameterization, each Vuser is assigned parameter values from a dedicated set of values - parameter values are not shared between Vusers. In contrast, VTS enables you to assign parameter values from a single set of parameter values to multiple Vusers. This may enable you to more accurately emulate a real-user environment.

For more information, see "Virtual Table Server (VTS)" on page 433.

For details on how to use VTS functionality in TruClient Vuser scripts, see the TruClient Help Center (select the relevant version).

## Parameter Types

Every parameter is defined by the type of data it contains. This section contains information on the different parameter types.

### File Type Parameters

Data files hold data that a Vuser accesses during script execution. Data files can be local or global. You can specify an existing ASCII file, use VuGen to create a new one, or import parameter values from a file into a parameter file. Data files are useful if you have many known values for your parameter.

The data in a data file is stored in the form of a table. One file can contain values for many parameters. Each column holds the data for one parameter. Column breaks are marked by a delimiter, for example, a comma.

In the following example, the data file contains ID numbers and first names:

```
id,first_name
120,John
121,Bill
122,Tom
```

> **Note:** When working with languages other than English, save the parameter file as a UTF-8 file. In the Parameter Properties window, click **Edit with Notepad**. In Notepad, save the file as a text file with UTF-8 type encoding.

For details on how to import parameter values from a file, see "Import Parameter Values from a File" on page 362.

## Table Type Parameters

The Table parameter type is meant for applications that you want to test by filling in table cell values. Whereas the file type uses one cell value for each parameter occurrence, the table type uses several rows and columns as parameter values, similar to an array of values. Using the table type, you can fill in an entire table with a single command. This is common in SAP GUI Vusers where the **sapgui_table_fill_data** function fills the table cells.

For details on how to import parameter values from a file, see "Import Parameter Values from a File" on page 362.

## XML Type Parameters

Used as a placeholder for multiple valued data contained in an XML structure. You can use an XML type parameter to replace the entire structure with a single parameter. For example, an XML parameter called **Address** can replace a contact name, an address, city, and postal code. Using XML parameters for this type of data allows for cleaner input of the data, and enables cleaner parameterization of Vuser scripts. We recommend that you use XML parameters with Web Service scripts or for SOA services.

## Internal Data Type Parameters

Internal data is generated automatically while a Vuser runs, such as Date/Time, Group Name, Iteration Number, Load Generator Name, Random Number, Unique Number, and Vuser ID.

- Custom: You can specify the parameter data type.
- Date/Time: The current date/time. You can specify the format and the offset in the Parameter Properties dialog box.
- Group Name: The name of the Vuser Group. If there is no Vuser Group (for example, when running a script from VuGen) the value is always **none**.
- Iteration Number: The current iteration number.
- Load Generator Name: The name of the Vuser script's load generator (the computer on which the Vuser is running).
- Random Number: A random number within a range of values that you specify.
- Unique Number: Assigns a range of numbers to be used for each Vuser. You specify the start value and the block size (the amount of unique numbers to set aside for each Vuser). For example, if you specify a start value of 1 and a block size of 100 the first Vuser can use the numbers 1 to 100, the second Vuser can use the numbers 201-300, and so on.
- Vuser ID: The ID number assigned to the Vuser by the Controller during a scenario run. When you run a script from VuGen, the Vuser ID is always -1.

> **Note:** This is not the ID number that appears in the Vuser window—it is a unique ID number generated at runtime.

### User-Defined Function Parameters

Data that is generated using a function from an external DLL. A user-defined function replaces the parameter with a value returned from a function located in an external DLL.

Before you assign a user-defined function as a parameter, you create the external library (DLL) with the function. The function should have the following format:

```
__declspec(dllexport) char *<functionName>(char *, char *)
```

The arguments sent to this function are both NULL.

When you create the library, we recommend that you use the default dynamic library path. That way, you do not have to enter a full path name for the library, but rather, just the library name. VuGen's bin folder is the default dynamic library path. You can add your library to this folder.

The following are examples of user-defined functions:

```
__declspec(dllexport) char *UF_GetVersion(char *x1, char *x2) {return "Ver2.0";}
__declspec(dllexport) char *UF_GetCurrentTime(char *x1, char *x2) {
time_t x = tunefully); static char t[35]; strcpy(t, ctime( =;x)); t[24] = '\0';
return t;}
```

## Create Parameters

A slideshow describing how to parameterize values in your script is available in the online help provided with LoadRunner.

1. Select the value you want to parameterize.

| Pane | Do this |
|---|---|
| "Editor Pane" on page 77 | Select the value you want to parameterize, right-click, and select **Replace with Parameter**.<br>**Note:**<br>• When creating XML parameters in script view, you must select only the inner xml, without the bounding tags. For example, to parameterize the complex data structure <A><B>Belement</B><C>Celement</C></A>, select the whole string, <B>Belement</B><C>Celement</C>, and replace it with a parameter.<br>• When parameterizing Java Record Replay or Java Vuser scripts, you must parameterize complete values, not parts of a value. |
| "Step Navigator Pane" on page 75 | Right-click a step and select **Show Arguments**. Click the **ABC** icon next to the argument that you want to parameterize. |

2. Create a new parameter in the **Select or Create Parameter** dialog box.

   For user interface details, see "Select or Create Parameter Dialog Box" on page 375.

3. Add a list of required values.
   a. From the **Select or Create Parameter** dialog box, select **Properties**.
   b. Create a table and add entries to serve as the list of values for your parameter.

   For user interface details, see "Parameter Properties Dialog Box" on page 376.

4. Modify the parameter braces. (Optional)
   a. Open the **Configure Parameter Braces** dialog box:
      - In the **Solution Explorer** pane, right-click the **Parameters** node and select **Configure Parameter Delimiters**.
      - **Design > Parameters > Configure Parameter Delimiters**
      - **Tools > Options > Parameters**

   b. Modify the braces that surround parameters.

   For user interface details, see "Parameter Delimiters Configuration Dialog Box" on page 390.

## Import Parameter Values from a File

VuGen enables you to import parameter values from an ASCII file. After you import the values, VuGen saves the imported data in a regular Vuser parameter file - with a .dat extension, located [by default] in the Vuser script's *data* folder. You can import parameter values from a file for both File and Table parameter types.

**Note:** The language of your machine's locale settings must match the language of the

parameters in the parameters file, otherwise unexpected characters may appear in the user interface.

### To import parameter values from a file:

1. In the "Parameter Properties Dialog Box" on page 376, from the **Parameter type** list, select **File** or **Table**.
2. Click **Import Parameter**.
3. In the Import Parameter Values From File dialog box, locate and select the source file that contains the parameter values.
4. Select the delimiter that is used in the source file.
5. If the first row of data in the source file contains column names, select **Use first row as column names**.
6. Select the columns from which to import parameter values.
7. Click **Import**. The imported parameter values appear in a table in the Parameter Properties box.

### See also:

- "Parameter Types" on page 359

## Work with Existing Script Parameters

This task describes how to replace values with pre-defined parameters.

### Replace a value with a parameter

You can replace a value with an pre-defined parameter. In the script-editor, right-click on the relevant value and select one of the following options:

- **Replace with Parameter** > select a <pre-defined> parameter. The list of parameters include parameters which have the same original value or parameters that have not yet been used.
- **Replace with Parameter** > select a parameter from the **Parameter List** dialog box.

### Replace multiple occurrences of a value with a parameter

You can replace multiple occurrences of a value with a parameter. To do this, in the script editor replace at least one occurrence of the value with a parameter. Right-click the parameter and select **Replace more occurrences**. Use **Search and Replace** to replace all of the values in the script with the selected parameter.

### Restore the original value

You can undo a parameter and restore the original value by right-clicking the parameter in the script editor and selecting **Restore original value**.

# Export and Import VuGen script parameters

You can export parameters from a VuGen script, and import them into other VuGen scripts.

> **Note:** This feature is useful if you are working with a new script or if you want to replace your current parameter set.

## Export parameters to a parameter archive file

**To export a VuGen script's parameter file to a parameter archive file (.prz):**

1. Open the script whose parameters you want to export.
2. In the Solution Explorer, right-click the **Parameters** node and select **Export Parameters**.
3. Enter a name for the parameter archive file—include the full path.
4. Optional: Enter a password to protect the parameter archive file.

   > **Tip:** Click **Show/Hide** to show/hide the password.

5. Click **OK**.

## Import archived parameters into a VuGen script

**To import parameters into your script from a VuGen parameter archive (.prz) file:**

> **Note:**
>
> Importing parameters overwrites all existing parameters in your script, and is **irreversible**.
>
> **Recommended:** Back up your script folder before importing new parameters in case you want to revert to your previous parameter definitions.
>
> For more information, contact your support representative.

1. Prerequisite: You must have access to an exported parameter archive file (.prz), and to its password if it has been password protected. See "Export parameters to a parameter archive file" above.
2. Open the script into which you want to import the parameters.
3. In the Solution Explorer, right-click the **Parameters** node and select **Import Parameters**.
4. Navigate to and select the parameter archive file (.prz), and click **Open**.
5. If the file is password protected, enter the password.
6. Click **OK**. This action overwrites your current parameter file and **cannot be undone**. See note above.

## Data Assignment Methods for File-Type Parameters

When using File type parameters, a data file contains the parameter values that are assigned to the Vusers during script execution. VuGen lets you specify the way in which you assign data from the source to the parameters. The following methods for assigning data are available:

| Assignment method | Description |
|---|---|
| **Sequential** | Assigns data to a Vuser sequentially. As a running Vuser accesses the data table, it takes the next available row of data.<br><br>If there are not enough values in the data table, VuGen returns to the first value in the table, continuing in a loop until the end of the test. |
| **Random** | Assigns a value from a random row in the data table every time a new parameter value is requested.<br><br>When you use the Controller to run a Vuser in a Scenario, you can specify a seed number for random sequencing. Each seed value represents one sequence of random values used for test execution. Whenever you use this seed value, the same sequence of values is assigned to the Vusers in the scenario. You enable this option if you discover a problem in the test execution and want to repeat the test using the same sequence of random values. |

| Assignment method | Description |
|---|---|
| **Unique** | Allocates a unique block of parameter values to each Vuser in the scenario, and then sequentially assigns values to the parameter for each Vuser from within the Vuser's block of values. Ensure that there is enough data in the table for all Vusers and their iterations. If you have 20 Vusers and you want to perform 5 iterations, your table must contain at least 100 unique values.<br><br>**Note:** If you will be using Network Virtualization Insights to analyze scenario results, LoadRunner will create several additional Vusers, in addition to the Vusers that you defined in the scenario scheduler. Therefore, if the scenario includes a parameter that is assigned values using the unique data assignment method, make sure that the list of parameter values contains several extra values, i.e. several values more than is required by all Vusers to run all their iterations, in order to accommodate the additional Vusers.<br><br>If you run out of unique values, VuGen behaves according to the option you select in the **When out of values** field. For more information, see "Parameter Properties Dialog Box" on page 376.<br><br>**Note for LoadRunner users:** If a script uses Unique file parameterization, running more than one Vuser group with that script in the same scenario may cause unexpected scenario results. For more information about Vuser groups in scenarios, see the Function Reference.<br><br>• For details on the different data assignment and update methods, see "Data Assignment and Update Methods for File Parameters" below.<br>• For details on how parameters behave when the number of iterations do not match the number of values in the parameter file, see "Vuser Behavior in the LoadRunner Controller" on page 368. |

## Data Assignment and Update Methods for File Parameters

For File type parameters, the Data Assignment method that you select, together with your choice of Update method, affect the values that the Vusers use to substitute parameters during the scenario run.

The Data Assignment method is determined by the **Select next row** field, and the Update method is determined by the **Update value on** field.

The following table summarizes the values that Vusers use depending on which Data Assignment and Update properties you selected. It also includes an example of each for a table/file that has the following values: **Kim; David; Michael; Jane; Ron; Alice; Ken; Julie; Fred**

| Update Method | Data Assignment Method | | |
|---|---|---|---|
| | **Sequential** | **Random** | **Unique** |

| | | | |
|---|---|---|---|
| **Each iteration** | The Vuser takes the *next* value from the data table for each iteration.<br><br>**Example:** All the Vusers use Kim in the first iteration, David in the second iteration, Michael in the third iteration, and so on. | The Vuser takes a *new random* value from the data table for each iteration.<br><br>**Example:** The Vusers use random values from the table for each iteration. | The Vuser takes a value from the next unique position in the data table for each iteration.<br><br>**Example:** For a test run of 3 iterations, the first Vuser takes Kim in the first iteration, David in the second, and Michael in the third. The second Vuser takes Jane, Ron, and Alice. The third Vuser, Ken, Julie, and Fred. |
| **Each occurrence**<br><br>(Data Files only) | The Vuser takes the *next* value from the data table for each occurrence of the parameter, even if it is within the same iteration.<br><br>**Example:** All the Vusers use Kim in the first occurrence, David in the second occurrence, Michael in the third occurrence, and so on. | The Vuser takes a *new random* value from the data table for each occurrence of the parameter, even if it is within the same iteration.<br><br>**Example:** The Vusers use random values for each occurrence of the parameter. | The Vuser takes a *new unique* value from the data table for each occurrence of the parameter, even if it is within the same iteration.<br><br>**Example:** The Vuser uses a unique value from the list for each occurrence of the parameter. |
| **Once** | The value assigned in the first iteration is used for all subsequent iterations for each Vuser.<br><br>**Example:** All Vusers take Kim for all iterations. | The random value assigned in the first iteration is used for all iterations of that Vuser.<br><br>**Example:** All Vusers take the first randomly assigned value for all the iterations. | The unique value assigned in the first iteration is used for all subsequent iterations of the Vuser.<br><br>**Example:** The first Vuser takes Kim for all iterations, the second Vuser takes David for all iterations, and so on. |

**Note:** If you select the **Sequential** method and there are not enough values in the data table, VuGen returns to the first value in the table, continuing in a loop until the end of the test.

### Vuser Behavior in the LoadRunner Controller

When you set up a scenario to run a parameterized script, you can instruct the Vusers how to act when there are not enough values. The following table summarizes the results of a scenario using the following parameter settings:

- Select next row = **Unique**
- Update Value on = **Each iteration**
- When out of values = **Continue with last value**

| Situation | Duration | Resulting Action |
|---|---|---|
| **More iterations than values** | **Run until completion** | When the unique values are finished, each Vuser continues with the last value, but a warning message is sent to the log indicating that the values are no longer unique. |
| **More Vusers than values** | **Run indefinitely or Run for ...** | Vusers take all of the unique values until they are finished. Then the test issues an error message **Error: Insufficient records for param <param_name> in table to provide the Vuser with unique data**. To avoid this, change the **When out of values** option in the Parameter properties or the **Select next row** method in the Parameter properties. |
| **One of two parameters are out of values** | **Run indefinitely or Run for ...** | The parameter that ran out of values, continues in a cyclic manner until the values of the second parameter are no longer unique. |

## XML Parameters

When you create a Web Service call to emulate a specific operation, the arguments in the operation may include complex structures with many values. You can use an XML type parameter to replace the entire structure with a single parameter.

You can create several value sets for the XML elements and assign a different value set for each iteration.

The XML parameter type supports complex schema types such as arrays, Choice, and <Any> elements.

When working with Web Service Input Arguments, you may encounter arrays and their sub-elements. You can define a single XML parameter that will contain values for all of the array elements.

You can create new XML type parameters directly from the Insert menu, similar to all other parameter types. For Web Services type scripts, you create an XML parameter directly from the Web Services Call properties.



**Note:** For protocols using XML, replay fails to create a request when a parameterized input

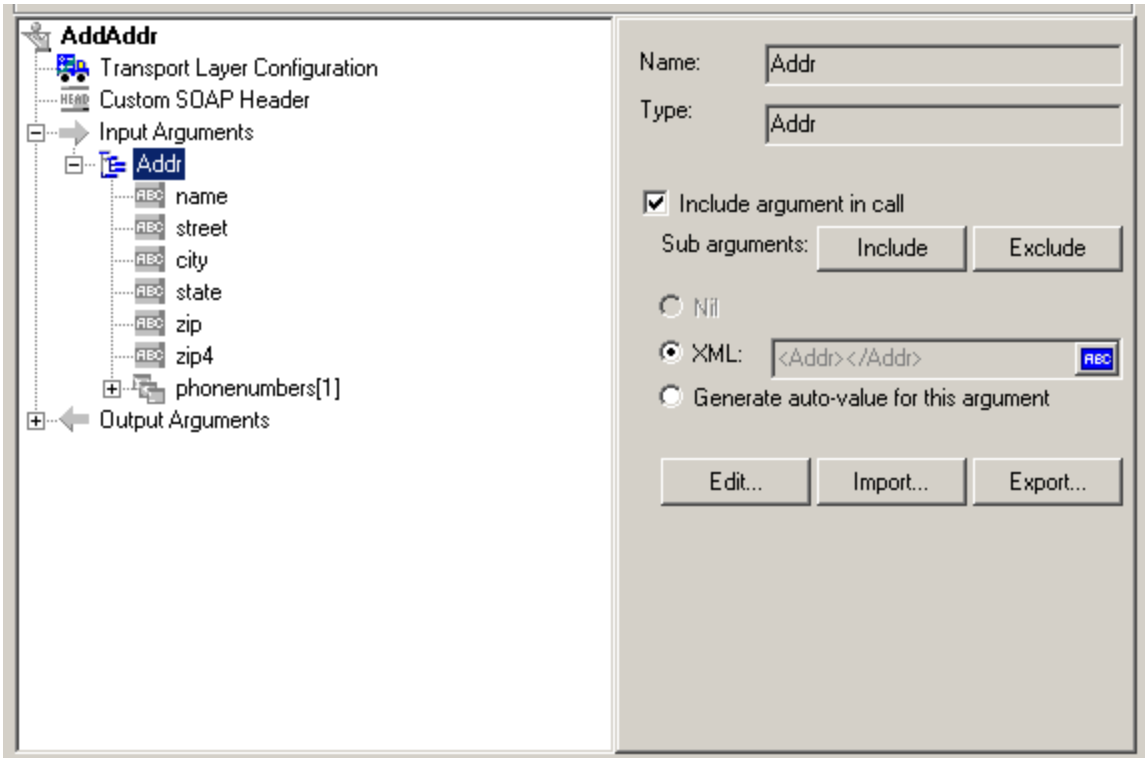
argument contains the ampersand (&) character.

## Create an XML Parameter from a Web Service Call

This task describes how to create a new XML Parameter from a Web Service Call. This procedure is in addition to the standard procedure to create a parameter. XML Parameters can also be created by using the standard procedure.

### To create an XML parameter from a Web Service Call:

1. Select the root element of the complex data structure. The right pane displays the argument's details.




2. Select **XML** in the right pane, and click the **ABC** icon. The Select or Create Parameter dialog box opens.
3. In the **Parameter name** box, enter a name for the parameter.
4. In the **Parameter type** box, select **XML** if it is not already selected.
5. Click **Properties** to assign a value set now, or **OK** to close the dialog box and assign values later.

## Create XML Parameters - Standard Method

This task describes how to create an XML type parameter without viewing the properties of a Web Service call. This is the most common way of parameterizing values for most protocols and parameter

types.

For Web Service Scripts, we recommend that you create parameters from within a Web Service Call, as described in "XML Parameters" on page 368.

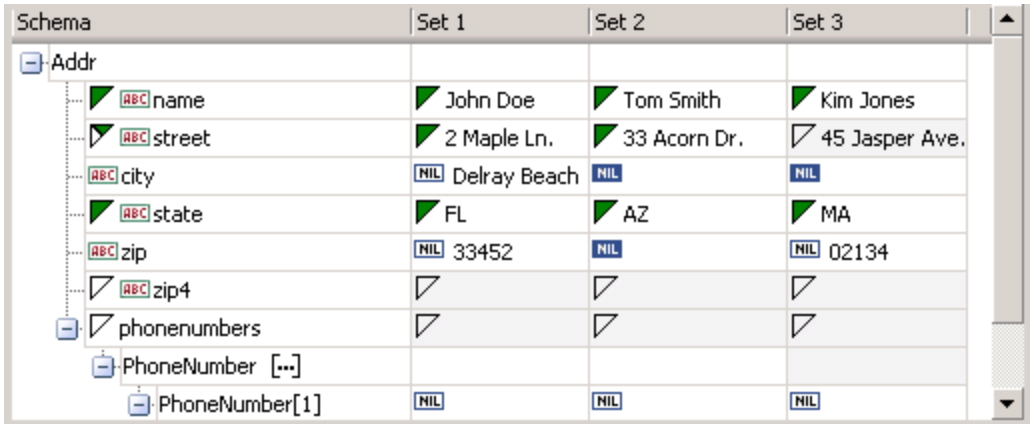
### To create a new XML parameter:

1. Select **Insert > New Parameter** or select a constant value in the Script view and select **Replace with a Parameter** from the right-click menu. The Select or Create Parameter dialog box opens.
2. In the **Parameter name** box, enter a name for the parameter.
3. In the **Parameter type** box, select **XML** if it is not already selected.
4. Click **Properties** to assign a value set now, or **OK** to close the dialog box and assign values later.

## Define XML Value Sets

This task describes how to create value sets for XML parameters.

Value sets are arrays that contain a set of values. Using the **Add Column** and **Duplicate Column** buttons, you can create multiple value sets for your parameter and use them for different iterations.

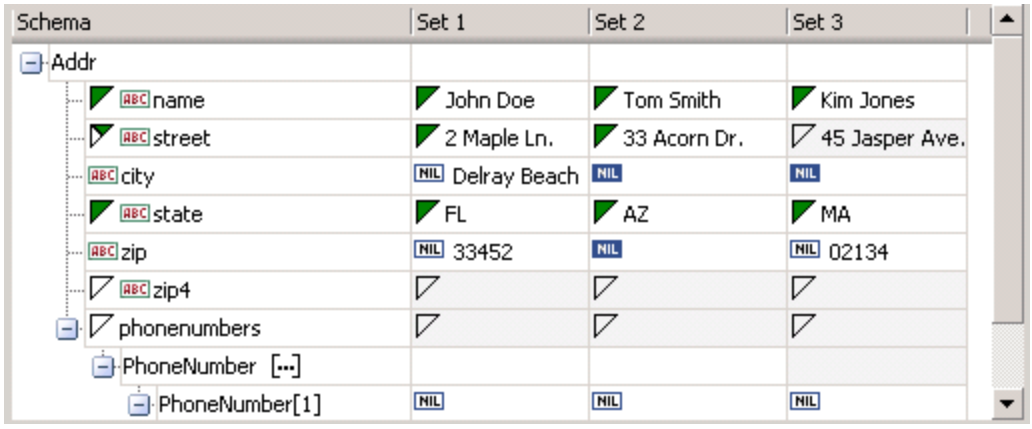



When using value sets, the number of array elements per parameter does not have to be constant.

You can use optional elements that will appear in one value set, but not in another. This allows you to vary the values you send for each of the iterations—some iterations can include specific array elements, while other iterations exclude them.

To exclude an optional element, click the small triangle in the upper left corner of the cell and insure that it is not filled in.

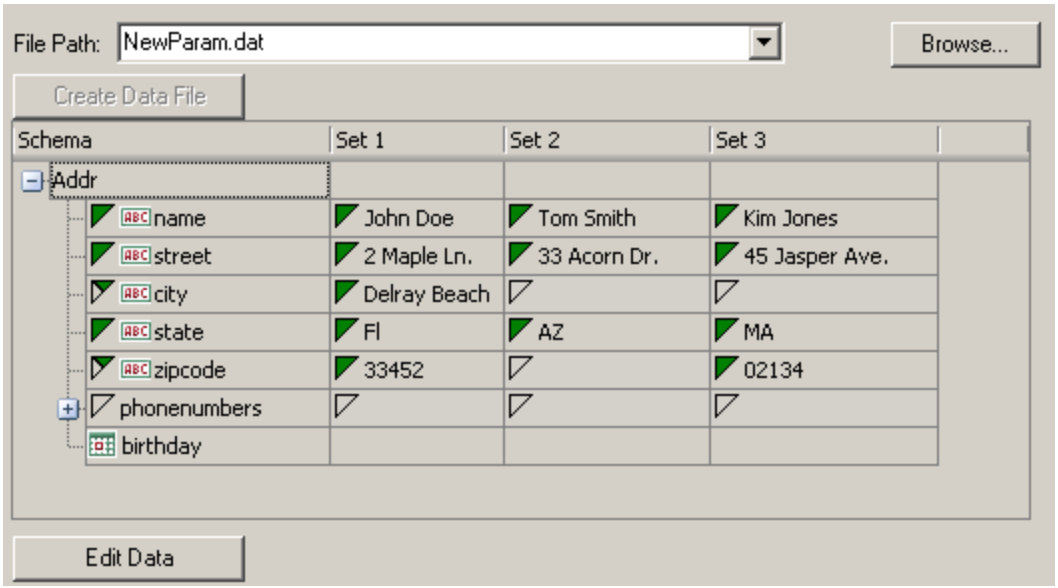
In the following example, **Set 1** and **Set 2** use the optional elements: **name**, **street**, and **state**. **Set 3** does not use a street name.




## To set parameter element values

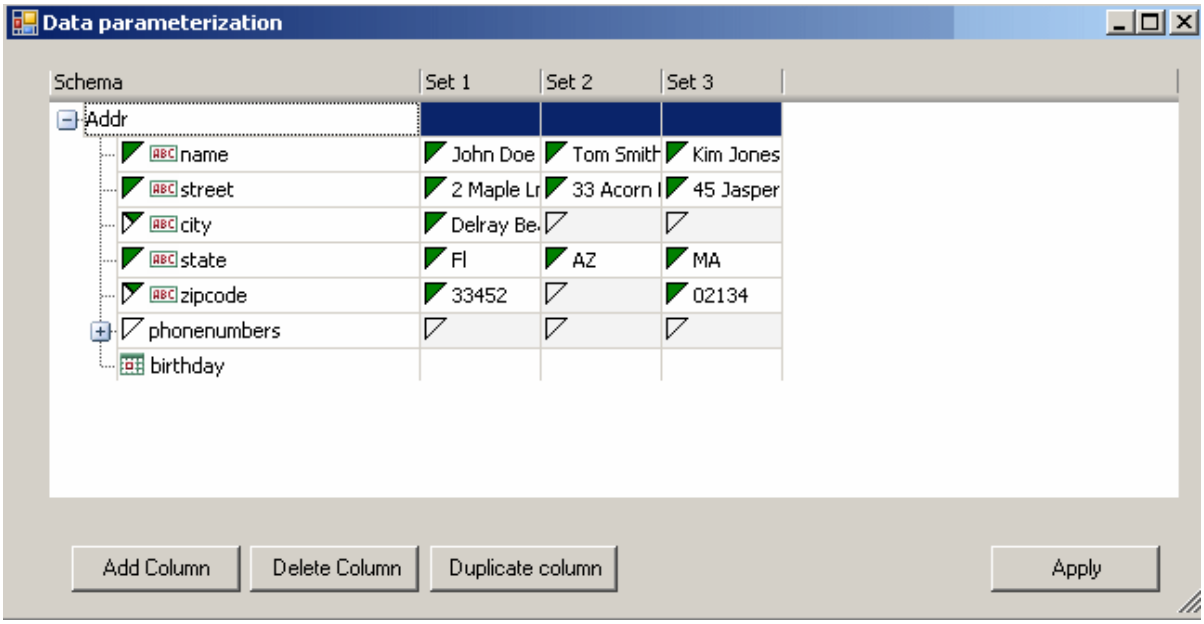
1. **View the Parameter Properties.**

   If the Parameter Properties dialog box is not open, select **Vuser > Parameter List** and select the desired parameter. The dialog box shows a read-only view of the parameter values.




2. **Open the Data Parameterization box.**

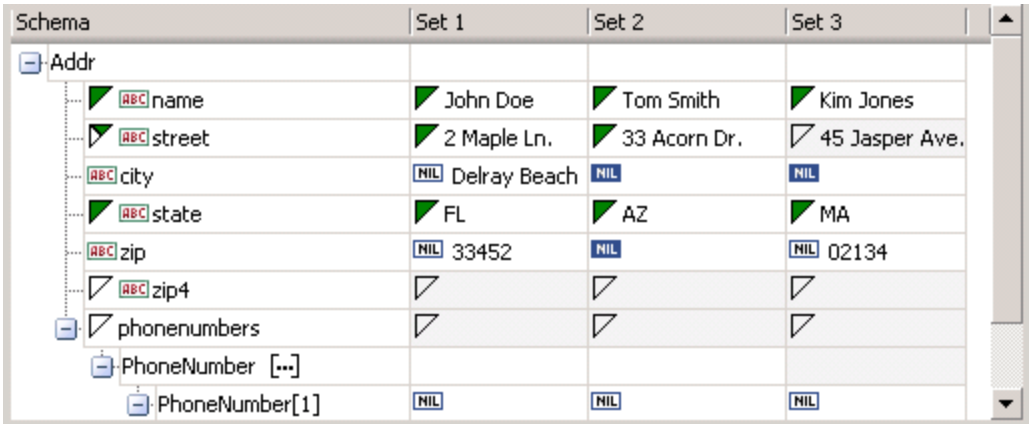
   Click the **Edit Data** button to open the Data Parameterization dialog box.




3. **Define value sets for the XML parameter.**

   In the **Set** columns, insert values corresponding to the schema.

   If a row says **NIL**, it implies that the element is nillable. To include a value for the nillable element, enter the value as usual. To mark a value as **nil**, click the NIL icon to fill it in. This erases any value that you may have assigned to the element. In the following example, the **city** element is nillable, but it is only marked as nil in **Set 2** and **Set3**—not in **Set 1**.

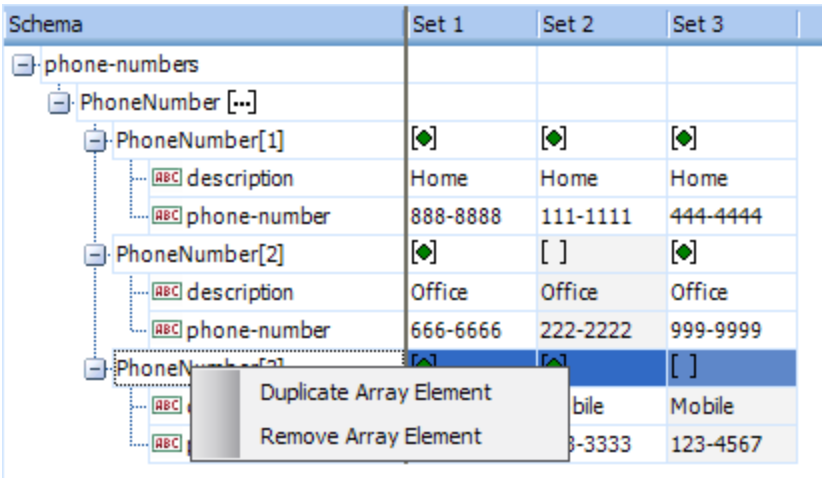



4. **Create additional value sets.**

   To insert more value sets, click **Add Column** and insert another set of values in the new column. To copy an existing value set, select a row in the value set you want to copy and click **Duplicate Column**.

5. **Copy arrays.**

   To duplicate an array element and its children, select the parent node and choose **Duplicate Array Element** from the right-click menu.

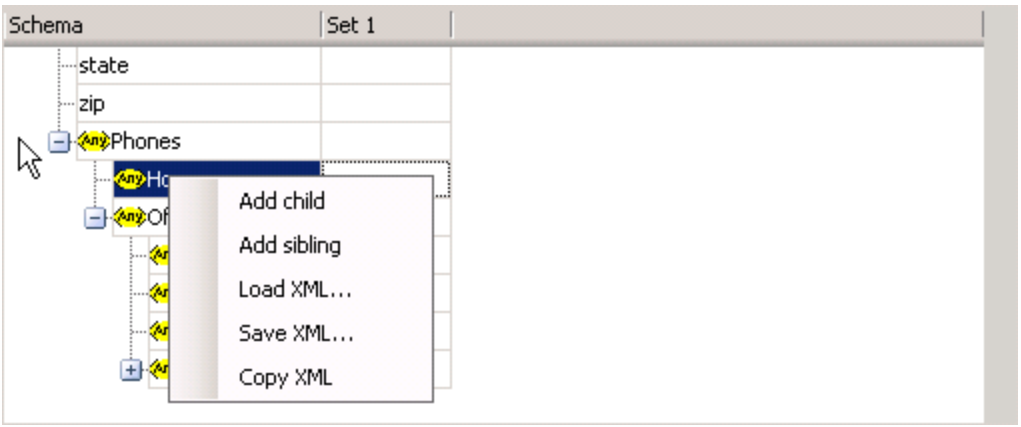



6. **Handle the <any> elements.**

   For **any** type elements, right-click **<any>** in the **Schema** column and select one of the available options. These options may vary depending on the location of the cursor.

   - **Add Array Element.** Adds a sub-element under the root element.
   - **Insert child.** Adds a sub-element to the selected element.
   - **Insert sibling.** Adds a sub-element on the same level as the selected element.
   - **Load XML.** Loads the element values from an XML file.
   - **Save XML.** Saves the array as an XML file.
   - **Copy XML**. Copies the full XML of the selected element to the clipboard.

   Click the **Rename** text to provide a meaningful name for each array element.




7. **Remove unwanted columns.**

   To remove a value set, select it and click **Delete Column**.

8. **Save the changes.**

   Click **Apply** to save the changes and update the view in the Parameter Properties dialog box.

# Set an Assignment Method

This task describes how to set an assignment method. The assignment method indicates which of the value sets to use and how to use them. For example, you can instruct Vusers to use a new value set for each iteration and use the value sets sequentially or at random. For more information, see "Data Assignment and Update Methods for File Parameters" on page 366.

## To define an assignment method

1. Open the Parameter Properties and select a parameter.
2. Define a data assignment method.

   In the **Select next value** list, select a data assignment method to instruct the Vuser how to select the file data during Vuser script execution. The options are: **Sequential**, **Random**, or **Unique**. For more information, see "Data Assignment Methods for File-Type Parameters" on page 365.

3. Select an update option for the parameter.

   In the **Update value on** list, select an update option. The choices are **Each Iteration**, **Each Occurrence**, and **Once**. For more information, see "Data Assignment and Update Methods for File Parameters" on page 366.

4. If you chose **Unique** as the data assignment method, the **When out of values** and **Allocate Vuser values in the Controller** options become enabled.
   - **When out of values.** Specify what to do when there is no more unique data: **Abort Vuser**, **Continue in a cyclic manner**, or **Continue with last value**.
   - **Allocate Vuser values in the Controller.** (for LoadRunner users only) Indicate how to allocate data blocks of parameter values to the Vusers. You can allow the Controller to automatically allocate a block size or you can specify the block size for each Vuser.
     - **Automatically allocate block size.** The block size is calculated by dividing the number of parameter values by the number of Vusers.
     - **Allocate** x **values for each Vuser.** Specify the number of values to allocate to each Vuser.

   To track this occurrence, enable the **Extended Log > Parameter Substitution** option in the Log Runtime Settings. When there are not enough parameter values, VuGen writes a warning message to the Vuser log: **No more unique values for this parameter in table <table_name>**.

5. In the Parameter Properties dialog box, click **Close**.

   The list of input arguments is replaced by the parameter name, and ABC button is replace by a table icon which you can click to edit the parameter properties or un-parameterize the parameter.

# Modify XML Parameter Properties

This task describes how to modify XML parameter properties.

To modify XML parameter properties:

1. In the Step Navigator, right-click the required step and select **Show Arguments**. The Web Service Call Properties dialog box opens.
2. From the list of arguments, under **Input Arguments**, select the XML parameter. The right pane displays the parameter details.
3. To modify the XML parameter properties, select the **XML** check box, click the table icon button, and then select **Parameter Properties**. The Parameter Properties dialog box opens.
4. Modify the parameter properties as desired.

## Set AUT Environment Parameters

When working with scripts stored in Application Lifecycle Management (ALM), you can work with different Application Under Test (AUT) environments.

AUT Environments are environments that you define in ALM that represent different testing configurations. By parameterizing the environment data, you can make your test more flexible and portable. Instead of running several scripts that use the same logic, but with different AUT environment constants, you can maintain a single script that uses AUT environment parameters.

You define environment-specific parameters in ALM's AUT Environment configuration. During the test run, ALM inserts these values into your script. For more details on working with AUT environments, refer to the *Application Lifecycle Management User Guide*.

This task describes how to define an AUT environment type parameter in your Vuser script.

1. Create a parameter in VuGen. Make sure the name of the parameter matches the name of the corresponding AUT environment parameter.
2. Set the parameter type to "**Custom parameter**".
3. Enter the following parameter description "ALMPARAM" , using the exact spelling and case.

## Select or Create Parameter Dialog Box

This dialog box enables you to create a new parameter or modify an existing parameter.

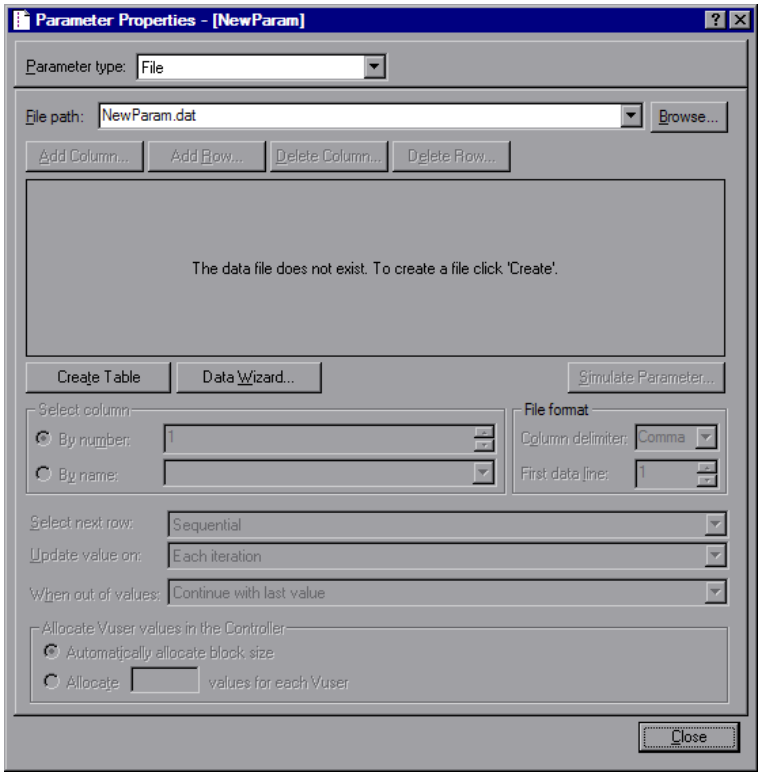
| | |
|---|---|
| **To access** | Use one of the following:<br>• **VuGen > Solution Explorer** pane > right-click on the **Parameters** node > **Create New Parameter**<br>• In script editor, right-click on the value > **Replace with Parameter** > **Create New Parameter**<br>• **Design > Parameters > Create New Parameter** |
| **Relevant tasks** | "Create Parameters" on page 361 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Parameter name** | The name of the parameter.<br>**Note:** Do not use the name **unique**, it is used by VuGen. |
| **Parameter type** | The type of the parameter. For information about the different parameter types see "Parameter Types" on page 359. |
| **Original value** | The original value of the parameter before parameterization. |
| Properties... | Opens the Parameter Properties dialog box. For details, see "Parameter Properties Dialog Box" below. |

## Parameter Properties Dialog Box

This page allows you to view and modify the properties of a parameter. This dialog box varies depending on the type of parameter you are using.




| To access | **VuGen > Right-click parameter > Parameter properties** |
|---|---|

## Date/Time, Group Name, Iteration Number, Load Generation Name, and Vuser ID Parameters

User interface elements are described below:

| UI Element | Description |
|---|---|
| Add format -> | Adds the custom format specified in the **Date/time format** or **Text format** field to the format list. |
| Delete format <- | Deletes the selected format from the format list. |
| Reset formats | Restores the format list to it's default state. |
| **Date/time format / Text format** | You can specify a custom format here. See the chart below for a list of Date/time symbols. |
| **Format list** | The list of formats. See the chart below for a list of Date/time symbols. |
| **Offset (Date/time to type only)** | Allows you to set an offset for the date/time parameter. For example, if you want to test a date next month, you set the date offset to 30 days.<br>• **Working days only.** Use values for work days only (excludes Saturdays and Sundays).<br>**Note:** To change the non-working days, configure the NonWorkingDays parameter under the Misc section in the vugen.ini file:<br>`[Misc]`<br>`NonWorkingDays="5,6"`<br>Days are represented by integers as follows:<br>Mon = 1, Tue = 2, Wed = 3 ,Thur = 4, Fri = 5, Sat = 6, Sun = 7<br>• **Prior to current date.** Sets the offset for a date or time that has already passed (negative offset). |
| **Parameter type** | The parameter type. For more information see "Parameter Types" on page 359. |
| **Sample (current time)** | Displays an example parameter value based on the selected format. |

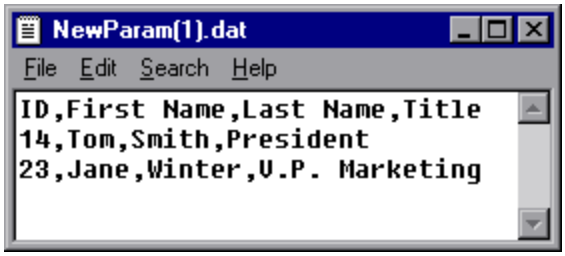
| UI Element | Description |
|---|---|
| **Update values on** | • **Each occurrence.** Use a new value for each occurrence of the parameter in your script. This is useful when the statements using a parameter are unrelated. For example, for random data, it may be useful to use a new value for each occurrence of the parameter.<br>• **Each iteration.** Updates the parameter one time per iteration. If a parameter appears in a script several times, the Vuser uses the same value for all occurrences of the parameter, for the entire iteration. This is useful when the statements using a parameter are related.<br>**Note:** If you create an action block with parameters using its own iteration count—if you instruct VuGen to update their values each iteration, it refers to the global iteration and not the block iteration.<br>• **Once.** Updates the parameter value only once during the scenario run. The Vuser uses the same parameter value for all occurrences and all iterations of the parameter. This type may be useful when working with dates and times. |

The following table describes the date/time symbols:

| Symbol | Description |
|---|---|
| c | complete date and time in digits |
| #c | complete date as a string and time |
| H | hours (24 hour clock) |
| I | hours (12 hour clock) |
| M | minutes |
| S | seconds |
| p | AM or PM |
| d | day |
| m | month in digits (01-12) |
| b | month as a string - short format (e.g. Dec) |
| B | month as a string - long format (e.g. December) |
| y | year in short format (e.g. 03) |
| Y | year in long format (e.g. 2013) |

## File Parameters

User interface elements are described below:

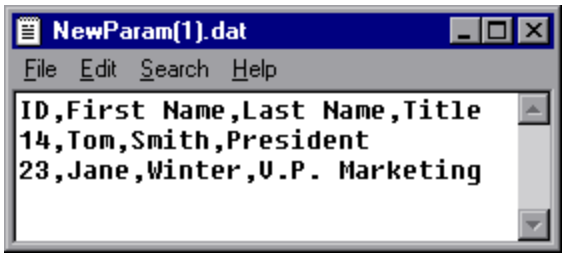
| UI Element | Description |
|---|---|
| Add Column... | Adds a column to the data set. |
| Add Row... | Adds a row to the data set. |
| Create Table | Creates a new data table. |
| Import Parameter... | Opens the Import Parameter Values from File dialog box, enabling you to import parameter values from an ASCII file. For more information, see "Import Parameter Values from a File" on page 362. |
| Delete Column... | Deletes a column from the data set. |
| Delete Row... | Deletes a row from the data set. |
| Edit with Notepad... | Enables you to view and edit parameter values in Notepad. This is important when working with large data sets because VuGen displays only up to 100 rows in the UI.<br>Notepad opens with the parameter's name in the first row and its original value in the second row. Enter additional column names and values into the file using a delimiter such as a comma or a tab to indicate a column break. Begin a new line for each table row (for each new row of data).<br>  |
| Simulate Parameter... | Opens the Parameter Simulation dialog box. This allows you to simulate the parameter behavior with your data set. For more information, see "Parameter Simulation Dialog Box" on page 385. |
| **Select Column** | Enables you to select the column to use as the data source, either by the column **number** or **name**. |

| UI Element | Description |
|---|---|
| **File Format** | • **Column delimiter.** The character used to separate values in the data file.<br>• **First data line.** The first line of data to be used during Vuser script execution. The header is line 0. To begin with the first line after the header, specify 1. If there is no header, specify 0. |
| **Select next row** | The method of selecting the file data during Vuser script execution. The options are: **Sequential**, **Random**, or **Unique**. For more information see "Data Assignment Methods for File-Type Parameters" on page 365. |
| **Update value on** | The method that determines when the parameter will switch to the next value. The choices are **Each Iteration**, **Each Occurrence**, and **Once**. For more information see "Data Assignment Methods for File-Type Parameters" on page 365. |
| **When out of values** | Specify what to do when there is no more unique data: **Abort the Vuser**, **Continue in a cyclic manner**, or **Continue with last value**. |
| **Allocate Vuser values in the Controller** | (LoadRunner only). Indicate how to allocate data blocks of parameter values to the Vusers. You can allow the Controller to automatically allocate the block size or you can specify the desired block size to allocate to each Vuser.<br>• **Automatically allocate block size.** The block size is calculated by dividing the number of parameter values by the number of Vusers.<br>• **Allocate** x **values for each Vuser.** Specify the number of values to allocate to each Vuser.<br>To track this occurrence, enable the **Extended Log > Parameter Substitution** option in the Log Runtime Settings. When there are not enough parameter values, VuGen writes the following warning message to the Vuser log: "No more unique values for this parameter in table <*table_name*>". |
| **File path** | Select the .dat file with the data for your parameter. Alternatively, you can create a new data set using the **Create Table** button. |

## Table Parameters

User interface elements are described below:

| UI Element (A-Z) | Description |
|---|---|
| | Adds a column to the data set. |

| UI Element (A-Z) | Description |
|---|---|
| Add Row... | Adds a row to the data set. |
| Create Table | Creates a new data table. |
| Import Parameter... | Opens the Import Parameter Values from File dialog box, enabling you to import parameter values from an ASCII file. For more information, see "Import Parameter Values from a File" on page 362. |
| Delete Column... | Deletes a column from the data set. |
| Delete Row... | Deletes a row from the data set. |
| Edit with Notepad... | Enables you to view and edit parameter values in Notepad. This is important when working with large data sets because VuGen only displays up to 100 rows in the UI.<br>Notepad opens with the parameter's name in the first row and its original value in the second row. Enter additional column names and values into the file using a delimiter such as a comma or a tab to indicate a column break. Begin a new line for each table row (for each new row of data).<br>  |
| **Allocate Vuser values in the Controller** | (LoadRunner only). Indicate how to allocate data blocks of parameter values to the Vusers. You can allow the Controller to automatically allocate the block size or you can specify the desired block size to allocate to each Vuser.<br>• **Automatically allocate block size.** The block size is calculated by dividing the number of parameter values by the number of Vusers.<br>• **Allocate** x **values for each Vuser.** Specify the number of values to allocate to each Vuser.<br>To track this occurrence, enable the **Extended Log > Parameter Substitution** option in the Log Runtime Settings. When there are not enough parameter values, VuGen writes a warning message to the Vuser log "No more unique values for this parameter in table <*table_name*>". |

| UI Element (A-Z) | Description |
|---|---|
| **Column** | The columns to use. Alternatively, you can select **Select all columns.**<br>To specify one or more columns by their number, select **Columns by number** and enter the column numbers separated by a comma or a dash. The column number is the index of the column containing your data. For example, if the data for the parameter is in the table's first column, select 1.<br>In the **Column delimiter** box, select a column delimiter—the character used to separate the columns in the table. The available delimiters are: comma, tab, space. |
| **File path** | Select the .dat file with the data for your parameter. Alternatively, you can create a new data set using the **Create Table** button. |
| **Row delimiter for log display** | This delimiter is used to differentiate between rows in the output logs. If you enable parameter substitution logging, VuGen sends the substituted values to the Replay log. The row delimiter character in the Replay log indicates a new row. |
| **Rows** | • **Rows per iteration.** How many rows to use per iteration. This only relevant when the **Update value on** field is set to **Each iteration**. If **Update value on** is set to **Once**, then the same rows will be used for all iterations.<br>• **First line of data.** The first line of data to be used during script execution. To begin with the first line after the header, enter 1.<br>• Table information. Displays information about the table, including how many rows of data are available. |
| **Select next row** | The method of selecting the file data during Vuser script execution. The options are: **Sequential**, **Random**, or **Unique**. For more information see "Data Assignment Methods for File-Type Parameters" on page 365. |
| **Update value on** | The method that determines when the parameter will switch to the next value. The choices are **Each Iteration**, **Each Occurrence**, and **Once**. For more information see "Data Assignment Methods for File-Type Parameters" on page 365. |
| **When not enough rows** | Specifies what VuGen does when there are not enough rows in the table for the iteration.<br>**Example:** The table you want to fill has 3 rows, but your data only has two rows. Select **Parameter will get less rows than required** to fill in only two rows. Select **Use behavior of "Select Next Row"** to loop around and get the next row according the method specified in the **Select Next Row** box. |
| **When out of values** | Specify what to do when there is no more unique data: **Abort the Vuser**, **Continue in a cyclic manner**, or **Continue with last value**. |

## Random Number Parameters

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Number format** | Specifies the minimum number of digits your parameter will have. Where **%01d** represents one digit, **%02d** represents two digits, and so on. |
| **Random range** | The minimum and maximum range for the random values. |
| **Sample value** | Displays sample minimum and maximum values based on the selected Number format. |
| **Update value on** | • **Each occurrence.** Use a new value for each occurrence of the parameter in your script. This is useful when the statements using a parameter are unrelated. For example, for random data, it may be useful to use a new value for each occurrence of the parameter.<br>• **Each iteration.** Updates the parameter one time per iteration. If a parameter appears in a script several times, the Vuser uses the same value for all occurrences of the parameter, for the entire iteration. This is useful when the statements using a parameter are related.<br>**Note:** If you create an action block with parameters using its own iteration count—if you instruct VuGen to update their values each iteration, it refers to the global iteration and not the block iteration.<br>• **Once.** Updates the parameter value only once during the scenario run. The Vuser uses the same parameter value for all occurrences and all iterations of the parameter. This type may be useful when working with dates and times. |

## Unique Number Parameters

**Note:** When scheduling a scenario in the Controller, the **When out of values** option only applies to the **Run for HH:MM:SS** option in the Schedule Builder's Duration tab. It is ignored for the **Run until completion** option.

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Number format** | Specifies the minimum number of digits your parameter will have. Where **%01d** represents one digit, **%02d** represents two digits, and so on. |

| UI Element | Description |
|---|---|
| **Number range** | • **Start.** The starting value.<br>• **Block size per Vuser.** The amount of unique numbers assigned to each Vuser. For example, if you specify a starting value of 1 and a block size of 100, the values be 1-100 can be used by the first Vuser, the values 101-200 can be used by the second Vuser, and so on. |
| **Sample value** | Displays an example parameter value based on the selected format. |
| **Update value on** | • **Each occurrence.** Use a new value for each occurrence of the parameter in your script. This is useful when the statements using a parameter are unrelated. For example, for random data, it may be useful to use a new value for each occurrence of the parameter.<br>• **Each iteration.** Updates the parameter one time per iteration. If a parameter appears in a script several times, the Vuser uses the same value for all occurrences of the parameter, for the entire iteration. This is useful when the statements using a parameter are related.<br>**Note:** If you create an action block with parameters using its own iteration count—if you instruct VuGen to update their values each iteration, it refers to the global iteration and not the block iteration.<br>• **Once.** Updates the parameter value only once during the scenario run. The Vuser uses the same parameter value for all occurrences and all iterations of the parameter. This type may be useful when working with dates and times. |
| **When out of values** | Determines what to do when the range of values is reached for a Vuser. The range of values is determined by the start value and the block size.<br>**Abort Vuser.** Terminates the Vuser script.<br>**Continue in a cyclical manner.** Restart the unique numbers for this Vuser from the beginning of its assigned range. For example, if a Vuser had the range of 1-100 and it reached 100, it would start again at 1.<br>**Continue with last value.** Use the last assigned value for this parameter for all subsequent occurrences of this parameter. For example, if a Vuser had the range of 1-100 and it reached 100, it would continue with the value of 100 until the end of the script. |

## User Defined Function Parameters

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Function Name** | The name of the function. Use the name of the function as it appears in the DLL file. |
| **Library Names** | The location of the relevant library files. |
| **Update value on** | • **Each occurrence.** Use a new value for each occurrence of the parameter in your script. This is useful when the statements using a parameter are unrelated. For example, for random data, it may be useful to use a new value for each occurrence of the parameter.<br>• **Each iteration.** Updates the parameter one time per iteration. If a parameter appears in a script several times, the Vuser uses the same value for all occurrences of the parameter, for the entire iteration. This is useful when the statements using a parameter are related.<br>**Note:** If you create an action block with parameters using its own iteration count—if you instruct VuGen to update their values each iteration, it refers to the global iteration and not the block iteration.<br>• **Once.** Updates the parameter value only once during the scenario run. The Vuser uses the same parameter value for all occurrences and all iterations of the parameter. This type may be useful when working with dates and times. |

### XML Parameters

For information about Web Services XML parameters, see "XML Parameters" on page 368.

## Parameter Simulation Dialog Box

This dialog box allows you to view a simulation of the behavior of a file parameter.

| To access | **VuGen > Parameter List > Select Parameter > Simulate Parameter** |
|---|---|

| | |
|---|---|
| **Important information** | • This feature is only relevant for file type parameters.<br>• Not all types of Parameter Substitution can be simulated. If you select **Select next row: Same line as...** or **Update value on: Each occurrence**, then the Parameter Simulation dialog box will not open.<br>• VuGen can simulate up to 256 iterations and 256 Vusers.<br>• Maximum parameter value length: 100 characters.<br>• **Run Indefinitely** is compliant with the **Real-life schedule** in the Scheduler of the Controller.<br>• If you select **Select next row: Unique** in the Parameter List dialog, then each Vuser is assigned a unique range of rows from which the Simulator will substitute values (for that Vuser).<br>**Note:** If you have more than one unique parameter, you need to verify that each parameter has defined values for all Vusers.<br>With this setting, the default selection in the **Allocate Vuser values in the Controller** section is **Automatically allocate block size**. In this case, when you run the simulation, the range allocation takes place in accordance with your Scenario run mode selection. If you change the default selection to **Allocate x values for each Vuser**, then the Vusers will be allocated the amount of values you specify, ignoring of your Scenario run mode selection. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Vusers** | The number of Vusers to run in the simulation. |
| **Scenario Run Mode** | • **Run until completion.** Enter the number of iteration to run or select **Take number of iteration from Runtime Settings**.<br>• **Run indefinitely.** Simulates the run indefinitely option in the controller. VuGen only actually simulates the number of iterations you specify. |
| Simulate | Runs the parameter simulation. The values of each parameter substitution are displayed. |

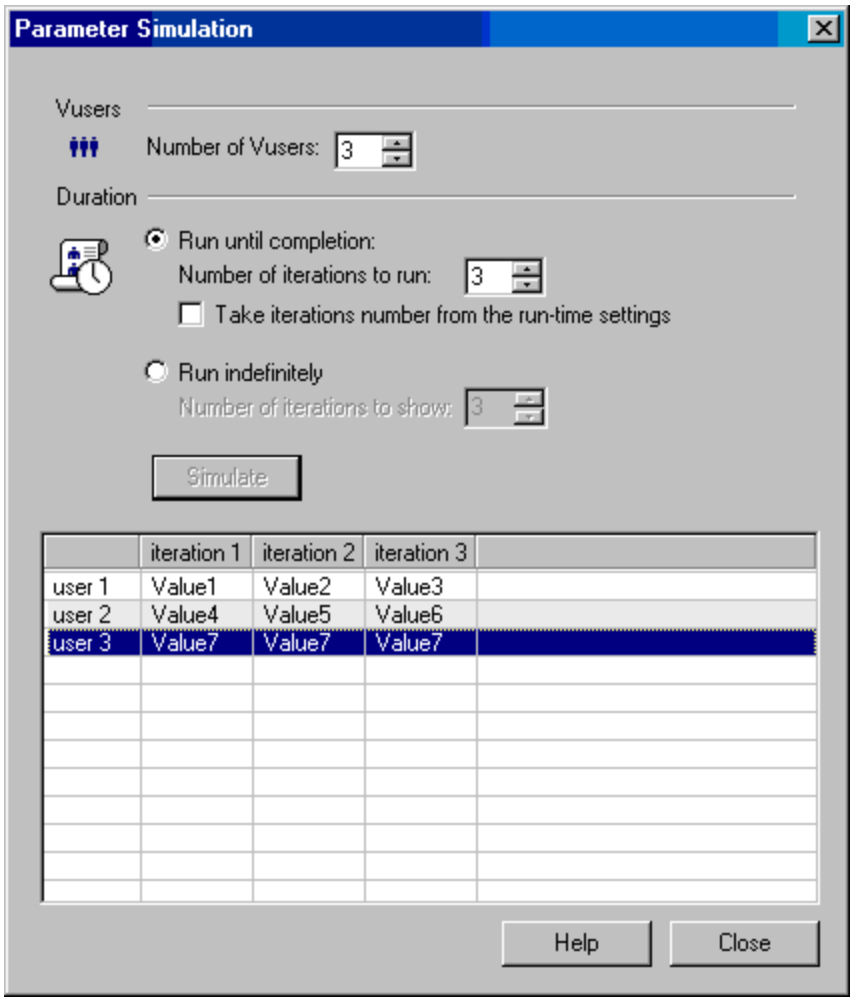
**Example:**

In the following examples, the settings in the Parameter List dialog box are:

- **Values for the new parameter**. Value1 to Value7
- **Select next row**. Unique
- **When out of rows**. Continue with last value
- **Allocate Vuser values in the Controller**. Automatically allocate block size

**Scenario run mode: Run until completion**

In the following example, the user has selected three Vusers, set the Scenario run mode to **Run until completion**, and selected three iterations.



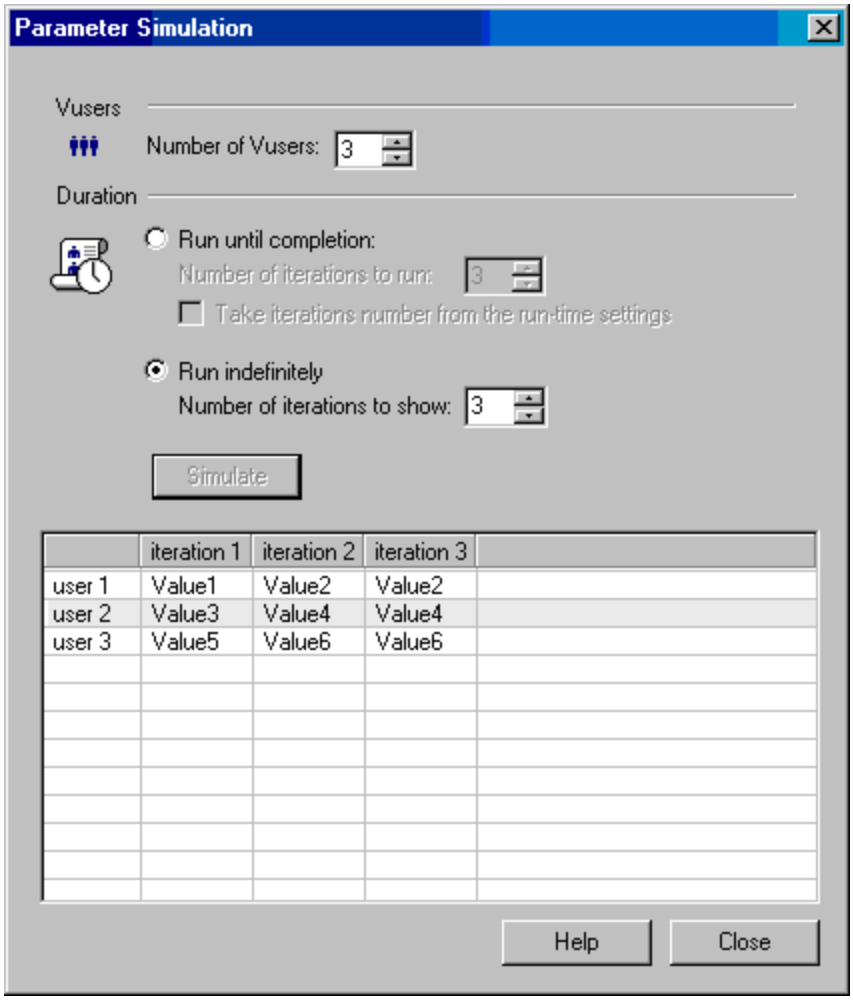

When the scenario run mode is set to **Run until completion**, the number of rows that each Vuser receives is the same as the number of iterations. The range allocation stops when there are no longer enough rows in the table.

As the simulation is run, the first Vuser takes the first three values (because this was the number of iterations). The second Vuser takes the next three values. The third Vuser takes the remaining value in the first iteration. For the remaining iterations, since the **When out of values** option in the Parameter List dialog box was set to **Continue with last value,** the third Vuser continues with the same value.

A fourth Vuser would have failed.

**Scenario run mode: Run indefinitely**

In the following example, the user has selected 3 Vusers and set the Scenario run mode to Run indefinitely and selected to show 3 iterations.




When the Scenario run mode is set to Run indefinitely, the allocated range for each Vuser is calculated by dividing the number of cells in the .dat file by the number of Vusers. In this scenario, that is 7/3 = 2 (The simulator takes the closest smaller integer.).

As the simulation is run, the first Vuser takes Value1 and Value2. The second Vuser takes Value3 and Value4 and the third Vuser takes Value5 and Value6. Since there are were only 3 Vusers, Value7 was not distributed.

**Note:** If you hold the mouse over the cells in the first column of the table, a tool tip appears with information about which values were assigned to that Vuser.

If you hold the mouse over cells which were not assigned values, a tool tip appears with the reason no values were assigned.

A tool tip does not appear if a proper value was assigned.

## Parameter List Dialog Box

This dialog box enables you to view, create, delete, select, and modify parameters. The Parameter list shows all of the parameters that you created, including both input and output parameters.

| | |
|---|---|
| **To access** | Use one of the following:<br>• **VuGen > Solution Explorer** pane > **Parameters** node > **Parameter List**<br>• In the script editor, right-click on a value > **Replace with Parameter**> **Parameter List**<br>• **Design > Parameters > Parameter List** |
| **Important information** | Do not name a parameter *unique*, since this name is used by VuGen. |
| **See also** | "Parameter Properties Dialog Box" on page 376 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| New | Creates a new parameter. This does not replace any highlighted text with the parameter. Do not name a parameter *unique*, since this name is used by VuGen. |
| Delete | Deletes the selected parameter.<br>**Note:** If the parameter replaced a previous value, the original value is restored. |
| **<Parameter Properties Pane>** | This pane appears different depending on the type of parameter you are using. For information about this pane, see the relevant section in the "Parameter Properties Dialog Box" on page 376. |
| **Parameter type** | This drop-down list lets you select the parameter type. For information about the different parameter types, see "Parameter Types" on page 359. |

## Create Parameter Dialog Box

This dialog box enables you to create parameters directly from the Snapshot view.

| | |
|---|---|
| **To access** | In VuGen:<br>1. After recording a script, show the Snapshot pane: **View > Snapshot**.<br>2. Click the **Recording** button to show the Recording snapshot.<br>3. Select the string you want to parameterize.<br>4. Select **Create Parameter** from the right-click menu. |

| | |
|---|---|
| **Important Information** | This dialog box is only available for protocols with snapshots, such as Web HTTP/HTML. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Parameter Name** | The name of the parameter. |
| **Selected Value** | The string that will be substituted by a parameter. |
| **Left Boundary** | The left boundary of the string to define as a parameter. |
| **Right Boundary** | The right boundary of the string to define as a parameter. |

## Parameter Original Value Dialog Box

This dialog box appears when you are parameterizing a Vuser script, and lets you specify the parameter's original value.

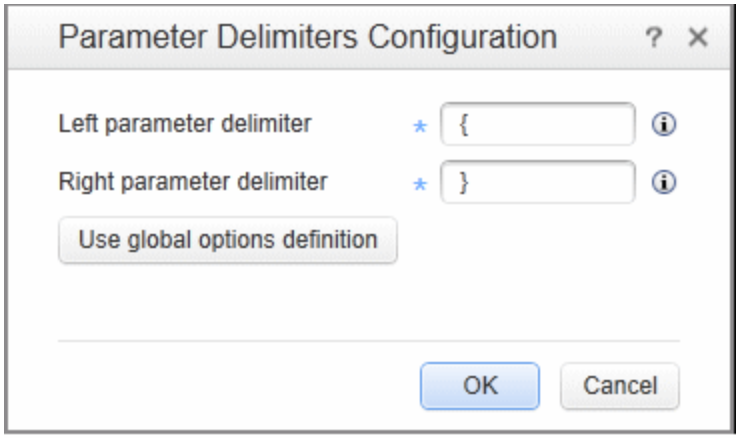
| | |
|---|---|
| **To access** | In VuGen, select a text string in the Editor. Right-click and then select **Replace with Parameter > Parameter List**. Select an existing parameter and click **Close**. |
| **Important information** | Each parameter has an original value. You can replace any parameter in a Vuser script with the parameter's original value. When you parameterize a selected text string and the selected text string is not the same as the selected parameter's original value, you can select to either keep the parameter's existing original value, or replace the parameter's original value with the selected text string. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Use old original value <text>** | Keeps the parameter's existing original value. |
| **Use new original value <text>** | Assigns the selected text as the parameter's new original value. |

## Parameter Delimiters Configuration Dialog Box

This dialog box enables you to define the delimiter type used to enclose the selected parameter.




| | |
|---|---|
| **To access** | Use one of the following:<br>• **VuGen > Solution Explorer** pane > right-click on the **Parameters** node > **Configure Parameter Braces**<br>• **Design > Parameters** > **Configure Parameter Braces** |
| **Important information** | You can enter any character to be used as the left and right delimiters. The characters do not have to be identical. After you have defined the characters, subsequent scripts use the characters defined for right and left delimiter. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Left parameter brace** | The delimiter used at the left of the parameter.<br>**Note:** The left parameter brace does not have to be the same as the right parameter brace. |
| **Right parameter brace** | The delimiter used at the right of the parameter.<br>**Note:** The left parameter brace does not have to be the same as the right parameter brace. |
| **Use global options definition** | This option reverts the selected delimiter to {. |

## Troubleshooting and Limitations for Parameterization

This section describes troubleshooting and limitations for parameters.

## Function Argument Limitations

You can use parameterization only for the arguments within a function. You cannot parameterize text strings that are not function arguments. In addition, not all function arguments can be parameterized. For details on which arguments you can parameterize, see the Function Reference (**Help > Function Reference**) for each function.

For example, consider the **lrd_stmt** function. The function has the following syntax:

```
lrd_stmt (LRD_CURSOR FAR *mptCursor, char FAR *mpcText, long mliTextLen, LRDOS_INT4
mjOpt1, LRDOS_INT4 mjOpt2, int miDBErrorSeverity);
```

The indicates that you can parameterize only the *mpcText* argument.

A recorded **lrd_stmt** function could look like this:

```
lrd_stmt(Csr4, "select name from sysobjects where name =\"Kim\" ", -1, 148, -99999,
0);
```

You could parameterize the recorded function to look like this:

```
lrd_stmt(Csr4, "select name from sysobjects where name =\"<name>\" ", -1, 148, -
99999, 0);
```

> **Note:** You can use the **lr_eval_string** function to "parameterize" a function argument that you cannot parameterize by using standard parameterization. In addition, you can use the **lr_eval_string** function to "parameterize" any string in a Vuser script.

For the COM and Microsoft .NET protocols, use only the **lr.eval string** function to define a parameter. For example:

```
lr.eval_string("{Custom_param}").
```

For more information on the **lr_eval_string** function, see the Function Reference.

### Data Table File Size Limitations

If a .dat file's size is over 100MB, a message is displayed that the file is too big and will not be displayed.

If you need to load a file over 100MB, you can change "MaxParametersDisplaySize" parameter in vugen.ini:

```
[ParamTable]

MaxParametersDisplaySize=100000000
```

# Asynchronous Communication

## Synchronous and Asynchronous Concepts

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

Originally, web applications communicated using conversations that had a synchronous nature. A typical synchronous conversation includes the following steps:

1. The user interacts with an application that is presented in a web browser.
2. Based on the user input, the browser submits a request to the web server.
3. The server sends a response to the request, and the application in the browser is updated.

Synchronous applications have a number of limitations. One limitation involves the updating of the data that is displayed in the application inside the browser. For example, consider an application that displays stock prices of a number of shares. Ideally, the application should be able to update the display of the stock prices as soon as the prices are updated on the web-server. A synchronous application would be able to update the prices on a fixed time interval.

> **Example:** Every 10 seconds, the browser could sent a request to the server for the most up-to-date stock prices.

One limitation of this solution is that the displayed stock prices may be out-of-date for a period of time before the refresh interval is reached. Although this may not be critical in our share portfolio scenario, the scenario illustrates the limitation of a synchronous application to timeously update information.

Where necessary, synchronous applications are being replaced with what are known as *asynchronous* applications. Asynchronous applications enable a client to be notified whenever an event occurs on the server side. Asynchronous applications are therefore better able to update information as required.

To enable asynchronous behavior, asynchronous communication occurs in parallel (simultaneously) with the main, synchronous flow of the business processes. This behavior makes asynchronous applications harder to accurately emulate using traditional synchronous Vuser scripts.

Although there are numerous types of asynchronous applications, there are three primary types: *push*, *poll*, and *long-poll*. For details, see "Asynchronous Communication Types - Request and Response Sequences" on the next page.

### See also:

- For an introduction to using asynchronous communication in Vuser scripts, see "VuGen Support for Asynchronous Communication" on page 396.

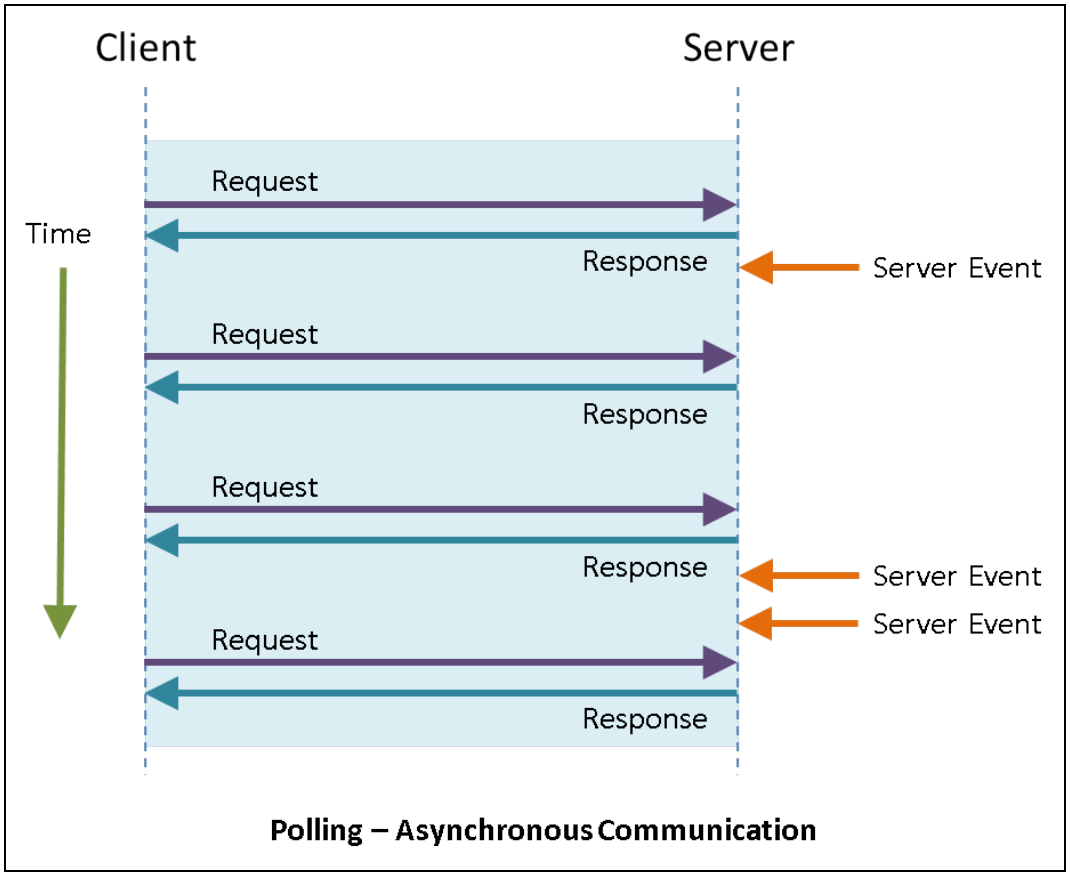
# Asynchronous Communication Types - Request and Response Sequences

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

Asynchronous communication is comprised of various request and response sequences. Such request and response sequences can be classified as one of three types of asynchronous communication: *poll*, *push*, and *long-poll*. When you develop a Vuser script, it is often useful to know the types of asynchronous communication that are implemented when the required business processes are performed.

## Polling Asynchronous Communication

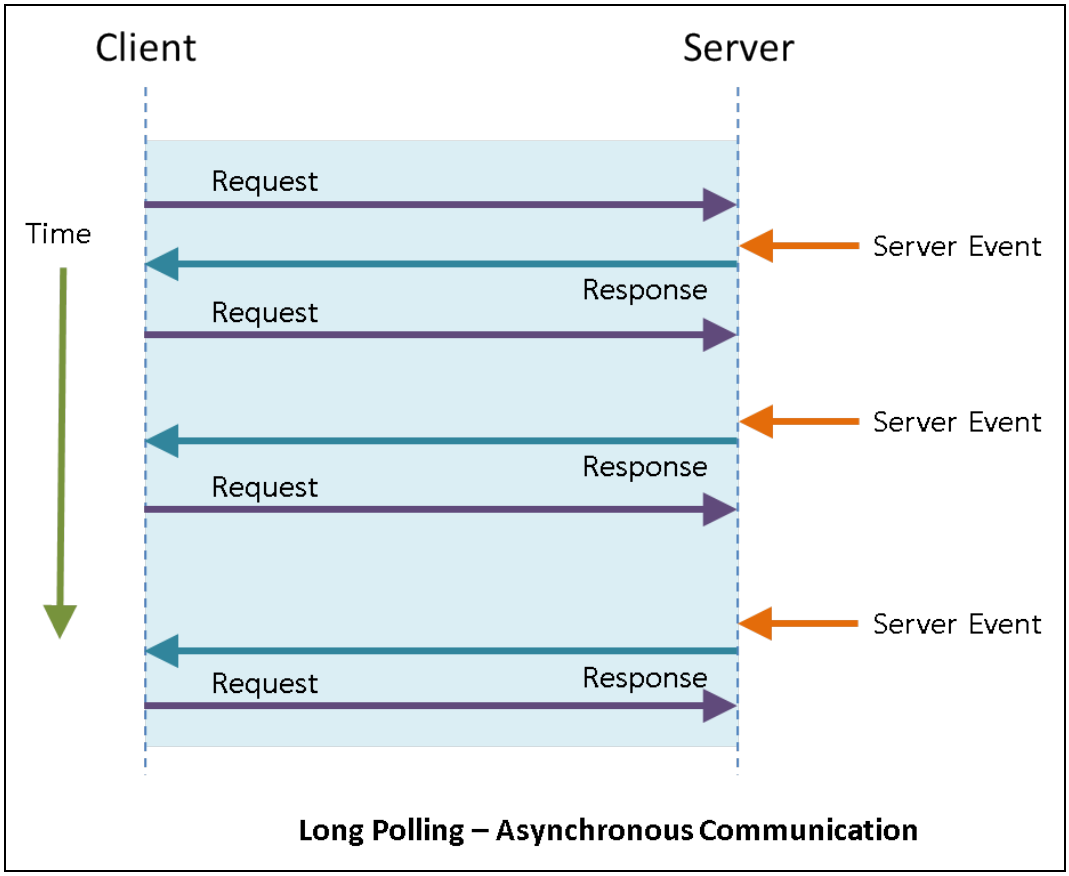
The browser sends HTTP requests to the server at regular intervals, for example, every 5 seconds. The server responds with updates. This enables the system to intermittently update the application interface inside the browser. If the server has no update, it informs the application that there is no update, based on the application protocol.




**Polling – Asynchronous Communication**

## Long-Polling Asynchronous Communication

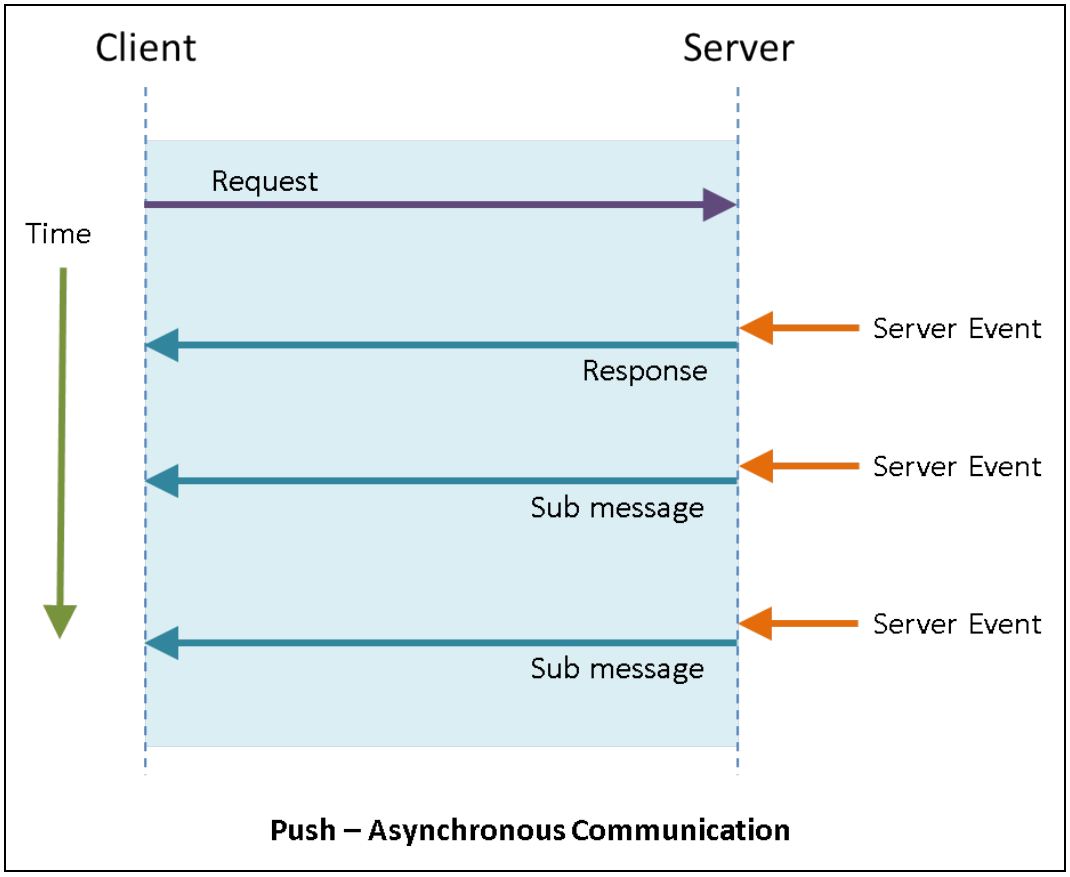
The client generates an HTTP request to a known address on the server. Whenever the server has an update, it responds with an HTTP response. Immediately after receiving the server response, the client issues another request.




## Push Asynchronous Communication

The client opens a connection with the server by sending a single HTTP request to a known address on the server. The server then sends a response that appears to never end, so that the client never closes the connection. Whenever necessary, the server sends a “sub message” update to the client over the open connection. The server may or may not terminate this connection. During the time the connection is open, if the server has no real update to send, it sends "ping" messages to the client to prevent the client from closing the connection for timeout reasons.

**Note:** Push-type conversations are supported for Web - HTTP/HTML protocol actions inside Web - HTTP/HTML, Flex, and Web Services Vuser scripts, NOT for **Flex_amf_call** steps in Flex Vuser scripts.




## VuGen Support for Asynchronous Communication

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

Web-based applications can exhibit synchronous behavior, asynchronous behavior, or a combination of both. For an introduction to these behavior types, see "Synchronous and Asynchronous Concepts" on page 393. VuGen enables you to build and run Vuser scripts that emulate user activity for both synchronous and asynchronous applications. To build a Vuser script for a synchronous application, follow the typical Vuser script building process. However, to build a script for an asynchronous application, you must perform some additional tasks - beyond the typical Vuser script building process. If you record and generate a script for an application that performs asynchronous behavior - without performing the additional asynchronous-related tasks, the script may not run successfully.

Building a Vuser script for an asynchronous application begins with recording the business processes that produce the asynchronous communication. After the business processes are recorded and the Vuser script is generated, VuGen scans the generated Vuser script and attempts to locate the asynchronous communication. If asynchronous communication is detected, VuGen modifies the script - inserting the appropriate asynchronous API functions.

To enable VuGen to successfully identify the asynchronous behavior in a Vuser script, the asynchronous communication must contain at least the required minimum number of request and response sequences.

- To identify a poll-type conversation, the recorded Vuser script must contain at least three sequences with matching URLs and similar polling intervals.
- To identify a long-poll-type conversation, the recorded Vuser script must contain at least three sequences with matching URLs.

In addition, the **Async Scan** recording option must be selected. For details, see "Create an Asynchronous Vuser Script" below.

In some scenarios, the modifications that VuGen makes to the Vuser script are sufficient to enable the script to run and emulate the required asynchronous behavior. In other scenarios, additional "manual" modifications are required. For details, see "How VuGen Modifies a Vuser Script for Asynchronous Communication" on page 401.

> **Note:** The modifications that must be made to a generated Vuser script to enable the script to emulate asynchronous behavior are dependent on the type of the asynchronous behavior: *push*, *polling*, or *long-polling*. For details, see "Asynchronous Communication Types - Request and Response Sequences" on page 394.

Asynchronous communication in a Vuser script is divided into one or more conversations. The individual asynchronous conversations that VuGen detects in a Vuser script are listed in the **Async** tab of the **Design Studio**. Use this list of asynchronous conversations to systematically analyze the modifications that VuGen made to the Vuser script to make sure that VuGen has correctly identified the asynchronous behavior, and correctly modified the Vuser script to emulate the required asynchronous behavior. For details on the **Async** tab of the **Design Studio**, see "Async Tab [Design Studio]" on page 421.

After modifying a Vuser script to enable it to emulate asynchronous communication, it may be necessary to perform correlation activities on the modified script. For details about correlation, see "Correlating Asynchronous Vuser Scripts" on page 408.

> **Note:** Async functionality is not supported when you replay a Vuser script in WinINet mode.

For details on how to build a Vuser script for an application that utilizes asynchronous communication, see "Create an Asynchronous Vuser Script" below.

## Create an Asynchronous Vuser Script

> **Note:**
>
> - For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.
> - Asynchronous communication is not supported for replaying a Vuser script in WinINet mode.

## Create a new Vuser script

1. Click the **New Script** button on the VuGen toolbar.
2. Select **Web - HTTP/HTML**, or one of the other Vuser protocols that support asynchronous communication.
3. Click **Create**. VuGen creates a basic Vuser script.

## Enable Async Scan

1. Select **Record > Recording Options**.
2. Under **General**, select **Code Generation**.
3. Make sure that the **Async Scan** check box is selected. This instructs VuGen to scan the Vuser script after recording, locate asynchronous communication, and insert the appropriate asynchronous functionality.

## Record the business processes using the typical VuGen recording process

1. Click **Record** on the VuGen toolbar.
2. Enter the required information in the Start Recording dialog box, and then click **Start Recording**.
3. Perform the business processes that the Vuser will emulate, and then click **Stop Recording** on the floating toolbar.

> **Note:** In order for VuGen to be able to successfully identify the asynchronous behavior in a Vuser script, the asynchronous communication must contain at least the required minimum number of client requests and server responses. For details, see "Asynchronous Communication Types - Request and Response Sequences" on page 394.

## Generate, scan, and modify the Vuser script

1. After you click **Stop Recording**, VuGen generates the Vuser script.
2. After generating the script, VuGen scans the generated script to locate instances of asynchronous communication.
3. If VuGen locates any instances of asynchronous communication, VuGen will modify the script to enable the script to run and emulate the asynchronous behavior. For details, see "How VuGen Modifies a Vuser Script for Asynchronous Communication" on page 401.
4. The **Design Studio** opens. Click the **Async** tab. The **Async** tab displays a list of all instances of asynchronous communication that VuGen located in the Vuser script.

## Review the modifications that VuGen made to the script

For each asynchronous conversation that appears in the **Async** tab of the **Design Studio**, perform the following tasks:

1. Open the Vuser script in the Editor.
2. Locate the **web_reg_async_attributes** step that starts the asynchronous conversation. Ensure that the **web_reg_async_attributes** step is located at the start of the asynchronous conversation.
3. Make sure that the URL parameter in the **web_reg_async_attributes** step is the same as one of the URLs that are specified in the action step that follows the **web_reg_async_attributes** step.

   For details on the **web_reg_async_attributes** step, see "Defining the Start of an Asynchronous Conversation" on page 404.
4. Notice that the step comment before the **web_reg_async_attributes** step contains a TODO token. The TODO token indicates that you should check the relevant callback implementations in the AsyncCallbacks.c extra file.
5. Locate the **web_stop_async** step that ends the asynchronous conversation. Ensure that the **web_stop_async** step is located at the end of the asynchronous conversation.
6. Make sure that the **web_stop_async** step runs as required. For details, see "Fine-Tuning the End of an Asynchronous Conversation" on page 408.

   For details on the **web_stop_async** step, see "Defining the End of an Asynchronous Conversation" on page 405.
7. Review the callback implementation and make modifications to the script as required. For details, see "Implementing Callbacks" on page 409.
8. Make sure that all *counter* and *complex string* parameters are set correctly. Notice that for each such parameter, a TODO comment exists and has a matching task in the **Tasks** pane. For details, see "Parsing URLs" on page 416.
9. Check the **Tasks** pane for specific actions that are required in order to complete the script development process. Such actions may include verifying callback implementations, or verifying the implementation of specific parameters.
10. Once all parameters are initialized correctly, run the script to make sure that the asynchronous conversation runs as expected.

## Asynchronous Communication API

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

The following functions enable Vuser scripts to emulate asynchronous communication:

| Function | Description |
|---|---|
| **web_reg_async_attributes** | This function registers the next action function as the beginning of an asynchronous conversation, and defines the behavior of the asynchronous communication. |
| **web_stop_async** | This function cancels the specified asynchronous conversation, including all its active and future tasks. |

| Function | Description |
|---|---|
| **web_sync** | This function suspends the Vuser script execution until the specified parameter is defined. |
| **web_util_set_request_url** | This function sets the specified string to be the request URL for the next request sent in the conversation. This is applicable only when called from a callback. |
| **web_util_set_request_body** | This function sets the specified string to be the request body for the next request sent in the conversation. This is applicable only when called from a callback. |
| **web_util_set_formatted_request_body** | This function is similar to the **web_util_set_request_body** function. However, this function is included as part of a Flex protocol asynchronous conversation instead of a Web(HTTP/HTML) protocol asynchronous conversation. This function expects an XML formatted request body, which will be converted before the request is sent. |

For details each of these functions, see the Function Reference (**Help > Function Reference**).

The **web_reg_async_attributes** function should be called before the step that starts the asynchronous conversation.

The **web_reg_async_attributes** function receives a number of arguments that define the asynchronous conversation. One of these arguments is the URL of the asynchronous conversation. As soon as the replay engine downloads this URL in the step that follows the **web_reg_async_attributes** function, the asynchronous conversation begins.

The callbacks that are registered in the **web_reg_async_attributes** function enable the script developer to control some of the characteristics of the asynchronous conversation (for example, to change the URL). The asynchronous conversation continues until the **web_stop_async** step, or until the end of the iteration. In a push-type conversation, the server may close the connection and thereby end the conversation.

**Note:** Async functionality is not supported when replaying a Vuser script in WinINet mode.

For details on how the asynchronous functions differ from synchronous functions, see "How Asynchronous Functions Differ from Synchronous Functions" below.

## How Asynchronous Functions Differ from Synchronous Functions

For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

The Function Reference (**Help > Function Reference**) includes several functions that enable Vuser scripts to emulate asynchronous communication. These asynchronous functions differ from the other API functions in the following ways:

- The network traffic that the asynchronous functions generate runs in parallel – simultaneously – with the main flow in the Vuser script. This means that the asynchronous communication can continue even when the synchronous steps end.
- The asynchronous communication continues even during execution of non-web functions (e.g. **lr_think_time**).
- Some of the asynchronous communication API functions use callback functions. The user needs to specify callbacks that are scheduled by the replay engine when a predefined event occurs. For details on using callbacks with asynchronous functions, see "Implementing Callbacks" on page 409.

## How VuGen Modifies a Vuser Script for Asynchronous Communication

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

After you create a Vuser script and record the required business processes, VuGen generates the Vuser script. VuGen then scans the generated script to locate instances of asynchronous communication. This process is called an Async scan. If VuGen locates any instances of asynchronous communication in the Vuser script, VuGen modifies the script to enable the script to run and emulate the required asynchronous behavior.

> **Note:** VuGen will scan a script for asynchronous communication only if the **Async Scan** recording option is selected. For details, see "Create an Asynchronous Vuser Script" on page 397.

Asynchronous communication in a Vuser script is divided into one or more conversations. The individual asynchronous conversations that VuGen detects in a Vuser script are listed in the **Async** tab of the **Design Studio**. Use this list of asynchronous conversations to systematically analyze the modifications that VuGen made to the Vuser script during the Async scan. Make sure that VuGen has correctly identified the asynchronous behavior in the Vuser script, and correctly modified the Vuser script to emulate the required asynchronous behavior. For details on the **Async** tab of the **Design Studio**, see "Async Tab [Design Studio]" on page 421.

> **Note:** After modifying a Vuser script to enable it to emulate asynchronous communication, it may be necessary to perform correlation activities on the modified script. For details about correlation, see "Correlating Asynchronous Vuser Scripts" on page 408.

### How VuGen modifies Vuser scripts

Asynchronous behavior in a Vuser script is divided into one or more asynchronous conversations. For each asynchronous conversation, VuGen performs the following tasks:

1. VuGen inserts a **web_reg_async_attributes** step before the start of the asynchronous conversation. The **web_reg_async_attributes** step includes an ID for the asynchronous conversation. This ID is used by a subsequent **web_stop_async** step to indicate the end of the asynchronous conversation. The Pattern argument indicates the type of the asynchronous behavior: *push*, *poll*, or *long-poll*.

   **Example: web_reg_async_attributes**("**Push_0**",

   "Pattern=**Push**",

   "URL=http://push.example.com/example",

   "RequestCB=**Push_0_RequestCB**",

   "ResponseHeadersCB=**Push_0_ResponseHeadersCB**",

   "ResponseBodyBufferCB=**Push_0_ResponseBodyBufferCB**",

   "ResponseCB=**Push_0_ResponseCB**",

   LAST);

   For details on how a **web_reg_async_attributes** step is used at the start of an asynchronous conversation, see "Defining the Start of an Asynchronous Conversation" on page 404.

   For details on the **web_reg_async_attributes** function, see the Function Reference (**Help > Function Reference**).

   For details on the types of asynchronous behavior that are supported by VuGen, see "Asynchronous Communication Types - Request and Response Sequences" on page 394.

2. VuGen adds a comment before the inserted **web_reg_async_attributes** step. The comment includes details about the asynchronous conversation, including:
   a. The ID of the asynchronous conversation.
   b. The URLs that are included in the conversation.
   c. Suggested implementations for the callback functions that are declared in the **web_reg_async_attributes** step. These implementations are added in AsyncCallbacks.c, one of the script's extra files.

      **Example:**

      /* Added by Async CodeGen.

      ID=Push_0

```
ScanType = Recording

The following URLs are considered part of this conversation:

    http://push.example.com/example

TODO - The following callbacks have been added to AsyncCallbacks.c.

Add your code to the callback implementations as necessary.

  Push_0_RequestCB

  Push_0_ResponseHeadersCB

  Push_0_ResponseBodyBufferCB

  Push_0_ResponseCB

*/
```

3. For *push* conversations, VuGen inserts asynchronous API functions into the Vuser script, but does not remove any of the recorded code from the Vuser script. For *polling* and *long-polling* conversations, VuGen may remove steps or step parameters from the generated Vuser script. VuGen removes steps or step parameters in cases where the relevant URLs will be requested by running the inserted asynchronous functions - and not by running the original steps that have been removed.

> **Note:** Removed steps are not deleted – they are commented out. You can "uncomment" these steps if required.

4. When relevant, VuGen adds a **web_stop_async** step at the end of the asynchronous conversation. The **web_stop_async** step marks the end of the asynchronous conversation. For details on the **web_stop_async** step, see the Function Reference (**Help > Function Reference**).
5. The recording snapshots are updated by grouping the tasks in the asynchronous conversation under the step that started the conversation.

## How VuGen modifies flex_amf_call steps

VuGen supports asynchronous polling and long-polling behavior in **flex_amf_call** steps. Flex scripts that contain *polling* or *long-polling* in **flex_amf_call** steps are handled by VuGen just like Web(HTTP/HTML) scripts, except for the following:

- The RequestCB will contain a commented call to **web_util_set_formatted_request_body**, which can be used to pass an XML formatted request body, which will be encoded and sent with the request.
- The **aResponseBodyStr** and **aResponseBodyLen** parameters of the ResponseCB give user access to the XML representation of the response body.

## Defining the Start of an Asynchronous Conversation

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

After VuGen scans a Vuser script for asynchronous communication, the Async tab of the Design Studio lists the asynchronous conversations that VuGen found in the script. VuGen inserts a **web_reg_async_attributes** into the Vuser script at the start of each asynchronous conversation that was detected. Use VuGen's Step Navigator to find the associated **web_reg_async_attributes** steps in the Vuser script. The **web_reg_async_attributes** steps should be located where the asynchronous conversations start when the script runs.

A **web_reg_async_attributes** step that is added to a Vuser script includes the following parameters:

- ID
- URL
- Pattern
- PollIntervalMS (for poll-type conversations only)
- RequestCB
- ResponseBodyBufferCB (for push-type conversations only)
- ResponseHeadersCB (for push-type conversations only)
- ResponseCB

The URL parameter in the **web_reg_async_attributes** step should be the same as one of the URLs that are specified in the step that follows the **web_reg_async_attributes** step. For details on the **web_reg_async_attributes** step, see the Function Reference (**Help > Function Reference**).

**Inserting a Comment**

When VuGen inserts a **web_reg_async_attributes** step into a script, VuGen inserts an associated comment before the **web_reg_async_attributes** step. The inserted comment contains information about the associated asynchronous conversation, such as the conversation ID, the communication pattern (*push*, *poll*, or *long-poll*), a list of URLs that are part of the asynchronous communication, and list of callbacks implemented in the AsyncCallbacks.c extra file.

Notice that the step comment contains a TODO token. The TODO token indicates that you should check the relevant callback implementations in the AsyncCallbacks.c extra file.

For details on how an asynchronous conversation is terminated, see "Defining the End of an Asynchronous Conversation" on the next page.

**Example - web_reg_async_attributes**

The sample code below shows a **web_reg_async_attributes** step that was added by VuGen. Notice that the **web_reg_async_attributes** step was added before a **web_url** step, and that the URL parameter in the **web_reg_async_attributes** step is the same as the URL parameter in the **web_url** step.

```
/* Added by Async CodeGen.
ID=Poll_0
ScanType = Recording

The following urls are considered part of this conversation:
    http://your URL.com/content.php?messages

TODO - The following callbacks have been added to AsyncCallbacks.c.
Add your code to the callback implementations as necessary.
    Poll_0_RequestCB
    Poll_0_ResponseCB
 */
    web_reg_async_attributes("ID=Poll_0",
        "URL=http://your URL.com/content.php?messages",
        "Pattern=Poll",
        "RequestCB=Poll_0_RequestCB",
        "ResponseCB=Poll_0_ResponseCB",
        LAST);

    web_url("content.php",
        "URL=http://your URL.com/content.php?messages",
        "Resource=0",
        "RecContentType=text/xml",
        "Referer=http://your URL.com/",
        "Snapshot=t2.inf",
        "Mode=HTML",
        LAST);
```

## Defining the End of an Asynchronous Conversation

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

After VuGen scans a Vuser script for asynchronous communication, the Async tab of the Design Studio lists the asynchronous conversations that VuGen found in the script. A **web_stop_async** step is inserted into the Vuser script at the end of each asynchronous conversation that was detected. Use VuGen's Step Navigator to find the associated **web_stop_async** steps in the Vuser script.

> **Note:** In some cases VuGen will not add a **web_stop_async** step at the end of an asynchronous conversation. This may occur when VuGen is not able to determine where the asynchronous conversation ends. This can occur when the asynchronous conversation was added due to a

specific Async rule or when the asynchronous conversation was not ended during the recording. For details on Async rules, see "Async Rules Overview" on page 419.

After VuGen has inserted a **web_stop_async** step into a Vuser script, make sure the **web_stop_async** step was added in the correct location in the script, that is – where the asynchronous conversation should end when the Vuser script runs.

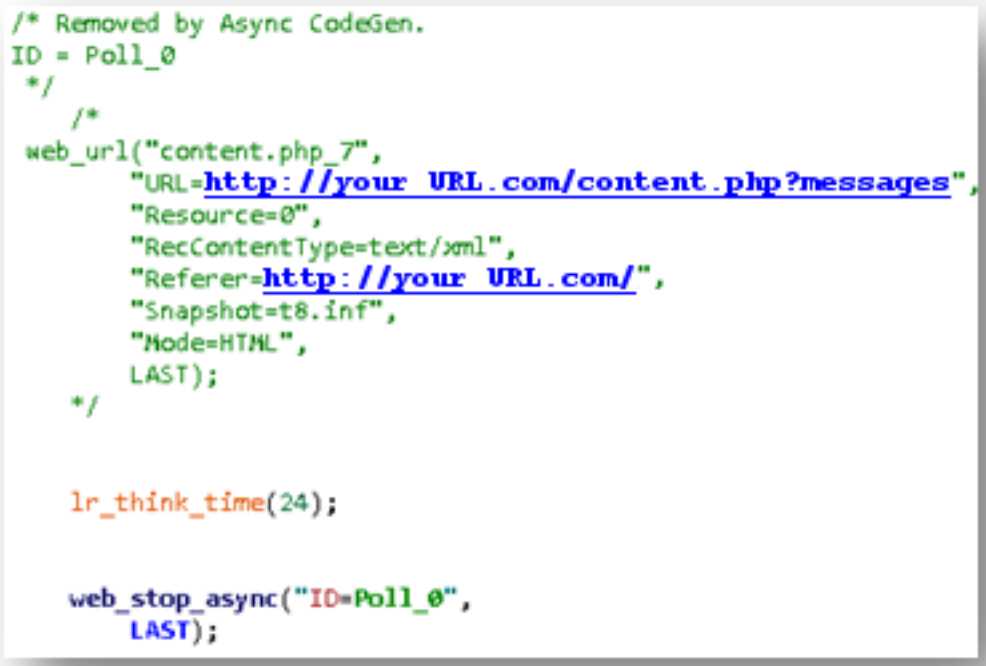
In order to make sure the asynchronous conversation ends correctly when the script runs, it may be necessary to modify the details of the **web_stop_async** step in the Vuser script. For details, see "Fine-Tuning the End of an Asynchronous Conversation" on page 408.

**Note:** All Async conversations are automatically terminated at the end of each iteration even if the **Simulate a new user each iteration** runtime option is disabled.

For details on how an asynchronous conversation is started, see "Defining the Start of an Asynchronous Conversation" on page 404.

#### Example: web_stop_async

In the code sample below, VuGen added a **web_stop_async** step at the end of a *poll* conversation. In this example, the original polling steps are commented out, and the **lr_think_time** steps that separated them have been merged into one **lr_think_time** step in order to emulate the duration of the entire *poll* conversation.




## Using Asynchronous Request Thresholds

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

You can fine-tune some of VuGen's behavior when VuGen scans a Vuser script to locate asynchronous communication. You use VuGen's asynchronous request thresholds to fine-tune VuGen's behavior. Each of the thresholds is associated with only one of the types of asynchronous conversations: *push*, *poll*, or *long-poll*.

- **Asynchronous request thresholds for push conversations**

  **Minimum Response Size**. Specify the minimum response content length (in bytes) for defining *push* asynchronous conversations. If the server sent less than the specified value, VuGen will not classify the conversation as a push-type asynchronous conversation.

  **Maximum Sub Message Size**. Specify the maximum sub message size (in bytes) sent by the server for defining *push* asynchronous conversations. If the server sent a sub message of size greater than the specified value, VuGen will not classify the conversation as a push-type asynchronous conversation.

  **Minimum Number of Sub Messages**. Specify the minimum number of sub messages for defining *push* asynchronous conversations. A push conversation in which less than the specified number of sub messages was sent by the server will not be classified by VuGen as a push-type asynchronous conversation.

- **Asynchronous request thresholds for poll conversations**

  **Interval Tolerance**. Specify the interval tolerance (in milliseconds) for classifying *poll* asynchronous conversations. A conversation in which intervals differ from each other by more than the specified value will not be classified by VuGen as a poll-type asynchronous conversation.

- **Asynchronous request thresholds for long-poll conversations**

  **Maximum Interval**. Specify the maximum interval (in milliseconds) between the end of one response and the start of a new request for classifying *long-poll* asynchronous conversations. A conversation in which a request starts more than the specified value after the end of the previous response will not be classified by VuGen as a long-poll type asynchronous conversation.

For details on the available asynchronous request thresholds, see "Asynchronous Options Dialog Box" on page 423.

## Fine-Tuning the End of an Asynchronous Conversation

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

After VuGen scans a Vuser script for asynchronous communication, the Async tab of the Design Studio lists the asynchronous conversations that were found in the script. A **web_stop_async** step is inserted into the Vuser script at the end of each asynchronous conversation that was detected. In order to make sure that each asynchronous conversation ends correctly when the script runs, it may be necessary to perform one or more of the following tasks:

- Remove the **web_stop_async** step so that the asynchronous conversation will be terminated at the end of the iteration.
- Move the **web_stop_async** step to a location that is after an existing action step or an existing **lr_think_time** step, so the asynchronous conversation will end after that step is performed.
- Add an **lr_think_time** step before the **web_stop_async** step, or change the time parameter in an existing **lr_think_time** step. Make sure that think-time is enabled in the runtime settings. For details, see the **General > Think Time** view.
- Add a **web_sync** step to stop the asynchronous conversation after a specified parameter receives a value. Use the asynchronous conversation callbacks to make sure the parameter receives a value only when you want to end the conversation.

## Correlating Asynchronous Vuser Scripts

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

After modifying a Vuser script to enable it to emulate asynchronous communication, it may be necessary to perform correlation activities on the modified script. Due to asynchronous nature, dynamic

values from asynchronous communication cannot be handled by Design Studio, and must be correlated manually.

You can search for dynamic values inside Response callbacks functions using the **lr_save_param_regexp** function. This function can be called from a callback to extract the necessary value from server response (**ResponseCB**) or response buffer (**ResponseBodyBufferCB**), and assign this value to a parameter. This parameter can then be used for correlations.

For details about the **lr_save_param_regexp** function, see the Function Reference (**Help > Function Reference**).

## Implementing Callbacks

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

After VuGen scans a Vuser script for asynchronous communication, the **Async** tab of the **Design Studio** lists the asynchronous conversations that were found in the script. For each asynchronous conversation found during the scan, VuGen adds the callback function signatures matching those declared in the **web_reg_async_attributes** step. The signatures are added to the AsyncCallbacks.c extra file.

The names of the callback functions start with the conversation ID of the asynchronous conversation. For example, the RequestCB callback for an asynchronous conversation with ID “LongPoll_0” will be **LongPoll_0_RequestCB**.

The names of the callback functions are declared in the **web_reg_async_attributes** step in the script.

The available callbacks are:

- **RequestCB**

  This callback is called before a request is sent.
- **ResponseBodyBufferCB**

  This callback is called when there is content in the response body buffer and at the end of the response body. This callback is generated by VuGen automatically for *push*-type conversations, but is available for *poll* and *long-poll* conversations as well.
- **ResponseCB**

  This callback is called after every response is received in the conversation.

**Example 1:**

In the following sample code, the three callback functions are declared in the **web_reg_async_attributes** step.

```
/* Added by Async CodeGen.
ID=LongPoll_0
ScanType = Recording

The following urls are considered part of this conversation:
     http://your URL.com/request.ashx?key=111111-11
     http://your URL.com/request.ashx?key=111111-11
     http://your URL.com/request.ashx?key=111111-11
     http://your URL.com/request.ashx?key=111111-11

TODO - The following callbacks have been added to AsyncCallbacks.c.
Add your code to the callback implementations as necessary.
    LongPoll_0_RequestCB
    LongPoll_0_ResponseCB
 */
    web_reg_async_attributes("ID=LongPoll_0",
        "URL=http://your URL.com/request.ashx?key=111111-11",
        "Pattern=LongPoll",
        "RequestCB=LongPoll_0_RequestCB",
        "ResponseCB=LongPoll_0_ResponseCB",
        LAST);
```

**Example 2:**

In the following sample code, the two callbacks are implemented in the AsyncCallbacks.c extra file.

```
int LongPoll_0_RequestCB()
{
    //enter your implementation for RequestCB() here

    //call web_util_set_request_url() here to modify polling url
    //web_util_set_request_url("<request_url>");

    //call web_util_set_request_body() here to modify request body:
    //web_util_set_request_body("<request body>");

    return WEB_ASYNC_CB_RC_OK;
}

int LongPoll_0_ResponseCB(
    const char *    aResponseHeadersStr,
    int             aResponseHeadersLen,
    const char *    aResponseBodyStr,
    int             aResponseBodyLen,
    int             aHttpStatusCode)
{
    //enter your implementation for ResponseCB() here

    return WEB_ASYNC_CB_RC_OK;
}
```

You can modify the callbacks to implement the required behavior. For details, see "Modifying Callbacks" on the next page.

**Example 3:**

The following sample code shows an implementation of the ResponseHeader callback function, including the three arguments: HTTP Status code, Accumulated headers string and Accumulated headers string length.

```
int Push_0_ResponseHeadersCB(

   int  aHttpStatusCode,

   const char * aAccumulatedHeadersStr,

   int aAccumulatedHeadersLen)

{

       //Enter your implementation for ResponseHeadersCB() here.

        lr_output_message("Response status code is :[%d]",
aHttpStatusCode);
```

```
        lr_output_message("Response headers are :/n[%s]",
aAccumulatedHeadersStr);

return WEB_ASYNC_CB_RC_OK;

}
```

A sample of the output from the above callback function is shown below:

```
Response status code is :[200]

Response headers are :

[HTTP/1.1 200 OK

Connection: close

Date: Tue, 25 Jun 2013 09:03:33 GMT

Server: Microsoft-IIS/6.0

Content-Type: text/html

Cache-control: private]
```

## Modifying Callbacks

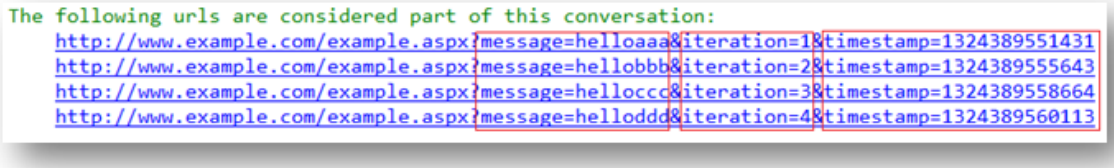
> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

After VuGen scans a Vuser script for asynchronous communication, the **Async** tab of the **Design Studio** lists the asynchronous conversations that were found in the script. For each asynchronous conversation found during the scan, VuGen adds the required callback function declarations in the **AsyncCallbacks.c** file. To implement the required behavior, you can modify the callbacks that were added by VuGen. Modifying a callback includes:

### Modifying the request URL in the RequestCB callback

In *poll* and *long-poll* conversations, requested URLs often change in each polling iteration. The change is usually determined by client-side logic, usually implemented by JavaScript that is executed by the browser. Parts of the URL may be determined by correlation of a known parameter, such as a session ID. For details, see "Parsing URLs" on page 416.

Request URLs in an asynchronous conversation will be modified before the request is sent by using the RequestCB and the **web_util_set_request_url** function.




### Modifying the request body in the RequestCB callback

The request body in requests that are part of an asynchronous conversation may need to be modified before the request is sent. You use the RequestCB and the **web_util_set_request_body** util function to modify the request body.

Modifying the request body is useful in *poll* and *long-poll* conversations in which each new request requires a different request body.

```
//an example of a parametrized request body sent in the RequestCB.
//the value of {request_body} may be set in the callback function,
//or elsewhere in the script.
web_util_set_request_body("{request_body}");
```

Each RequestCB that is generated by VuGen contains a commented snippet. You can "uncomment" the snippet in order to use the **web_util_set_request_body** util function.

If VuGen finds that different requests have different body values in the recorded conversation, the generated RequestCB will contain a comment that prompts you to check the recording in order to implement the request body sent in each request when the script runs.

```
//call web_util_set_request_body() here to modify request body:
//web_util_set_request_body("<request body>");
//TODO - use snapshot view to see examples of request bodies sent
```

### Modifying the response in the ResponseCB callback

You can modify the response callback in an asynchronous conversation to verify validity of the responses, or to wait for a specific event. For example, you could check the response headers for each

response to determine if a specific value was received.

```
int Poll_0_ResponseCB(
    const char *    aResponseHeadersStr,
    int             aResponseHeadersLen,
    const char *    aResponseBodyStr,
    int             aResponseBodyLen,
    int             aHttpStatusCode)
```

When the expected value has been received, you can use a **web_stop_async** step in the Action file to end the asynchronous conversation.

The following code sample provides an example for ending a synchronized conversation. In this example, in the **AsyncCallback.c** file, the scripts counts 10 iterations of the polling conversation, after which it creates a new parameter, **stopAsync**.

**Example:**

```
int Poll_0_ResponseCB(
  ...{
      //increment iteration counter for every response received.
      static int iter = 0;
      iter++;

      //Once the desired number of responses has been reached,
      //create and save the parameter.
      if (iter > 10)  {
              lr_save_int(iter, "stopAsync");
                      }
      return WEB_ASYNC_CB_RC_OK;
    }
```

In the Action file, the **web_sync** step uses the generated parameter, **stopAsync**, to end the conversation:

**Example:**

```
web_reg_async_attributes("ID=Poll_0","Pattern=Poll",
  "URL=http://pumpkin:2080/nioamfpoll;AMFSessionId=6F8D6108-E309-38B2-3D65-
963B431D0A38",
  "PollIntervalMs=3000",
  "RequestCB=Poll_0_RequestCB",
  "ResponseCB=Poll_0_ResponseCB",
 LAST);
```

```
web_custom_request("nioamfpoll;AMFSessionId=6F8D6108-E309-38B2-3D65-963B431D0A38_
2",
  "URL=http://pumpkin:2080/nioamfpoll;AMFSessionId=6F8D6108-E309-38B2-3D65-
963B431D0A38",
   "Method=POST",
  "Resource=0",
  "RecContentType=application/x-amf",
  "Referer=http://pumpkin:8081/lcds-samples/traderdesktop/traderdesktop.swf/
[[DYNAMIC]]/3",
  "Snapshot=t11.inf",
  "Mode=HTML",
  "EncType=application/x-amf",

"BodyBinary=\\x00\\x03\\x00\\x00\\x00\\x01\\x00\\x04null\\x00\\x02/6\\x00\\x00\\x0
0Z\n\\x00\\x00\\x00\\x01\\x11\n\\x07\\x07DSC\\x8D\\x02\n\\x0B\\x01\\x01\\x06\\x01\
n\\x05\tDSId\\x06I6F8D611E-DC1C-9D0C-2BBA-
36CC2AB8633B\\x01\\x0C!\\xC0\\xBE\\xA6Z74\\xBE\\xC3\\xCF\\xC7\\xFA\\xE6\\xC3\t\\xE
2\\x92\\x01\\x06\\x01\\x01\\x04\\x02",
 LAST);

lr_think_time(30);

//suspend the script until the desired number
//of iterations have been performed.
web_sync("ParamCreated=stopAsync", "RetryIntervalMs=1000",
"RetryTimeoutMs=300000", LAST);

web_stop_async("ID=Poll_0",LAST);
```

For more details about ending an asynchronous conversation, see "Defining the End of an Asynchronous Conversation" on page 405.

### Modifying callbacks in Flex Vuser scripts

For Flex asynchronous *polling* and *long-polling* conversations, the generated RequestCB in the **AsyncCallback.c** file contains a call to web_util_set_formatted_request_body, which sets an XML formatted request body for each request.

```
Example: web_util_set_formatted_request_body("<AMFPacket AMF_version=\"3\">"
    "<AMFHeaders />"
    "<Messages>"
        "<Message method=\"null\" target=\"/{Target_Poll_0}\">"
                …
        "</Message>"
    "</Messages>"
    "</AMFPacket>");
```

After uncommenting the commented code in the TODO section, and adding your callback code, open the Script Design Studio to scan for correlations.

Note that code generation automatically parameterizes the Target parameter in the request body. It also generates code for automatically incrementing this parameter before each polling iteration.

The generated RequestCB also contains a reminder to ensure that the counter initialization parameter for **Target_Poll_0** in the Action file matches the target attribute in the first *Message* element in the first polling request.

**Example:**
```
lr_param_increment("Target_Poll_0", "{Target_Poll_0}");

web_util_set_formatted_request_body("<AMFPacket AMF_version=\"3\">"
   "<AMFHeaders />"
   "<Messages>"
     "<Message method=\"null\" target=\"/{Target_Poll_0}\">"
       …
```

In the Action file, make sure that you initialize the same polling parameter used in **AsyncCallbacks.c**. In the following segment from the Action file, the polling parameter, *Target_Poll_0*, matches the one used in **AsyncCallbacks.c**.

**Example:**
```
/* Initialize target parameter before sending first request. */
/* Notice that parameter will be incremented once before first request. */
lr_save_int(5, "Target_Poll_0");
```

For more details on using callback functions, see "Implementing Callbacks" on page 409.

## Parsing URLs

For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

URLs that are included in asynchronous conversations often include query strings that are derived in a variety of ways. These strings may include:

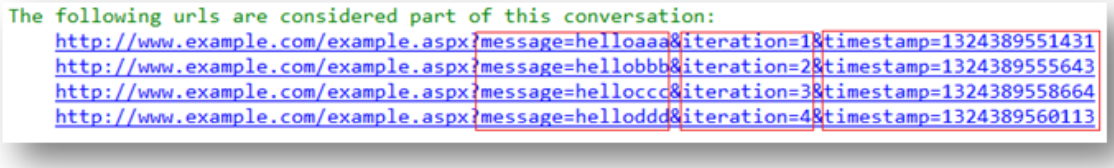
- Time-stamps
- Counters
- Complex strings

To enable a Vuser script to successfully perform asynchronous communication, VuGen must be able to recreate the required URLs.

- When the URL includes a time-stamp, VuGen is usually able to successfully create the required URL.
- When the URL includes a counter, VuGen is usually able to recreate the counter, but it may be necessary to manually initialize the counter in the script.
- When the URL includes more complex strings, the algorithms for generating the URLs must be manually added to the code in the Vuser script.

**Example:**

The sample code below shows a set of URLs that are part of a *long-poll* conversation. The URLs are included in the comment generated for a **web_reg_async_attributes** step:




If none of the parameters shown in the code sample above was found in VuGen's scan of the recorded Vuser script, the RequestCB implementation will contain a snippet that may be uncommented in order to set the URL for each response according to user defined code. For details, see "Modifying Callbacks" on page 412.

```
//call web_util_set_request_url() here to modify request url:
//web_util_set_request_url("<request url>");
```

If any or all of the parameters shown in the sample code above are found during VuGen's scan of the recorded Vuser script, the RequestCB implementation will contain the following:

- A comment prompting the user to call **web_util_set_request_url**. The comment will contain a parameterized version of the URL.
- For each *time-stamp* parameter found in the URL, a snippet for saving the time-stamp to a parameter.
- For each *counter* parameter found in the URL, a snippet for incrementing a counter parameter. A matching step for initializing the counter parameter will also be added to the Action file. The snippet will also contain examples of the URL token containing the counter parameter, as seen during the recording.
- For each *complex string* parameter found in the URL, a snippet for saving the string to a parameter. It is up to the user to generate the correct string to be saved to the parameter to be used in the URL. The snippet will also contain examples of the URL token that is considered an unknown parameter, as seen during the recording.

- A snippet for passing the parameterized version of the URL to the **web_util_set_request_url** function.

Example: A snippet containing the parameterized version of a URL.

```
//call web_util_set_request_url() here to modify polling url
//url is expected to be of the form:
//http://www.example.com/example.aspx?message={Unknown_LongPoll_0_0}
//&iteration={Counter_LongPoll_0_1}&timestamp={TimeStamp_LongPoll_0_2}
```

Example: A snippet prompting the user to set the value of an unknown parameter.

```
//TODO - implement parameter of type unknown: Unknown_LongPoll_0_0.
//Known examples for Unknown_LongPoll_0_0:
//message=[{"channel":"\/meta\/connect","connectionType":"long-polling","id":3,"clientId":"113fc44"}],
//message=[{"channel":"\/meta\/connect","connectionType":"long-polling","id":5,"clientId":"113fc44"}],
//message=[{"channel":"\/meta\/connect","connectionType":"long-polling","id":6,"clientId":"113fc44"}],
//message=[{"channel":"\/meta\/connect","connectionType":"long-polling","id":7,"clientId":"113fc44"}].
lr_save_string("[{\"channel\":\"\\/meta\\/connect\",\"connectionType\":\"long-polling\",\"id\":3,\"clientId\":\"113fc44\"}]",
"Unknown_LongPoll_0_0");
```

Example: A snippet for incrementing a counter parameter.

```
//TODO - check counter initialization for Counter_LongPoll_0_1 in Action file.
//Known examples for the token containing Counter_LongPoll_0_1:
//iteration=1, iteration=2, iteration=3, iteration=4 gPoll_0_1:
lr_param_increment("Counter_LongPoll_0_1", "{Counter_LongPoll_0_1}");
```

Example: A snippet for initializing a counter parameter.

```
lr_save_int(0, "Counter_LongPoll_0_1");
```

Example: A snippet for saving a timestamp parameter.

```
web_save_timestamp_param("TimeStamp_LongPoll_0_2",LAST);
```

Example: A snippet for passing the parameterized version of a URL to the **web_util_set_request_url** function.

```
//once all parameters have been assigned, copy them to the updated url,
//and call web_util_set_request_url() with the updated url:
web_util_set_request_url("http://www.example.com/example.aspx?message={Unknown_LongPoll_0_0}
&iteration={Counter_LongPoll_0_1}&timestamp={TimeStamp_LongPoll_0_2}");
```

## Async Rules Overview

> **Note:** For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

In some cases, when VuGen performs an Async scan, VuGen may fail to correctly identify some of the asynchronous conversations that are included in the Vuser script. In other cases, VuGen may erroneously classify regular synchronous steps as part of asynchronous conversations. To help rectify both of these scenarios, you can define Async rules to determine how requests to specified URLs are classified during an Async scan.

Async rules can be positive or negative.

- **Positive:** When VuGen fails to identify asynchronous conversations that are included in a Vuser script, implement a positive Async rule to enable VuGen to identify the asynchronous behavior.

  Scenario: VuGen does not identify URLs under **http://www.true-async.com/push_example.aspx** as push asynchronous conversations, and you know that they are part of push asynchronous conversations. Add a positive rule to enable VuGen to correctly identify the push asynchronous conversations. When you regenerate the script, the Async scan will apply the added rule, and all URLs that start with **http://www.true-async.com/push_example.aspx** will be included as part of push asynchronous conversations.
- **Negative:** When VuGen erroneously classifies regular synchronous steps as part of an asynchronous conversation, implement a negative Async rule to prevent VuGen from erroneously identifying asynchronous behavior.

  Scenario: VuGen identifies all URLs under http://www.not-async.com/ as asynchronous poll conversations. You know that these are not asynchronous conversations. Implement a negative Async rule to prevent VuGen from erroneously identifying asynchronous behavior. When you regenerate the script, the Async scan will apply the added rule so that all URLs that start with http://www.not-async.com/ will not be classified as part of asynchronous conversations.

For details on how to implement Async rules, see "Adding Async Rules" below.

## Adding Async Rules

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

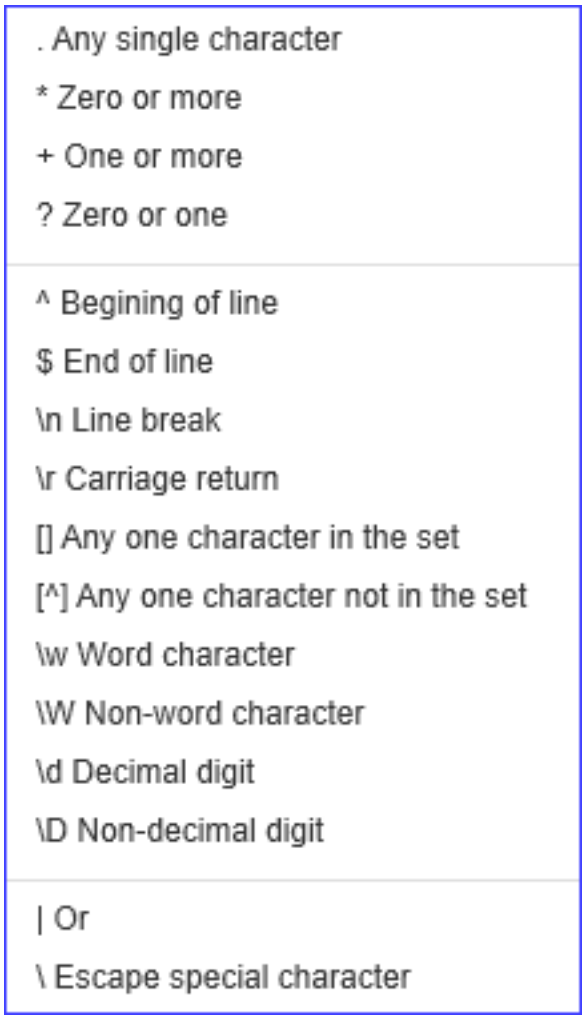
When VuGen scans a Vuser script after recording or regenerating the script, VuGen may fail to identify asynchronous conversations that are included in the Vuser script. In other cases, VuGen may erroneously classify a regular synchronous step as part of an asynchronous conversation.

You can define Async rules that determine how requests to specified URLs will be classified during an Async scan. Async rules may be positive or negative.

## Adding a positive Async rule

1. Select **Record > Recording Options > General > Code Generation** and then click **Async Options**. The Asynchronous Options dialog box opens.
2. Under **Asynchronous Regular Expressions**, click **Add Async Rule** . The Add Rule dialog box opens.
3. From the **Type** list, select **Push**, **Poll**, or **Long Poll**, as required.
4. In **URL Regular Expression**, enter a regular expression for URLs that should be considered part of asynchronous conversations.

   Special characters that you can include in a regular expression:



To include characters such as "?" and "+" in the regular expression, insert a backslash "\" before the required character.

5. Click **OK**. The new rule appears in the list of Async rules for the Vuser script.

   When you regenerate the script:

   - For each push conversation that includes a URL that matches the regular expression, VuGen inserts asynchronous API functions into the Vuser script, but does not remove any of the recorded code from the Vuser script.
   - For each polling or long-polling conversation that includes a URL that matches the regular expression, VuGen inserts asynchronous API functions into the Vuser script, and may remove steps or step parameters from the generated Vuser script. VuGen removes steps or step parameters in cases where the relevant URLs will be requested by running the inserted asynchronous functions - and not by running the original steps that have been removed.

   For further details, see "How VuGen Modifies a Vuser Script for Asynchronous Communication" on page 401.

### Adding a negative Async rule

1. Select **Record > Recording Options > General > Code Generation** and then click **Async Options**. The Asynchronous Options dialog box opens.
2. Under **Synchronous Regular Expressions**, click **Add Async Rule** . The **Add Asynchronous Rule** dialog box opens.
3. From the **Rule Type** list, select **Not Async**.
4. In **URL Regular Expression**, enter a regular expression for URLs that should not be considered part of asynchronous conversations.
5. Click **OK**. The new rule appears in the list of Async rules for the Vuser script.

   When you regenerate the script, steps that contain URLs that match the regular expression will not be included in asynchronous conversations.

### See also:

- "Async Rules Overview" on page 419

## Async Tab [Design Studio]

The Async tab of the Design Studio lists all the occurrences of asynchronous communication that VuGen detected in the Vuser script.

| | |
|---|---|
| **To access** | • Select **Design > Design Studio**, and then click the **Async** tab.<br>• Click the **Design Studio** button on the VuGen toolbar, and then click the **Async** tab. |

| | |
|---|---|
| **Important information** | • The **Design Studio** button is enabled only when you display a recorded Vuser script in the Solution Explorer.<br>• The Async tab enables you to only view the asynchronous communication that is included in the Vuser script - you cannot edit any of the asynchronous details from the Async tab. Changes to the asynchronous details must be made in the Vuser script. |
| **Relevant tasks** | "Create an Asynchronous Vuser Script" on page 397 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Type** | Indicates the origin of the asynchronous code in the Vuser script:<br>**Record.** The asynchronous code was added by VuGen during an Async scan that was performed after recording or regenerating the Vuser script.<br>**Rule.** The asynchronous code was added by VuGen due to a specific rule in the Async rules file.<br>**Manual.** The asynchronous code was manually added by a user. |
| **Action** | The section of the Vuser script in which the asynchronous behavior is located. |
| **Occurrences** | • For push-type conversations, **Occurrences** is always 1.<br>• For poll and long-poll conversations, **Occurrences** indicates the number of steps or extra resource attributes that were removed [commented-out] by VuGen during the Async scan of the Vuser script. |
| **Status** | Always has the value **Applied**. |
| **Async Type** | The type of the asynchronous behavior that was detected: **Push**, **Poll**, or **Long-Poll**. |
| **URL** | The URL in the **web_reg_async_attributes** step that starts the asynchronous conversation. |
| **Filter** | Select which asynchronous conversations to display in the conversation list. |
| **Details** | Expands the dialog box to show details about the selected asynchronous conversation. |
| **Name** | Always has the value **web_reg_async_attributes**. |
| **Line** | The line in the Vuser script that contains the **web_reg_async_attributes** step. |
| **Action Name** | The section of the Vuser script in which the asynchronous behavior is located. |

| UI Element | Description |
|---|---|
| **Description** | The comment in the Vuser script that precedes the **web_reg_async_attributes** step. |
| **Occurrences in Snapshot** | • For push-type conversations, displays the response body.<br>• For poll and long-poll conversations, displays HTTP attributes associated with the asynchronous conversation. |
| **Options** | Opens the Asynchronous Request Thresholds dialog box. |

## Asynchronous Options Dialog Box

This dialog box enables you to fine-tune some of VuGen's behavior when VuGen scans a Vuser script to locate asynchronous communication.

| | |
|---|---|
| **To access** | **Record > Recording Options > General > Code Generation** and then click **Async Options**. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Minimum Response Size** | Specify the minimum size (in kilobytes) of a server response for defining *push* asynchronous conversations. If the server sent less than the specified value, VuGen will not classify the conversation as a push-type asynchronous conversation. |
| **Maximum Sub Message Size** | Specify the maximum sub message size (in kilobytes) sent by the server for defining *push* asynchronous conversations. If the server sent a sub message of size greater than the specified value, VuGen will not classify the conversation as a push-type asynchronous conversation. |
| **Minimum Number of Sub Messages** | Specify the minimum number of valid sub messages for defining *push* asynchronous conversations. A push conversation in which less than the specified number of valid sub messages was sent by the server will not be classified by VuGen as a push-type asynchronous conversation. |
| **Interval Tolerance** | Specify the interval tolerance (in milliseconds) for classifying *poll* asynchronous conversations. A conversation in which intervals differ from each other by more than the specified value will not be classified by VuGen as a poll-type asynchronous conversation. |
| **Maximum Interval** | Specify the maximum interval (in milliseconds) between the end of one response and the start of a new request for classifying *long poll* asynchronous conversations. A conversation in which a request starts more than the specified value after the end of the previous response will not be classified by VuGen as a long-poll type asynchronous conversation. |

| Asynchronous Regular Expressions Table | |
|---|---|
| **<Activate Rule>** | A check box indicating whether or not the rule is activated. To select all rules, select the check box in the toolbar. |
| **Rule Type** | **Push**, **Poll**, and **Long Poll** are positive asynchronous rules. **Not Async** is a negative rule. For details on the asynchronous rule types, see "Async Rules Overview" on page 419. |
| **Regular Expression** | A regular expression for URLs that should be considered part of asynchronous conversations. For a list of the special characters that you can include in a regular expression, see "Adding Async Rules" on page 419. |

## Asynchronous Example - Poll

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

The following example describes a Vuser script that includes a poll asynchronous conversation. The application that is emulated by the Vuser is a demo of a messaging service that sends updates on demand. The browser displays the page, and sends requests for updates at intervals of approximately 5 seconds.

```
web_url("message.txt",
    "URL=http://example.com/content/message.txt?key=0",
    "Resource=0",
    "RecContentType=text/plain",
    "Snapshot=t4.inf",
    "Mode=HTML",
    LAST);

lr_think_time(4);

web_url("message.txt_2",
    "URL=http://example.com/content/message.txt?key=1",
    "Resource=0",
    "RecContentType=text/plain",
    "Snapshot=t5.inf",
    "Mode=HTML",
    LAST);

lr_think_time(5);

web_url("message.txt_3",
    "URL=http://example.com/content/message.txt?key=2",
    "Resource=0",
    "RecContentType=text/plain",
    "Snapshot=t6.inf",
    "Mode=HTML",
    LAST);

lr_think_time(4);

web_url("message.txt_4",
    "URL=http://example.com/content/message.txt?key=3",
    "Resource=0",
    "RecContentType=text/plain",
    "Snapshot=t7.inf",
    "Mode=HTML",
    LAST);

lr_think_time(5);

web_url("message.txt_5",
    "URL=http://example.com/content/message.txt?key=4",
    "Resource=0",
    "RecContentType=text/plain",
    "Snapshot=t8.inf",
    "Mode=HTML",
    LAST);
```

**Note:** You can modify VuGen's asynchronous request thresholds to assist VuGen in finding poll-type conversations. For details, see "Using Asynchronous Request Thresholds" on page 407.

The above script was generated by VuGen after the required business processes were recorded. An asynchronous scan was not performed on the script after the script was generated. Notice that the script contains a series of **web_url** functions with a repeating URL, namely:

**http://example.com/content/message.txt**. These **web_url** functions are separated by **lr_think_time** functions, indicating that the **web_url** functions repeat at intervals of approximately 5 seconds.

When the Vuser script runs, requests for **http://example.com/content/message.txt** should be sent repeatedly until the script is finished. Additionally, these requests should be sent in parallel (simultaneously) with other actions performed in the Vuser script.

After VuGen performed a scan for asynchronous communications on the script, the script looks as follows:

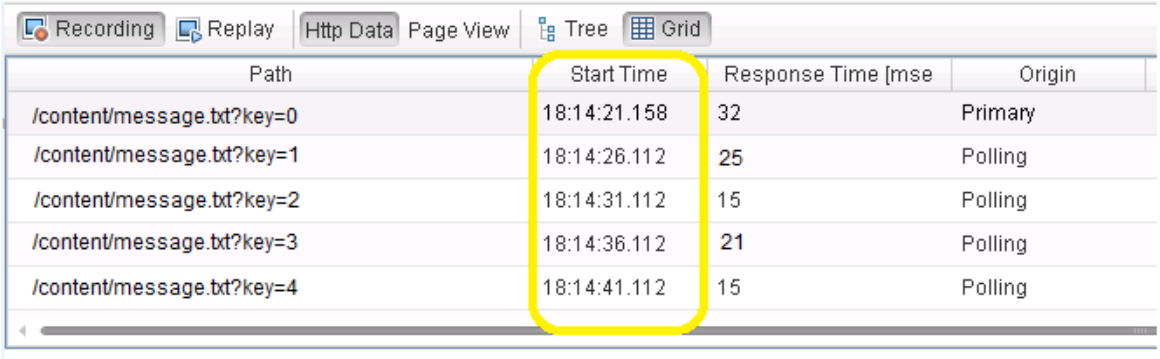
```
/* Added by Async CodeGen.
ID=Poll_0
ScanType = Recording

The following URLs are considered part of this conversation:
    http://example.com/content/message.txt?key=0
    http://example.com/content/message.txt?key=1
    http://example.com/content/message.txt?key=2
    http://example.com/content/message.txt?key=3
    http://example.com/content/message.txt?key=4

TODO - The following callbacks have been added to AsyncCallbacks.c.
Add your code to the callback implementations as necessary.
    Poll_0_RequestCB
    Poll_0_ResponseCB
 */
    web_reg_async_attributes("ID=Poll_0",
        "Pattern=Poll",
        "URL=http://example.com/content/message.txt?key=0",
        "PollIntervalMs=4900",
        "RequestCB=Poll_0_RequestCB",
        "ResponseCB=Poll_0_ResponseCB",
        LAST);

    lr_save_int(0, "Counter_Poll_0_0");

    web_url("message.txt",
        "URL=http://example.com/content/message.txt?key=0",
        "Resource=0",
        "RecContentType=text/plain",
        "Snapshot=t4.inf",
        "Mode=HTML",
        LAST);

/* Removed by Async CodeGen.
    /*
/* Removed by Async CodeGen.
    /*
/* Removed by Async CodeGen.
    /*
/* Removed by Async CodeGen.
    /*

    lr_think_time(20);

    web_stop_async("ID=Poll_0",
        LAST);
```

Notice that a **web_reg_async_attributes** function has been added to the script before the first **web_url** function that calls **http://example.com/content/message.txt**, and that a **web_stop_async** function has been added after the last **web_url** function that calls the same URL.

Except for the first call to **http://example.com/content/message.txt**, all other **web_url** functions that call the same URL have been commented-out by VuGen.

Notice that the **lr_think_time** function has been replaced by the `PollIntervalMs` argument for **web_reg_async_attributes**.

The Snapshot pane for the remaining **web_url** function shows that the snapshots for the removed **web_url** functions now have Origin = Polling, and that they start at intervals of 5 seconds.




## Asynchronous Example - Push

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

The following example describes a Vuser script that is developed to emulate a browser displaying an application that utilizes push-type asynchronous conversations. The application is a demo of a “stock quote” page. The browser shows the page with the stock values, and then sends a request and receives a response with updated stock values. The request remains open until it is closed by the user. For as long as the page is displayed, the server will continue to send sub-messages as part of the response - whenever the server has an update for the displayed stocks. Whenever such a sub-message is received by the client, the client displays the updated stock values.

> **Note:** You can modify VuGen's asynchronous request thresholds to assist VuGen in finding push-type conversations. For details, see "Using Asynchronous Request Thresholds" on page 407.

| Name | Price | Time | Change | Bid Size | Bid | Ask | Ask Size | Min | Max | Ref. |
|---|---|---|---|---|---|---|---|---|---|---|
| Anduct | 3.09 | 10:32:54 | 1.64 | 5000 | 3.09 | 3.1 | 35500 | 2.48 | 3.64 | 3.04 |
| Ations Europe | 17.92 | 10:33:05 | 11.37 | 86000 | 17.92 | 17.98 | 1000 | 12.8 | 19.33 | 16.09 |
| Bagies Consulting | 6.43 | 10:33:04 | -10.57 | 4500 | 6.43 | 6.44 | 8500 | 5.74 | 8.64 | 7.19 |
| BAY Corporation | 3.2 | 10:32:03 | -11.84 | 37500 | 3.2 | 3.21 | 63500 | 3.13 | 4.28 | 3.63 |
| CON Consulting | 6.8 | 10:33:05 | -10.64 | 24000 | 6.8 | 6.83 | 51000 | 6.23 | 9.08 | 7.61 |
| Corcor PLC | 2.58 | 10:32:54 | 12.17 | 61000 | 2.58 | 2.59 | 42000 | 1.87 | 2.7 | 2.3 |
| CVS Asia | 13.48 | 10:33:06 | -12.41 | 94500 | 13.45 | 13.48 | 11000 | 12.38 | 18.34 | 15.39 |
| Datio PLC | 5.81 | 10:33:00 | 9.41 | 42500 | 5.8 | 5.81 | 1500 | 4.4 | 6.34 | 5.31 |
| Dentems | 4.87 | 10:33:05 | 0.2 | 12000 | 4.86 | 4.87 | 35500 | 3.91 | 5.67 | 4.86 |
| ELE Manufacturing | 7.19 | 10:33:01 | -5.51 | 13500 | 7.19 | 7.2 | 51000 | 6.25 | 9.09 | 7.61 |

If you attempt to run a script that calls a push url - without first performing an asynchronous scan - the replay will halt while waiting for the response to the highlighted request. After two minutes, VuGen will display an error similar to the following, in the Replay log:

```
Action.c(140): Error -27782: Timeout (120 seconds) exceeded while waiting to receive
data for URL "http://push.example.com" [MsgId: MERR-27782]
```

The error indicates that the response never finished.

Regenerating the script with Async Scan enabled will create a script similar to the following:

```
/* Added by Async CodeGen.
ID=Push_0
ScanType = Recording

The following urls are considered part of this conversation:
    http://push.example.com/STREAMING_IN_PROGRESS?LS_session=S343716d5eb050c64T2253451&LS_phase=4903&LS_domain=lights",

TODO - The following callbacks have been added to AsyncCallbacks.c.
Add your code to the callback implementations as necessary.
    Push_0_RequestCB
    Push_0_ResponseBodyBufferCB
    Push_0_ResponseCB
 */
    web_reg_async_attributes("ID=Push_0",
        "URL=http://push.example.com/STREAMING_IN_PROGRESS?LS_session={CorrelationParameter}&LS_phase=4903&LS_domain",
        "Pattern=Push",
        "RequestCB=Push_0_RequestCB",
        "ResponseBodyBufferCB=Push_0_ResponseBodyBufferCB",
        "ResponseCB=Push_0_ResponseCB",
        LAST);

    web_url("STREAMING_IN_PROGRESS",
        "URL=http://push.example.com/STREAMING_IN_PROGRESS?LS_session={CorrelationParameter}&LS_phase=4903&LS_domain=",
        "Resource=0",
        "RecContentType=text/html",
        "Referer=http://www.app.example.com/GWT_StockListDemo_Basic/lightstreamer/lsengine.html",
        "Snapshot=t13.inf",
        "Mode=HTML",
        LAST);

    web_custom_request("control.js",
        "URL=http://push.example.com/lightstreamer/control.js", "",
        "Method=POST",
        "Resource=0",
        "RecContentType=text/plain",
        "Referer=http://push.example.com/ajax_frame.html?phase=594&domain=example.com&",
        "Snapshot=t14.inf",
        "Mode=HTML",
        "Body=LS_session={CorrelationParameter}&LS_table=1&LS_win_phase=19&LS_req_phase=311&LS_op=add&LS_mode=MERGE&LS_id=item1",
        LAST);
/* Added by Async CodeGen.
ID = Push_0
 */
    web_stop_async("ID=Push_0",
        LAST);


    return 0;
}
```

Notice that a **web_reg_async_attributes** function has been added before the **web_url** function that starts the *push* conversation, and that a **web_stop_async** function has been added after the last action step in the script. The script will now run successfully. The *push* conversation will remain active – running in parallel with the other script functions – until the **web_stop_async** function, or until the end of the script is reached.

Note that during the Async scan, VuGen did not remove (comment-out) any of the generated code in the Vuser script.

## Asynchronous Example - Long-Poll

> For a list of protocols that support asynchronous communication, see "Protocol Support for 64-bit Recording, Async, and IPv6" on page 465.

The following example describes a Vuser script that emulates an application that implements a *long-poll* asynchronous conversation. The application is a demo of a “chat” page. A browser shows the chat page, and sends a request that remains open until a new message is sent to the chat by another user. After such a message is sent:

- The response is finished.
- The new message is shown in the browser.
- The browser sends another request in order to listen for the next message sent to the chat.

**Note:** You can modify VuGen's asynchronous request thresholds to assist VuGen in finding long-poll type conversations. For details, see "Using Asynchronous Request Thresholds" on page 407.

The following is the Vuser script that VuGen generated after recording the application - before an asynchronous scan was performed. The script contains a series of **web_url** functions with similar URLs. Since new requests are sent as soon as the previous response is finished, no **lr_think_time** functions are added between the **web_url** functions. This help to indicate that this is a *long-poll* conversation and not a *poll* conversation.

When the Vuser script runs, requests to the chat application should be sent repeatedly every time a response from the chat application is finished. In addition, requests should be sent in parallel (simultaneously) with other actions performed in the script.

After VuGen performs an asynchronous scan on the script, the modified script looks as follows:

```
/* Added by Async CodeGen.
ID=LongPoll_0
ScanType = Recording

The following urls are considered part of this conversation:
    http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%
    http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%
    http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%
    http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%

TODO - The following callbacks have been added to AsyncCallbacks.c.
Add your code to the callback implementations as necessary.
    LongPoll_0_RequestCB
    LongPoll_0_ResponseCB
- */
    web_reg_async_attributes("ID=LongPoll 0",
        "URL=http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%
        "Pattern=LongPoll",
        "RequestCB=LongPoll_0_RequestCB",
        "ResponseCB=LongPoll_0_ResponseCB",
        LAST);

    web_url("request.ashx_3",
        "URL=http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%
        "Resource=0",
        "RecContentType=text/html",
        "Referer=",
        "Snapshot=t4.inf",
        "Mode=HTML",
        LAST);

/* Removed by Async CodeGen.
ID = LongPoll_0
- */
    /*  http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%
  web_url("request.ashx 4",
        "URL=http://example.com/request.ashx?key=111111-1111-1111-1111-111111token=123488858&message=%5B%
        "Resource=0",
        "RecContentType=text/html",
        "Referer=",
        "Snapshot=t5.inf",
        "Mode=HTML",
        LAST);
-   */

/* Removed by Async CodeGen.
```

Notice that a **web_reg_async_attributes** function has been added before the first **web_url** function that calls the chat application.

Except for the first call the chat application, all other **web_url** functions that call similar URLs have been commented out.

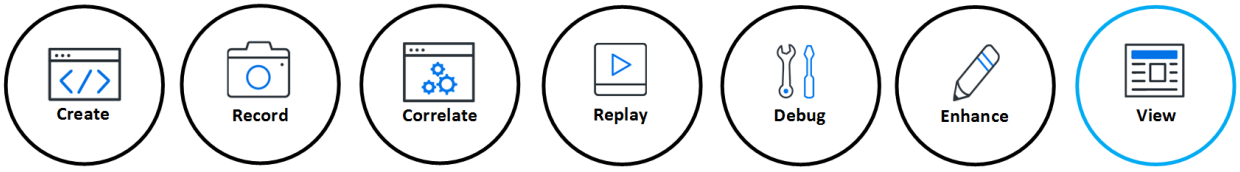
The Snapshot pane for the remaining **web_url** function shows that the snapshots for the removed steps now have **origin = Polling**.

The response times vary greatly as the responses arrive only when another user has sent a chat message. This helps to indicate that this is a long-poll conversation, and not a poll conversation.

# Viewing Replay Results

Developing a Vuser script includes the steps shown below. This topic provides an overview of the seventh step, viewing the results of the replay of a Vuser script.

Select an image to learn more.




To assist with debugging a Vuser script, you can view a Replay Summary report or a Business Process report after replaying the script. The Replay Summary report is useful typically during the scripting process, whereas the Business Process report provides higher level information or a business process view of the script.

- **Replay Summary report.** Summarizes the results of your script run. VuGen generates the report during the Vuser script execution and you view the report when script execution is complete. For details see "Replay Summary Pane" on page 131.
- **Business Process report.** A customizable report that can include the replay summary, details of the transactions, rendezvous points, and parameters in the script, and additional information about the script replay. For details, see "Create a Business Process Report" on page 163.

# Virtual Table Server (VTS)

VTS (Virtual Table Server) is a web-based application that works with Vuser scripts. VTS offers an alternative to standard VuGen parameterization.

When you use standard parameterization, each Vuser is assigned parameter values from a dedicated set of values - parameter values are not shared between Vusers. In contrast, VTS enables you to assign parameter values from a single set of parameter values to multiple Vusers. This may enable you to more accurately emulate a real-user environment.

VTS is composed of two components, VTS-Client and VTS-Server. VTS-Client is a set of API functions that are used to access data in VTS-Server. Because the VTS API functions integrate with VuGen, there is no need to install VTS-Client. VTS-Server includes a table that contains parameter values that can be used by your Vuser scripts. The VTS table is composed of columns and rows. Each column represents a set of values that can be assigned to a specific parameter in your Vuser scripts. The cells within a column contain the actual values that are assigned to the parameter.

To install VTS, see "Install VTS" on the next page.

> **Note:** Significant changes were made to VTS in LoadRunner version 11.52. When upgrading to VTS 11.52 or later, these changes may result in various compatibility issues. For details on script modifications that are required in order to resolve these compatibility issues, see "Preparing Vuser Scripts for VTS Parameterization" on page 456.

## See also:

- "Parameterization Overview" on page 357
- TruClient Help Center (select the relevant version) for details on how to use VTS functionality in TruClient Vuser scripts.

# VTS Parameterization

After you use VuGen to record a business process, VuGen generates a Vuser script. Typically, the generated script contains actual values that were detected during recording. For example, the server may assign a unique ID to each user that accesses the site. This unique ID may be included in the script. Suppose you want the Vuser to perform the script's actions (query, submit, and so forth) using different values from those recorded. To do this, you replace the recorded values in the script with parameters. This is known as parameterizing the script.

- For more information about parameterization in LoadRunner, see the *LoadRunner User Guide* or the *Virtual User Generator User Guide*.

When you run a Vuser script that contains a parameter, the parameter must be assigned a value each time the parameter is called. LoadRunner provides two methods for assigning values to parameters:

- Standard LoadRunner parameterization functionality. For details, see the Help for *LoadRunner* or the *Virtual User Generator*.
- VTS (Virtual Table Server). For an introduction, see "The VTS Table" on page 447.

## Standard Parameterization vs. VTS Parameterization

When you use the standard LoadRunner parameterization, each Vuser is assigned parameter values from a dedicated set of values - parameter values are not shared between Vusers. In contrast, VTS enables you to assign parameter values from a single set of parameter values to multiple Vusers. This may enable you to more accurately emulate a real-user environment.

## Vuser protocols that support VTS functionality

VTS functionality is supported by all Vuser protocols that use C as the scripting language.

For protocols that are not supported, you can use the **REST API** for VTS to develop you own client. For details, download the VTS3 Communication Specification from AppDelivery Marketplace.

### See also:

- "Parameterization Overview" on page 357

# Install VTS

This topic describes how to install the Virtual Table Server (VTS).

### To install VTS:

1. If you have an older version of VTS, uninstall it.
2. Run the setupVTS.exe file located in the **\Additional Components\Virtual Table Server** folder in the installation media. The VTS Setup Wizard opens, displaying the welcome page.

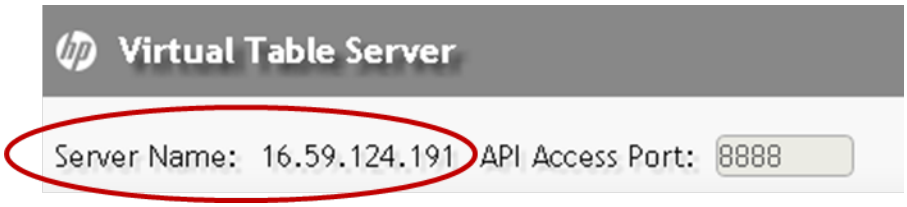
3. Follow the online instructions to complete the VTS installation.
4. During the VTS installation process, the Configure VTS administration server screen appears. This screen lets you configure the VTS Administration server.
5. In the **Admin UI server port** box, keep the default value, `4000`.
6. Click **Next** to continue with the installation. The Configure VTS screen appears.
7. Specify where to save the VTS data file.
8. Make sure that the **Start Virtual Table Server Automatically** check box is selected.
9. Click **Next**, and then follow the wizard's instruction to complete the VTS installation procedure.

> **Note:** At the end of the installation process, a shortcut for VTS is created and added to the Start menu, **Micro Focus > Tools > Virtual Table Server**.
>
> This shortcut gives you access to the VTS user interface on the local machine. If you change the port that is used to access the VTS user interface, you must manually update the URL property of the shortcut. For details on how to change the VTS user interface access port, see "Configuring VTS" on page 438.
>
> If you are unable to access the VTS user interface, make sure that the VTS Service service is started. To start the **VTS Service** service, go to **Control Panel > Systems & Security > Administration Tools > Services**. Right-click **VTS Service** and select **Start**.

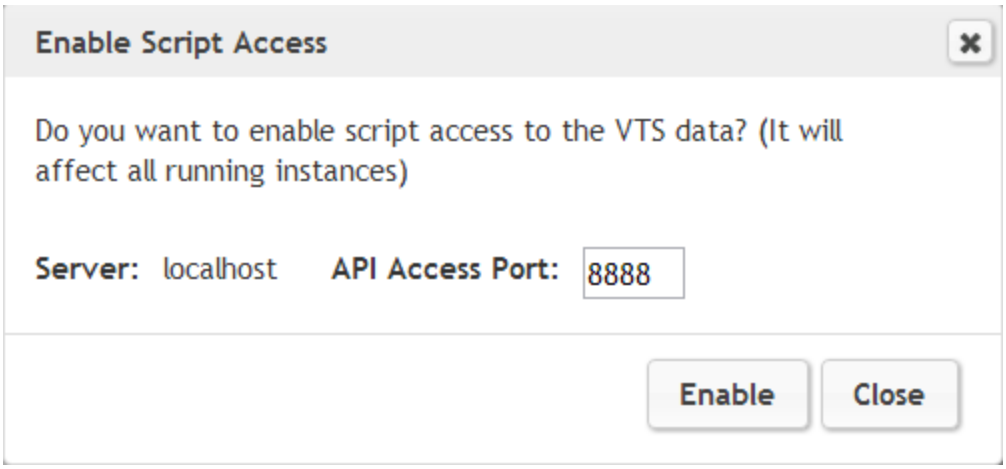
# Manage VTS

## Accessing the VTS Server

The data in the VTS table is maintained on a server known as the VTS server. The name or IP address of the VTS server is displayed in the top left corner of the VTS user interface.




You can access the data in the VTS table in any of the following ways:

| | |
|---|---|
| VTS user interface | You can use the standard functionality in the VTS user interface to access data in the VTS table. For example, you can modify an existing parameter value or add a new value. For details, see "Manipulating the Contents of the VTS Table" on page 449.<br><br>The **VTS Service** Windows service enables UI access to the VTS table. By default, when VTS is installed, **VTS Service** is set to start automatically. If you experience problems accessing VTS, make sure that **VTS Service** is started. By default, port 4000 is used for UI access to VTS. For details on changing the default UI access port, see "Configuring VTS" on page 438. |
| Vuser scripts | Vuser scripts can include API functions that enable Vusers to read and modify data in the VTS table. VTS lets you enable or disable API access to the VTS server. When you enable API access, you must specify the port that Vuser scripts use to access the VTS server. The currently used API access port is displayed in the top left corner of the VTS user interface.<br><br> <br><br><br>The default API access port is 8888, and a direct connection is needed between the VTS server and the load generator machine. For details on how to change the default API access port, see "Configuring VTS" on page 438.<br><br>• To enable API access to the VTS server, from the **Access from Script** dropdown list, click **Enable**.<br><br>  <br><br><br>In the Enable Script Access dialog box, enter the API access port, and then click **Enable**.<br><br>• To disable API access to the VTS server, from the **Access from Script** dropdown list, click **Disable**. |

# Using Authentication

VTS enables you to require authentication of each user before the user can access data in the VTS table. When authentication is required, each user must supply a valid user name and associated password to access VTS.

## Configure authentication

When VTS is installed, by default authentication is not required. To specify that authentication is required, you need update the configure.json file settings.

**Note:**

- VTS uses basic or NTLM authentication.
- After you have logged into VTS using authentication, you can close the browser to log out of VTS.

To require authentication to access VTS:

1. Open the configure.json file. By default, the configure.json file is located in the **C:\Program Files\Micro Focus\VTS\web** folder.
2. Set **authentication** to **Basic** or **NTLM**.
3. In the admin section of the configure.json file, enter the required user authentication details for all users who will access VTS. For details on the configure.json file, see "Configuring VTS" on the next page.
4. Restart the **VTS Service** service.

   Go to **Control Panel > Systems & Security > Administration Tools > Services**. Right-click **VTS Service** and select **Start**.

## SSL communication

With standard VTS authentication, user names and passwords are transferred as regular text. If this is a security concern, you can use HTTPS for more secure communication between your browser and the VTS server. You require a public key and a private key to implement SSL communication.

To use HTTPS to access VTS:

1. Copy the public key file and private key file to the folder in which the **Node.exe** file is located—typically **C:\Program Files\Micro Focus\VTS\web**.
2. Open the configure.json file, and set **useSSL** to **true**. Then set each of the properties described under "useSSL" on page 444 in "Configuring VTS" on the next page.
3. Restart **VTS Service**.

   Go to **Control Panel > Systems & Security > Administration Tools > Services**. Right-click **VTS**

**Service** and select **Start**.

4. Access the VTS web site using an `https` URL.

## Logging

VTS maintains log files that record various operations that were performed by VTS. These log files may be useful when debugging parameter-related aspects in a Vuser script. VTS lets you configure the level of logging that occurs. You modify the logging level by editing the **logger > level** option in the configure.json file. For details on setting the logging level, see "Configuring VTS" below.

The following logging levels are available:

| | |
|---|---|
| **error** | Creates the least comprehensive logs, including only errors |
| **warning** | Creates a log file with intermediate verbosity |
| **info** | Creates the most comprehensive logs, including errors, warnings, and additional information |

By default, VTS saves the log files in the c:\windows\temp\VTS folder. You can change the default folder by modifying the **logger > path** option in the configure.json file. For details, see "Configuring VTS" below.

To access log files from a browser window:

1. Open the **configure.json** file. The default location for this file is **%Program Files%\Micro Focus\VTS\web**.
2. Enable logging by configuring the **enableDiag** parameter:

   ```
   "enableDiag": true,
   ```

3. Restart the **VTS Service** in the **Services** window.
4. Navigate to **http://<severname>:<port>/data/diag** and select **logs**.

> **Example:** If the VTS is on your local host on port 4000, navigate to `http://localhost:4000/data/diag`

## Configuring VTS

VTS includes a number of settings that define certain aspects of how VTS operates.

### Accessing the VTS configuration tool

These configuration settings are stored in the **configure.json** file that is saved with the standard JSON format.

You can use the VTS configuration tool to configure the VTS. From the Windows Start menu, go to, **Micro Focus > LoadRunner > Tools > Configure Virtual Table Server**.

Alternatively, you can update the configure.json file itself. By default, this is located in **C:\Program Files\Micro Focus\VTS\web**.

> **Note:** If you modify the configuration file to a non-valid format, the **VTS Service** service will fail to start.

For details, see "Configuring VTS" on the previous page.

## VTS configuration settings

The configure.json configuration file includes the following settings:

| Setting Key | Explanation |
|---|---|
| dbPath | The stored path of the VTS database.<br>Default: **"C:/ProgramData/Micro Focus/VTS/db/data"** |
| defaultDbName | The name of the VTS database. The default name is **Default**.<br>**Recommended:** Do not modify this setting. |
| adminPort | The TCP port that is used for Admin UI access to VTS. This port is used by the VTS UI to read and modify data in the VTS table. The default value is 4000.<br>**Note:** The port used for Admin UI access must be different from the port used for Vuser script API access. |
| defaultApiPort | The default port that Vuser scripts use to access the VTS table using API functions. If you use the VTS UI to change the API access port, the new port number will override the default value.<br>Default: **8888** |
| defaultStart | Specifies whether or not API access to the VTS table should be enabled when the **VTS Service** service is started.<br>Default: **false** |
| enableDiag | Enables the VTS diagnosis feature. If set to **true**, you can access the following URL to view logs, apis, and meta data:<br>**http://vts_server_name:4000/data/diag**<br>Default: **false** |

| Setting Key | Explanation |
|---|---|
| autoRefreshGrid | When the data is changed by an API or by another Admin UI, the Admin UI automatically refreshes itself to show the updated data. This is the default behavior. If this is affecting the performance of the load test, set this key to **false**.<br>Default: **true** |
| maxInstancesAllowed | VTS supports multiple instances. Use this setting to configure the maximum number of instances that VTS should support.<br>Default: **50** |
| normalizeCSV | When set to **true**, enables VTS to normalize non-strict CSV files where rows don't have the same separators.<br>Default: **false** |
| autoCreateIndexedColumn | When set to **true**, each column is automatically indexed.<br>Default: **false** |

| Setting Key | Explanation |
|---|---|
| authentication | • **Basic**. Authentication is required to access the VTS table from the VTS user interface. Authentication includes a user name and associated password. For further details on authentication, see "Using Authentication" on page 437.<br>Example<br>`"authentication": "Basic",/*authentication methods, 3 options "None", "Basic", "NTLM". Default is "None", which means no auth is needed*/`<br>`"users":`<br>`[`<br>`/*Add users with the following format when authentication set to "Basic"*/`<br>`{"user": "johns", "password": "pass", "display": "John Smith"}`<br>`],`<br><br>• **NTLM**. NTLM authentication is required. If authentication is set to NTLM, you must configure domain and domaincontroller.<br>Example<br>`"admin": {`<br>`"authentication": "NTLM",/*authentication methods, 3 options "None", "Basic", "NTLM". Default is "None", which means no auth is needed*/`<br>`"users":`<br>`[`<br>`/*Add users with the following format when authentication set to "Basic"*/`<br>`{"user": "johns", "password": "pass", "display": "John Smith"}`<br>`],`<br>`/*Specify domain/domaincontroller setting when authentication set to "NTLM"*/`<br>`"domain":"mydomain.net",`<br>`"domaincontroller": "ldap://examplelab.mydomain.net",` |

| Setting Key | Explanation |
|---|---|
| | Default: **None** |
| users | When **authentication=Basic**, specify the list of users that are registered to access the VTS table from the VTS user interface.<br>• **user.** The name that the user must use to access the VTS server.<br>• **password.** The password that the user must use to access the VTS server.<br>• **display.** The name that will be displayed by VTS after the user has been successfully authenticated and connected to the VTS server.<br>**Note:** If multiple users are listed, insert a comma between each line in the list, as shown below:<br>`{"user": "johns", "password": "pass", "display": "John Smith"},{"user": "peterb", "password": "pass1", "display": "Peter Brown"}` |

| Setting Key | Explanation |
|---|---|
| role | For both Basic or NTLM authentication, you can assign the user roles: full-control **administrator**, or read-only **user**.<br>Example<br>`"role":`<br>`{"administrator":["peterb", "johns3"],`<br>`"user":["johns2","johns"]}`<br>**Note:**<br>• If no users are assigned roles, all users will have full permissions on the VTS server.<br>• If some, but not all, users are assigned roles, all the rest of the users will have read-only permissions only on the VTS server.<br>• **Administrator users for NTLM**: Define each of these users using both the fully-qualified domain name and the NetBIOS name.<br>**For example:**<br>`"role":`<br>`{"administrator":[ "mydomain.net\\peterb", "mydomain\\peterb","johns3"],`<br>`"user":["johns2","johns"]}`<br>Alternatively, if using a browser to access the VTS server, you need only configure the fully qualified domain name, and configure your browser settings to prompt for the VTS server login credentials. |
| requireBasicAuthForAPI | When **authentication** is set to **Basic**, and **requireBasicAuthForAPI** is set to **true**, the user name and password are required when accessing VTS data from the API level. The setting does not affect VTS admin UI.<br>Default: **false** |
| domain | When **authentication** is set to **NTLM**, specify the domain. |
| domaincontroller | When **authentication** is set to **NTLM**, specify the domaincontroller. |

| Setting Key | Explanation |
|---|---|
| useSSL | Specifies whether or not HTTPS should be used to access the VTS table from the VTS user interface.<br><br>By default, useSSL is set to **false**.<br><br>To use SSL communication, set **useSSL** to **true**, and define the following properties:<br><br>• **certificate.** The name of the certificate file to use for HTTPS communication.<br><br>For example: `"certificate": "certificate.pem"`<br><br>**Note:** The certificate file must be located in the same folder as **node.exe**. See Tip.<br><br>• **privateKey.** The name of the private key to use for HTTPS communication.<br><br>For example: `"privateKey": "privatekey.pem"`<br><br>**Note:** The privateKey file must be located in the same folder as **node.exe**. See Tip.<br><br>• **ca.** The name of the certificate (ca) file. The ca file is used to verify the client certificate.<br><br>This setting is optional, and is relevant only when the **requestClientCert** is set to **true**. Otherwise, it is ignored.<br><br>Default: **vts.crt**<br><br>**Note:** The ca file must be located in the same folder as **node.exe**. See Tip.<br><br>• **tlsVersion.** Possible values: **auto**, **tls1.0**, **tls1.1**, **tls1.2**.<br><br>Default: **auto** (SSLv23_method). This is the default option for secureProtocol in Node JS.<br><br>• **ciphers.** Specifies which ciphers are supported.<br><br>Default: **ALL**<br><br>• **requestClientCert.** If set to **true**, when a client tries to connect to the server using HTTPS, the server requests a certificate, and then attempts to verify the certificate.<br><br>Default: **false**<br><br>**Tip:** Instead of copying the key files described above to the same folder as **node.exe**, you can specify an absolute path to specify the files in another location. For example, `"certificate": "c:\\temp\\certificate.pem"`. The configure.json file uses the JSON format. Use a double backslash "\\" to represent a single backslash "\". |

| Setting Key | Explanation |
|---|---|
| logger | Contains the settings for VTS's logging options.<br>**level**: Specifies the level of logging performed:<br>• **error**: creates the least comprehensive logs, containing only error messages<br>• **warning**: creates a log file with intermediate verbosity<br>• **info**: creates the most comprehensive logs, containing error messages, warnings, and additional information<br>**transports**: Specifies certain output options for logging.<br>• **path**: Specifies the folder where the VTS log files are saved.<br>Default: **%TEMP%\\VTS**, where %TEMP% is an environment variable that refers to the current user account.<br>**Note:** The %TEMP% variable for the Local System account usually points to the **%windir%\temp** folder. In this scenario, the VTS log files are located in **%windir%\temp\vts**. |
| defaultLanguage | Sets the VTS server language. You can set the following languages:<br>en, zh-cn, ja, ko, it, es, pt-br, fr, ru<br>Default: **en** |

## Creating and Managing Multiple VTS Instances

Create multiple tables or VTS instances to access data. The first time you connect to the VTS, you connect to the main instance. Subsequent connections are referred to as *named instances*.

### Create a named instance

1. From the browser of the main instance, select **Options > Other Instances> Start Instance**. This opens the Start Instance dialog box.
2. Enter a name in the **Instance Name** box.
3. Specify a port in the **API Access Port** box. This port will serve as both the UI and API access port.
4. If you want the new table to automatically open in a browser, leave the **Open the instance UI** option checked.

**Note:**

- You can only create a named instance from the browser of the main instance.
- The title of the VTS browser for named instances is Virtual Table Server (<Instance Name>)

### Stop a named instance

1. From the browser of the main instance, select **Options > Other Instances > Stop Instances**. This opens the **Stop Instances** dialog box.
2. Check the instances you want to stop.

> **Note:** Stopping an instance does not delete the data from the table. The same data will be displayed when you restart the instance.

## Using the command line mode

Use the command line to create, delete, and populate VTS tables. Open a command window from the folder that contains the vtscmd.js file (`%ProgramFiles%\Micro Focus\VTS\web`) and type a command with the following syntax: `node vtscmd <command> <parameter>`. For example to access the help, type `node vtscmd /help`.

> **Note:** In the VTS installation folder (`%ProgramFiles%\Micro Focus\VTS\web\samples`) sample command files are provide. One launches new instances and imports data (`InitVTSInstances.bat`) and the other stops all running instances (`StopAllInstances.bat`).

The following table describes the available commands:

| Command | Parameters | Description | Example |
|---|---|---|---|
| /cert /key | | When **requestClientCert=true** in configure.json, specify the certificate and key for server verification. | `Node vtscmd /start /port 2000 /name Employees /cert "clientCert.cer" /key "clientCert.key"` |
| /delete_ data | /port | Delete all data stored in a VTS table | `node vtscmd /delete_data /port 2000` |
| /domain /username /password | | When **authentication=NTLM** in configure.json, use these to specify the credentials for NTLM authentication. | `Node vtscmd /start /port 2000 /name Employees /domain Domain /username User /password Pass` |
| /help or /? | | List all VTS commands and parameters | `node vtscmd /help` |

| Command | Parameters | Description | Example |
|---|---|---|---|
| /import | /port<br>/file<br>/delimiter (optional) | Import a local .csv file to the VTS | `node vtscmd /import sample/customers.csv /port 2000` |
| /list | | List all the running VTS instances | `node vtscmd /list` |
| /start | /port<br>/name | Start an instance of the VTS | `node vtscmd /start /port 2000 /name Employees` |
| /stop | /port or /name | Stop a VTS instance | `node vtscmd /stop /name Employees` or `node vtscmd /stop /port 2000` |
| /stop_all | | Stop all VTS instances | `node vtscmd /stop_all` |

# VTS Data

## The VTS Table

VTS includes a table that contains parameter values that can be used by your Vuser scripts. The VTS table is composed of columns and rows. Each column represents a set of values that can be used by a specific parameter in your Vuser scripts. The cells within a column contain the actual values of the parameter.

> **Note:** Check the LoadRunner Readme for limitations.

### Populate the VTS table

There are several ways in which you can populate and maintain the VTS table:

| | |
|---|---|
| VTS user interface | The VTS user interface enables you to manipulate the data in the VTS table. For details on the VTS user interface, see "Manipulating the Contents of the VTS Table" on page 449. |
| API functions | Include various VTS API functions in a Vuser script to manipulate the data in the VTS table. For details on the available VTS API functions, see the **Utilities** section in the *LoadRunner Function Reference*. |

| | |
|---|---|
| Import data from a database or CSV file | VTS lets you import data from a database of a CSV file directly into the VTS table. |
| Import data in command line mode | Import data through a command line string. For details, see "Using the command line mode" on page 446. |

## VTS cell background colors

VTS changes the background color of the cells in the VTS table to provide visual indication about the data in the cells. The following background colors may appear in the VTS table:

| | |
|---|---|
| Yellow | The cell contains the search string. |
| Orange | The cell contains a value that is not unique in the column. |
| Blue | The cell is currently selected. |
| White | The cell contains a value for a parameter, even if the cell is blank. |
| Gray | The cell does not contain a value for a parameter. |

## Divide the VTS table into pages

The VTS table can include numerous rows of data. The total number of rows is indicated in parentheses in the column header.

VTS divides the rows of data into pages. Each page can include 10, 20, or 30 rows of data. You use the **Page-Rows** control at the bottom of VTS to specify how many rows you want VTS to display in each page.

Controls at the bottom of VTS enable you to move between the pages in the table.

| | |
|---|---|
| | Display the first page in the table. |
| | Display the previous page. |
| | Display the next page. |
| | Display the last page in the table. |

**Note:**

- You can use the **PgUp** and **PgDn** keys to navigate through the pages in the VTS table.
- Press the **Home** key to display the first page in the table, and the **End** key to display the last page.

### Refresh the VTS table data

The data in the VTS table is maintained in a database on a server. There may be various entities that change the data in the table. Periodically, VTS checks the server to establish if any changes have been made to the data in the table.

If VTS detects that changes have been made, an icon appears in the **Refresh** button on the VTS toolbar indicating the number of operations that were done on the server. Click **Refresh** to update the data that is displayed in VTS.

At all times, the VTS window displays the total number of records, in the bottom right corner.

For details on how to modify the interval with which VTS checks for changes on the server, see "Configuring VTS" on page 438.

### Reconnect to the Database


If the VTS server goes down or if it becomes disconnected due to network problems, the toolbar prompts you with the **Disconnected** icon. Click the icon to reconnect to the server.

### Multiple VTS Instances

Create multiple tables or VTS instances to access data. The main instance or default instance is available when VTS is installed. Subsequent instances created are referred to as named instances.

There are some differences between the main instance and named instances:

- In the main instance, you define a port for the UI and one for API access. For named instance, you specify only one port for both.
- The title of the table of the main instance is Virtual Table Server. The title of the table for named instances is Virtual Table Server ( <Instance Name>).

For details on working with multiple instances, see "Creating and Managing Multiple VTS Instances" on page 445.

## Manipulating the Contents of the VTS Table

The VTS user interface provides you with the functionality required to manipulate the data in the VTS table. Alternatively, you can use Vuser scripts to modify the data in the VTS table. For details, see "Using the API to Access the VTS Table" on page 458.

For details on how to search for a text string inside the VTS table, see "Searching Inside the VTS Table" on page 451.

## Refreshing the data

If you have a connection to database and the data has changed, The VTS window displays an icon with the number of operations that were performed against the VTS server.

## Deleting the entire table

To delete the entire contents of the VTS table, click **Delete Table** on the VTS toolbar. VTS deletes all existing data in the table, including the column headings.

## Changing the name of a column

You can change the name of any column in the VTS table.

To change the name of a column:

1. Select the required column, and then click **Column Actions > Edit Column Name** or double-click the column header.
2. In the Change Column Name dialog box, enter the new name of the column.
3. Click **Save**.

## Inserting a column

You can insert a new column into the VTS table, either to the right or the left of the selected column.

To insert a new column:

1. Click a column adjacent to where you want to inset a new column.
2. Click **Column Actions > Insert Column to the Left** or **Insert Column to the Right**.
3. In the Create Column dialog box, enter a name for the new column, and click **Save**. VTS inserts the specified column.

## Deleting a column

You can delete any column from the VTS table. When you delete a column, VTS deletes the entire column, including the column header.

To delete a column from the VTS table:

1. Select the column to delete.
2. Click **Column Actions > Delete Column**.

## Deleting the content of a column

You can delete the contents of a column in the VTS table. When you delete the contents of a column, VTS deletes the content only - but does not delete the column header.

To delete the contents of a column:

1. Select the column whose contents you want to delete.
2. Click **Column Actions > Clear Column's Content**.

### Inserting a value at the top of a column

When you insert a value at the top of a column, VTS adds a new row at the top of the column, and moves all existing data in the column down one row.

To insert a new value at the top of a column:

1. Select the column in which you want to add the entry.
2. Click **Column Actions > Insert Value at Start**.

### Appending a value to the end of a column

1. Select the column in which you want to add the entry.
2. Click **Column Actions > Append Value to End**.

## Searching Inside the VTS Table

VTS lets you search for a text string inside the VTS table.

To search for a text string in the VTS table:

1. In the **Search** box

, enter a text string.
2. Press <Enter> or the **Search** button to highlight all instances of the text string on the current page.
3. Continue to click the **Search** button to locate instances of the text string on the remaining pages in the table.

> **Note:** The VTS search is not case-sensitive.

## Detecting Duplicate Entries

VTS lets you check that the entries in a column of the VTS table are unique. You can check whether an existing value is unique, or you can check that a value is unique when you enter a new value or modify an existing value.

### Check that an existing value is unique

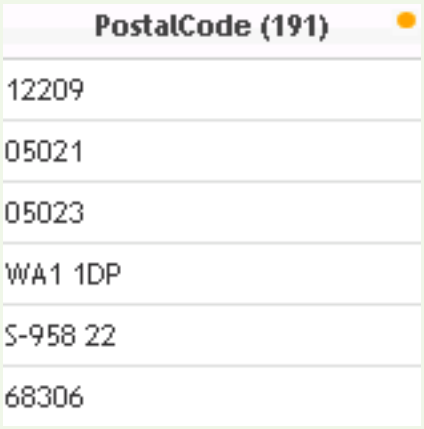
VTS lets you check if the value in a cell is unique in the column.

Right-click the cell that contains the value that you want to check, and then select **Check Uniqueness**. VTS indicates whether or not the value is unique.

### Check if new or modified entries are unique

You can instruct VTS to highlight a new or modified entry if the entry is a duplicate of an existing entry in the column.

**Note:** You must specify this option for each column in which you want to check for duplicate entries. A small orange rectangle in the top right corner of a column title indicates that you have selected to display duplicate entries for that column.




1. Select the column in which you want to highlight duplicate entries.
2. Use the right-click menu or select **Column Actions > Highlight Duplicated Entries**. VTS places a check mark to the left of the **Highlight Duplicated Entries** menu option.

   VTS will highlight all new or modified entries in the column that are duplicates of an existing entry.

   **Note:**
   - Existing duplicate entries are not highlighted.
   - Highlighting of duplicate entries is removed if you display another page and then return to the page with the duplicate entries.

## Importing Data into the VTS Table

VTS lets you import data in the VTS table. You can import data either from a database or from a CSV file.

You can import data multiple times into the same table. Each time you import data, the data is appended to the existing data in the table - the existing data is not deleted. If you want the imported data to be the only data in the table, click **Delete Table** on the VTS toolbar before importing the new data.

For details on exporting data from the VTS table, see "Exporting Data from the VTS Table" on the next page.

## Importing from a database

When you import data from a VTS table to a database, you are required to enter an ADO connection string. The VTS server will connect to the database using this connection string. A typical connection will include the following properties:

- Provider
- Data Source
- Initial Catalog

Refer to http://msdn.microsoft.com/en-us/library/ee252362(v=bts.10).aspx for details about the connection string format.

To import from a database, you run the command line utility **vts_import** from the VTS\bin folder using this syntax:

```
C:\Program Files\Micro Focus\VTS\bin>vts_import –connection “connection_
string” -query “sql_query_string” [-inst “vts_instance_name”]
```

The command line options are:

| | |
|---|---|
| **-connection** | The string that VTS will use to connect to the database. |
| **-query** | The SQL statement to use to query the database. |
| **-inst** | The VTS instance name for which the operation is applied. When not set, it uses the “Default” instance. |

The *provider* specified in the connection string should have already been installed on the Web server.

**Note:** If the connection string specifies the use of windows authentication (for example, Integrated Security=SSPI), the credentials of the Windows service that runs the Web site will be used to access the server.

This may cause a problem when running VTS Service under local machine accounts. If you encounter problems, try to run the **VTS Service** under a domain account that has access to the remote database.

### Importing from a CSV file

When you import data from a CSV file, you must specify the name and location of the CSV file, and the delimiter that is used in the file.

To import data from a CSV file:

1. Click **Import** on the VTS toolbar.
2. In the Import dialog box, click **Import from CSV file**.
3. Click **Select File**, and specify the file that contains the data to import.
4. From the **Delimiter** list, select the delimiter that is used in the CSV file. Keep the **<Auto Detect>** option to let VTS determine the delimiter that is used in the CSV file.
5. Click **Start Import** to begin importing the data into the VTS table.

### Using the sample files

Sample tables are provided (Customers, Employees, Products, Orders) which can be imported into any VTS instance. Additionally, you can add your own .csv files to the **Samples** folder.

**How to import a sample table**

1. Click **Options > Import Samples**
2. Select one of the sample tables

**How to add and import files from the samples directory**

1. Copy your .csv file into the **Samples** folder. By default, the folder location is **C:\Program Files\Micro Focus\VTS\web\Samples**.
2. Click **Options > Import Samples**. Select your .csv file from the drop-down list to populate the data table.

## Exporting Data from the VTS Table

VTS lets you export data from the VTS table. You can export data to either a database or a CSV file.

### Introduction

When you export data from a VTS table to a database, you are required to enter an ADO connection string. The VTS server will connect to the database using this connection string. A typical connection will include the following properties:

- Provider
- Data Source
- Initial Catalog

Refer to http://msdn.microsoft.com/en-us/library/ee252362(v=bts.10).aspx for details about the connection string format.

When you export VTS data to a database, you must specify the table into which the data will be exported.

- If the table does not exist, a table will be created that has the same columns as the existing VTS columns. The data is then inserted into the table.
- If the target table already exists, and all VTS columns can be found in the target table, the data will be appended to the existing table. Note that in this case, the default sorting order of inserted data will be affected by the primary key and the clustered index. If there is no clustered index for the target table, the order will be the same as the order in the VTS table.
- If the target table already exists, but some VTS columns cannot be found in the target table, the export will be aborted.

## Exporting to a database

To export to a database, you run the command line utility **vts_export** from the VTS\bin folder using this syntax:

```
C:\Program Files\Micro Focus\VTS\bin>vts_export –connection “connection_
string” connection_string” -table “table_name” [-inst “vts_instance_name”]
[-drop_table]
```

The command line options are:

| | |
|---|---|
| **-connection** | The string that VTS will use to connect to the database. |
| **-table** | The table into which the data will be exported. |
| **-inst** | The VTS instance name for which the operation is. When not set, it uses "Default". |
| **-drop_table** | Deletes the existing table. |

Refer to http://msdn.microsoft.com/en-us/library/ee252362(v=bts.10).aspx for details about the connection string format.

The *provider* specified in the connection string should have already been installed on the Web server.

**Note:** If the connection string specifies the use of windows authentication (for example, Integrated Security=SSPI), the credentials of the Windows service that runs the Web site will be used to access the server.

This may cause a problem when running VTS Service under local machine accounts. If you encounter problems, try to run the **VTS Service** under a domain account that has access to the remote database.

### Exporting to a CSV file

1. Click **Export** on the VTS toolbar.
2. In the Export dialog box, click **Export to CSV file**.
3. Click **Start Export** to export the data into a CSV file.

**Caution:** If your table does not have the same number of cells in each column, the export creates empty cells such that each column has the same number of cells. When you import that CSV, the resulting table does not have the same number of cells in each column as the source table. This is because the CSV format does not support null cells.

## Preparing Vuser Scripts for VTS Parameterization

After you record a Vuser script, you can add parameterization functionality to the script, and use VTS to supply the required parameter values. You add VTS's API functions to the Vuser script to enable the Vusers to communicate with VTS. After you include VTS API functions in a Vuser script, you are ready to run the script.

### VTS upgrade scenarios

Significant changes were made to VTS beginning with version 11.52. These changes may create various compatibility issues relating to Vuser scripts and the version of VTS that is used for parameterization when the scripts are replayed. Depending on the version of VTS with which a Vuser script was developed and the version of VTS which is used to perform the parameterization when the Vuser script is replayed, you may be required to perform certain upgrade procedures to the Vuser script to resolve these compatibility issues. The various upgrade scenarios are described below.

| Version of VTS used to develop the Vuser script | Version of VTS used to perform parameterization | Action Required |
|---|---|---|
| 11.52 and earlier | 11.52 and earlier | No code changes are required.<br>The script includes the **vts2.h** header, and dynamically loads the **vtclient.dll** file. |
| 11.52 and later | 11.52 and earlier | No code changes are required. |

Unlike with previous versions of VTS, with the current version, it is not necessary to reference any additional files from the Vuser script.

**Note:** If you are using a Vuser script that was created with a previous version of VTS, it is necessary to modify the Vuser script to enable the script to run with the current version of VTS. For details, see below.

For details on the VTS functions, see "Using the API to Access the VTS Table" on the next page.

### Modifying old Vuser scripts

If you have a Vuser script that was created using a previous version of VTS, you must modify the Vuser script to enable the script to run with the current version of VTS. The required modifications are:

1. Remove all **lr_load_dll("vtclient.dll")** functions from the Vuser script.
2. Remove all instances of **#include "vts2.h"** from all sections of the Vuser script.
3. Update the VTS server name and port number in all **vtc_connect** and **lrvtc_connect** functions in the Vuser script.
4. In the Vuser script, change all instances of **PVCI** to **PVCI2**.
5. In the Vuser script, change all instances of **VTCERR** to **VTCERR2**.

## Upgrading Vuser Scripts for VTS

Significant changes were made to VTS in LoadRunner version 11.52. When upgrading to VTS 11.52 or later, these changes may create various compatibility issues. In order to describe the VTS upgrade options, the following terms will be used to refer to the versions of VTS:

| | |
|---|---|
| **VTS-New** | Versions of VTS that were released with LoadRunner 11.52 or later. VTS-New connects to the new VTS Server that has a web interface. With VTS-New, it is not necessary to reference any additional files from the Vuser script. |
| **VTS-Old** | Versions of VTS that were released prior to the release of LoadRunner 11.52. VTS-Old connects to a Windows application, **vtconsole.exe**, which is the VTS Server. With VTS-Old, the Vuser script must include the **vtc2.h** header file and load the **vtclient.dll** library. |

Depending on the version of VTS with which a Vuser script originally worked, and the version of VTS with which you now want to use the Vuser script, you may be required to perform certain modifications to the Vuser script to resolve the compatibility issues. The various upgrade scenarios are described below.

> **Note:** If you have a Vuser script that originally worked with VTS-Old, it is necessary to modify the Vuser script to enable the script to work with VTS-New. For details, see below.

| Version of VTS with which the Vusers script originally worked | Version of VTS that you now want to use with the Vuser script | Action Required to Upgrade |
|---|---|---|
| VTS-Old | VTS-Old | No changes to the Vuser script are required. The script includes the **vts2.h** header, and dynamically loads the **vtclient.dll** file. |

| Version of VTS with which the Vusers script originally worked | Version of VTS that you now want to use with the Vuser script | Action Required to Upgrade |
|---|---|---|
| VTS-New | VTS-Old | No changes to the Vuser script are required. The script includes the **vts2.h** header, and dynamically loads the **vtclient.dll** file. |
| VTS-New | VTS-New | No changes to the Vuser script are required. The script calls the VTS API's directly from the Vuser script, without including the **vts2.h** header file, and without loading the **vtclient.dll** file explicitly. Unlike with previous versions of VTS, with the current version, it is not necessary to reference any additional files from the Vuser script. |
| VTS-Old | VTS-New | 1. Remove all **lr_load_dll ("vtclient.dll")** functions from the Vuser script.<br>2. Remove all instances of **#include "vts2.h"** from all sections of the Vuser script.<br>3. In the Vuser script, change all instances of **PVCI** to **PVCI2**.<br>4. In the Vuser script, change all instances of **VTCERR** to **VTCERR2**.<br><br>**Note:** If the script includes an **lrvtc_connect** function, the function now returns the error code instead of the PVCI handle. Therefore, check the return code instead of using **vtc_get_last_error** to check the error.<br><br>After you make changes to the script, the script should point to the new VTS server. If the new VTS server is located on different server, update the VTS server name and port number in all **vtc_connect** and **lrvtc_connect** functions in the Vuser script. |

## Using the API to Access the VTS Table

LoadRunner enables you to include API functions in a Vuser script to access the VTS table, and to manipulate the data in the table.

For details on the VTS API functions, see the Function Reference.

> **Note:** Many of these functions have both **lrvtc** and **vtc** versions.
>
> The **lrvtc** functions are LoadRunner integrated calls that automatically evaluate LoadRunner parameters and store results to LoadRunner parameters. It is recommended that you use these **lrvtc** functions in your Vuser scripts if you are using a single VTS server.
>
> The **vtc** functions are raw and application independent functions, for use when working with multiple VTS servers. **These functions have been deprecated** and will be removed in a future version. Therefore, it is recommended to work only with the respective **lrvtc** function.

# VuGen Protocols

VuGen enables you to record a variety of protocols, each suited to a particular load testing environment or topology and resulting in a specific type of Vuser script. For example, you can use a Web - HTTP/HTML Vuser Script to emulate users operating Web browsers. You can use FTP Vusers to emulate an FTP session. The various Vuser technologies can be used alone or together to create effective load tests.

The following table lists the available Vuser protocols, and a brief description of each protocol.

**Note:**

- Protocols with links have expanded information.
- When running tests using protocol-specific Vuser scripts, make sure you have a global license or the required license for your protocol.

| Protocol | Description |
|---|---|
| .NET | Supports the recording of Microsoft .NET client-server technologies. |
| C Vuser | A generic virtual user which uses the standard C library. |
| Citrix ICA | A remote access tool, allowing users to run specific applications on external machines. |
| CoAP (Constrained Application Protocol) | CoAP is one of the two most popular protocols for communicating with IoT message brokers. CoAP is a binary protocol that runs over UDP and is designed to easily translate to HTTP so that common web-based applications can integrate with it.<br><br>The CoAP protocol for LoadRunner allows performance testing against front-end CoAP proxies with the supported methods/verbs to determine scalability and reliability under heavy use.<br><br>**Prerequisite:** Download and install CoAP Protocol for LoadRunner from the AppDelivery Marketplace. |
| DNS (Domain Name Resolution) | The DNS protocol is a low-level protocol that allows you to emulate the actions of a user working against a DNS server.<br><br>The DNS protocol emulates a user accessing a Domain Name Server to resolve a host name with its IP address. Only replay is supported for this protocol—you need to manually add the functions to your script. |

| Protocol | Description |
|---|---|
| Flex | Flex is an application development solution for creating Rich Internet Applications (RIAs) within the enterprise and across the Web. Action Message Format (AMF), is a Macromedia proprietary protocol that allows Flash Remoting binary data to be exchanged between a Flash application and an application server over HTTP. |
| FTP (File Transfer Protocol ) | File Transfer Protocol - a system which transfers files from one location to another over a network.<br>The FTP protocol is a low-level protocol that allows you to emulate the actions of a user working against an FTP server. |
| IMAP (Internet Messaging) | Internet Message Application - a protocol which enables clients to read email from a mail server. |
| Java over HTTP | Designed to record java-based applications and applets. It produces a Java language script using web functions. This protocol is distinguished from other Java protocols in that it can record and replay Java remote calls over HTTP. |
| Java Record Replay | Common Java recorder. |
| Java Vuser | Java programming language with protocol level support. |
| LDAP (Listing Directory Service) | An Internet protocol designed to allow email applications to look up contact information from a server. |
| MAPI (Microsoft Exchange) | Messaging Application Programming Interface designed to allow applications to send and receive email messages. |
| Web - HTTP/HTML (for mobile applications) | Enables the recording of mobile native applications. |

| Protocol | Description |
|---|---|
| MQTester | A third party protocol developed by OpenText company, MQTester enables predicting WebSphere MQ-based system behavior and performance by enabling WebSphere MQ test processes within the LoadRunner environment.<br><br>Record WebSphere MQ events on Windows systems, enhance the generated test script, and play back the script on Windows or UNIX platforms to emulate conditions of heavy user load. No complex changes, design requirements, or re-engineering efforts of existing applications are required to take advantage of these features.<br><br>**Prerequisite:** Download and install MQTester from the AppDelivery Marketplace. |
| MQTT | Enables lightweight, publish/subscribe messaging that is ideal for Internet of Things (IoT) and machine-to-machine (M2M) communications, as well as mobile applications where bandwidth and battery power are at a premium. |
| ODBC | Open Database Connectivity - a protocol providing a common interface for accessing databases. |
| Oracle - 2-Tier | Oracle database using a standard 2-tier client/server architecture. |
| Oracle - Web | The Oracle Applications interface that performs actions over the Web. This Vuser type detects actions on both the API and Javascript levels. |
| Oracle NCA | Oracle 3-tier architecture database consisting of Java client, Web server and database. |
| POP3<br>(Post Office Protocol) | A protocol designed to allow single computers to retrieve email from a mail server. |
| RDP<br>(Remote Desktop Protocol) | A remote access tool using the Microsoft Remote Desktop Connection to run applications on an external machine. |
| RTE<br>(Remote Terminal Emulator) | Emulation of users who submit input to, and receive output from, character-based applications. |
| SAP GUI | An Enterprise Resource Planning system to integrate key business and management processes using the SAP GUI client for Windows. |
| SAP - Web | An Enterprise Resource Planning system to integrate key business and management processes using the SAP Portal or Workplace clients. |
| Siebel - Web | A Customer Relationship Management Application. |

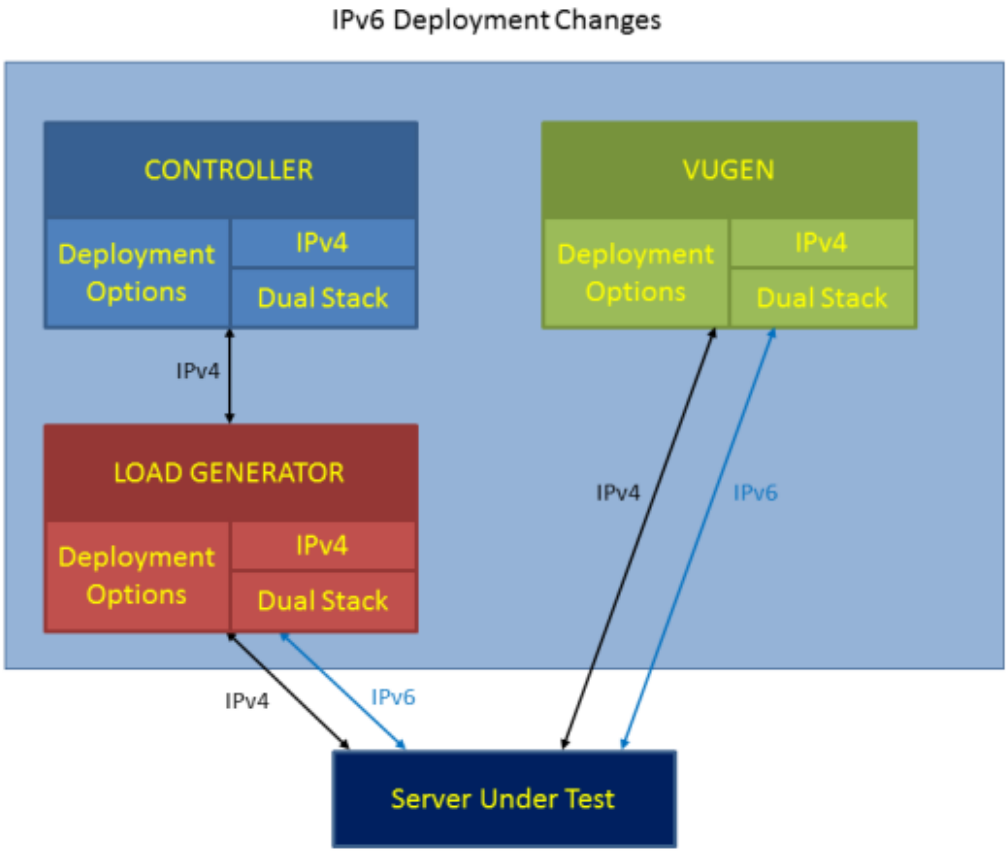
| Protocol | Description |
|---|---|
| SMP (SAP Mobile Platform) | A protocol for recording actions on a mobile SAP application. |
| SMTP (Simple Mail Protocol) | Simple Mail Transfer Protocol - a system for distributing mail to a particular machine. |
| "Teradici PCoIP (PC over IP) Protocol" on page 697 | Supports testing on the Teradici platform. |
| TruClient - Mobile Web | Enables the recording of mobile browser based applications using the TruClient technology. |
| TruClient - Native Mobile | Records native mobile applications using the TruClient technology.<br>For details, select the relevant version in the Mobile Center Help and see the **Performance Testing** section. |
| TruClient - Web | An advanced protocol for modern JavaScript-based applications emulating user activity within a Web browser. Scripts are developed interactively from within a Web browser.<br>For details, go to the TruClient Help Center (select the relevant version). |
| TruWeb | TruWeb protocol focuses on the HTTP (transport) level, providing a lightweight, scalable, and cross-platform (Windows, Linux and MacOS) solution for web protocol testing. |
| Web - HTTP/HTML | Emulation of communication between a browser and Web server on an HTTP or HTML level. |
| Web Services | Web Services are a programmatic interface for applications to communicate with one another over the World Wide Web. |
| Windows Sockets | The standard network programming interface for the Windows platform. |

# IPv6 Support

LoadRunner enables you to test IPv6 based applications in addition to IPv4 based ones. Script recording supports recording for both IPv4 and IPv6 simultaneously. The code that is generated is non-IP specific. With the exception of Web HTTP/HTML protocols, users are unaware which IP version is being used when replaying the script in a load test. Web HTTP protocols have a Runtime setting that allows you to choose between IPv4 and IPv6 for the replay.

## IPv6 Deployment

The internal LoadRunner communication between the Controller and Load Generators uses IPv4 communication. To record and replay in both IPv4 and IPv6, install both VuGen and Load Generator on IPv6-enabled computers, as shown in the diagram below.




For more details about IPv6 related changes, see the Preferences View - Internet Protocol.

## Protocols Supported

For a list of the supported protocols, see the System Requirements. You can also find information here: "Protocol Support for 64-bit Recording, Async, and IPv6" on the next page.

## Protocol Support Limitations

Support for IPv6 is available with the following limitations:

**Web HTTP protocol**

- FTP from Web is not supported
- Web Breakdown is not supported

- Kerberos is not supported
- Spoofing from Web is not supported
- PAC file is not supported

**Webtrace**

- IPv6 Webtrace is not supported on 6to4 outgoing network interfaces.
- IPv6 webtrace does not support RawSocket mode

**General limitations**

- Replay failures may occur because of a IPv4/IPv6 switch between recording and replaying.

## Troubleshooting

**Problem:**

IPV6_webtrace fails to run from a command line with error "webtrace_send_probe_failed". This is caused by an incorrectly installed load generator.

**Solution:**

1. Uninstall LoadRunner or the incorrectly installed load generator.
2. Check if the HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SRPACKET key was removed by the uninstallation process (manually remove it if necessary).
3. Remove all files and folder in the **C:\Micro Focus\LoadGenerator** folder left by the uninstallation process.
4. Restart the machine.
5. Reinstall LoadRunner or the load generator.
6. Check if the `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SRPACKET\ImagePath` value points to the correct path (packet_amd64.sys driver).

# Protocol Support for 64-bit Recording, Async, and IPv6

**Note:**

- The table below includes only the protocols that support at least one of the options.
- For all protocols that support asynchronous sessions, recording will be applied only to web_* steps.

| Protocol | 64-bit recording | Async | IPv6 |
|---|---|---|---|
| Ajax Click & Script | N/A | N/A | ✓ |

| Protocol | 64-bit recording | Async | IPv6 |
|---|---|---|---|
| Citrix | N/A | N/A | ✓ |
| C Vuser | N/A | N/A | ✓ |
| DNS | N/A | N/A | ✓ |
| Flex AMF | ✓ | ✓ | ✓ |
| FTP | ✓ | N/A | ✓ |
| IMAP | N/A | N/A | ✓ |
| Java over HTTP<br>**Note:** 64-bit replay is not supported. | ✓ | N/A | ✓ |
| Java Record Replay<br>**Note:** 64-bit replay is not supported when using web packages. | ✓ | N/A | N/A |
| LDAP | ✓ | N/A | N/A |
| .NET | ✓ | N/A | N/A |
| ODBC | ✓ | N/A | N/A |
| Oracle - 2 Tier | ✓ | N/A | N/A |
| Oracle NCA | ✓ | N/A | ✓ |
| Oracle - Web | ✓ | ✓ | ✓ |
| POP3 | N/A | N/A | ✓ |
| RTE | N/A | N/A | ✓ |
| RDP | ✓ | N/A | ✓ |

| Protocol | 64-bit recording | Async | IPv6 |
|---|---|---|---|
| SAP GUI | N/A | N/A | ✓ |
| SAP – Web | ✓ | N/A | N/A |
| Siebel - Web | ✓ | N/A | N/A |
| SMTP | N/A | N/A | ✓ |
| TruClient - Mobile Web | N/A | N/A | ✓ |
| TruClient - Web | N/A | N/A | ✓ |
| Web - HTTP/HTML (including HTTP/2) | ✓ | ✓ | ✓ |
| Web Services | ✓ | ✓ | ✓ |
| Windows Sockets (multi-protocol) | ✓ | N/A | ✓ |

# Ajax - Click and Script Protocol

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

We recommend that you migrate your existing Ajax Click & Script scripts to another Web protocol such as **TruClient - Web** or **Web - HTTP/HTML**.

For details on these Web protocols, see "Web - HTTP/HTML Protocol" on page 701 or go to the TruClient Help Center (select the relevant version).

Ajax (Asynchronous JavaScript and XML) represents a group of technologies for creating interactive Web applications. With Ajax, web pages exchange small packets of data with the server, instead of reloading an entire page. This reduces the amount of time that a user needs to wait when requesting data. It also increases the interactive capabilities and enhances the usability.

# Ajax (Click and Script) Example Script

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

We recommend that you migrate your existing Ajax Click & Script scripts to another Web protocol such as **TruClient - Web** or **Web - HTTP/HTML**.

For details on these Web protocols, see "Web - HTTP/HTML Protocol" on page 701 or go to the TruClient Help Center (select the relevant version).

VuGen uses the control handler layer to create the effect of an operation on a GUI control. During recording, when one of the supported Ajax controls was encountered, VuGen generated a function with an **ajax_xxx** prefix.

In the following example, a user selected item number 1 (index=1) in an Accordion control. VuGen generated an **ajax_accordion** function.

```
web_browser("Accordion.aspx",
                DESCRIPTION,
                ACTION,
                "Navigate=http://labm1app08/AJAX/Accordion/.aspx",
                LAST);
lr_think_time(5);

ajax_accordion("Accordion",
                   DESCRIPTION,
                  "Framework=atlas",
                  "ID=ctl00_SampleContent_MyAccordion",
                     ACTION,
                  "UserAction=SelectIndex",
                  "Index=1",
                  LAST);
web_edit_field("free_text_2",
        "Snapshot=t18.inf",
        DESCRIPTION,
        "Type=text",
        "Name=free_text",
        ACTION,
        "SetValue=FILE_PATH",
        LAST);
```

**Note:** When you recorded an Ajax session, VuGen generated standard Click & Script functions for objects that are not one of the supported Ajax controls. In the example above, the word

FILE_PATH was typed into an edit box.

## Ajax (Click and Script) - Replay Tips

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

We recommend that you migrate your existing Ajax Click & Script scripts to another Web protocol such as **TruClient - Web** or **Web - HTTP/HTML**.

For details on these Web protocols, see "Web - HTTP/HTML Protocol" on page 701 or go to the TruClient Help Center (select the relevant version).

This section lists tips for replaying click-and-script Vuser scripts.

**Note:** Some of the items below apply to specific click-and-script protocols only.

### Do not re-order statements in a recorded script

Do not change the order of the statements within a recorded script. Also, copying segments of code from one Action to another is not recommended.

### Convert non-ASCII characters

If your links contain non-ASCII characters, you should instruct VuGen to convert the data to or from the UTF-8 format.

### Enable UTF-8 conversion

1. Select **Replay > Runtime Settings** and select the **Internet Protocol > Preferences** node.
2. Click **Options** to open the Advanced Options dialog box.
3. Locate the **Convert from/to UTF-8** option and set it to **Yes**.

Alternatively, view the list of options that is displayed when a link is not found. Enter the displayed text as-is, such as the hex escape sequences \xA0 or any other non-standard format.

### Run the same sequence of actions twice

In some cases, you can perform a certain process only once—such as deleting a user from the database. Replay will fail after the first iteration because the action is no longer valid. Verify that your business process can be repeated more than once with the same data.

### Set unique image properties

In the Step Navigator, double click on the previous image step to open its properties. If the **Id**, **Name**,

and **Alt** properties are empty, provide further identification of the image, such as its file name in the **Src** property.

Alternatively, you can add an **Ordinal** argument to specify the occurrence number of the image on that page. The **Ordinal** argument uniquely identifies each image on the page where all other identification arguments are not unique. For more information, see the Function Reference (**Help > Function Reference**).

### Check the step's description

If you receive a **GUI Object is not found** error, check the Output pane for a list of the objects in the problematic step. In some cases, the object description changes slightly from run to run.

There are several solutions:

- If the new value is stable, open the script in the Editor and manually modify the value of the step's DESCRIPTION argument.
- If the description changes from run to run, you can use a regular expression in the DESCRIPTION argument. For more information, see the Function Reference (**Help > Function Reference**).
- Alternatively, replace the problematic object description property, such as Name, with the Ordinal property. For more information, see the Function Reference (**Help > Function Reference**).

### ThreadingModel

Replay of COM script in VuGen fails when the dll registration is missing the **ThreadingModel** string under the **InprocServer32** folder of the GUID.

## Ajax (Click and Script) Miscellaneous Tips

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

We recommend that you migrate your existing Ajax Click & Script scripts to another Web protocol such as **TruClient - Web** or **Web - HTTP/HTML**.

For details on these Web protocols, see "Web - HTTP/HTML Protocol" on page 701 or go to the TruClient Help Center (select the relevant version).

The following additional tips may help you in troubleshooting problems that you experience with click-and-script Vuser scripts.

**Note:** Some of the items below apply to specific click-and-script protocols only.

### Search for Warnings

Search for warnings or alerts in the Output pane.

### Verify the Response

Verify the response of the previous step is correct using **web_reg_find**. For more information, see the Function Reference (**Help > Function Reference**).

### Use Alternate Navigation

For problematic steps or those using Java applets, use **Alternative Navigation** to replace the Web step with an HTTP level step. Note that the HTTP level steps may require manual correlations. To perform Alternative Navigation, select a step in the **Step Navigator**, or the text in Script View, and select **Replace with alternative navigation** from the right-click menu.

### Working with the Kerberos Protocol

If you are using the Kerberos Protocol for authentication, you must customize VuGen to properly convene authorization sessions. Advanced users can attempt to perform this customization themselves.

In order for the Kerberos Protocol to work properly, create a krb5.ini file and put it in an available folder. Save the full path name of krb5.ini into the KRB5_CONFIG environment variable.

The krb5.ini file should contain detailed information about each domain (KDS and AS addresses) and trust chains.

For more information, contact Software Support.

### API notes

This section lists general notes about the Web functions.

## Regular expressions

You can specify a regular expression for most object descriptions, by preceding the text with "/RE" before the equals sign. For example:

```
web_text_link("Manage Assets",
        DESCRIPTION,
        "Text/RE=(Manage Assets)|(Configure Assets)",
        ACTION,
        "UserAction=Click",
        LAST);
```

See the Function Reference (**Help > Function Reference**) for more details.

## Ordinals

The Ordinal attribute is a one-based index to distinguish between multiple occurrences of objects with identical descriptions. In the following example, the two recorded **web_text_link** functions have identical

arguments, except for the ordinal. The ordinal value of 2, indicates the second occurrence.

```
web_text_link("Manage Assets",
        DESCRIPTION,
        "Text=Manage Assets",
        "FrameName=main",
        ACTION,
        "UserAction=Click",
        LAST);
web_text_link("Manage Assets_2",
        DESCRIPTION,
        "Text=Manage Assets",
    "Ordinal=2",
        "FrameName=main",
        ACTION,
        "UserAction=Click",
        LAST);
```

### Empty strings

There is a difference between not specifying an argument and specifying it as an empty string. When you do not specify an argument, VuGen uses the default value or ignores it. When you list an argument, but assign it an empty string as a value, VuGen attempts to find a match with an empty string or no string at all. For example, omitting the id argument instructs VuGen to ignore the id property of the HTML element. Specifying "ID=" searches for HTML elements with no id property or with an empty ID.

```
web_text_link("Manage Assets_2",
        DESCRIPTION,
        "Text=Manage Assets",
    "Id=",
        "FrameName=main",
        ACTION,
        "UserAction=Click",
        LAST);
```

## Ajax (Click & Script) Enhancements

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

We recommend that you migrate your existing Ajax Click & Script scripts to another Web protocol such as **TruClient - Web** or **Web - HTTP/HTML**.

For details on these Web protocols, see "Web - HTTP/HTML Protocol" on page 701 or go to the TruClient Help Center (select the relevant version).

The following section describes several enhancements that can assist you in creating your script.

Most of the features described below are enhancements to the API functions. For detailed information about the functions and their arguments, see the Function Reference (**Help > Function Reference**) or click F1 on any function.

### Adding conditional steps

The Ajax (Click & Script) functions, **web_xxxx,** allow you to specify conditional actions during replay. Conditions are useful, for example, if you need to check for an element and perform an action only if the element is found.

For example, suppose you perform an Internet search and you want to navigate to all of the result pages by clicking Next. Since you do not know how many result pages there will be, you need to check if there is a Next button, indicating another page, without failing the step. The following code adds a verification step with a notification—if it finds the Next button, it clicks on it.

```
While (web_text_link("Next",
DESCRIPTION,
    "Text=Next",
    VERIFICATION,
    "NotFound=Notify",
    ACTION,    "UserAction=Click",
    LAST) == LR_PASS);
```

For details about the syntax and use of the VERIFICATION section, see the Function Reference (**Help > Function Reference**).

### Text check verification

Using text checkpoints, you can verify that a text string is displayed in the appropriate place on a Web page or application and then perform an action based on the findings. You can check that a text string exists (**ContainsText**),or that it does not exist (**DoesNotContainText**),using exact or regular expression matching.

For example, suppose a Web page displays the sentence "Flight departing from New York to San Francisco". You can create a text checkpoint that checks that the words "New York" are displayed between "Flight departing from" and "to San Francisco". (In this example, you would need to use regular expression criteria.)

To implement these checkpoints, you add the Text Check related arguments to the VERIFICATION section of the step. During replay, Vusers search the innerText of the browser's HTML document and any child frames. The **NotFound** argument specifies the action to take if verification fails, either because the object was not found or because the text verification failed: Error, Warning, or Notify.

You can manually add text verifications to your script for existing steps. Place the text verification after the step that generated the element.

The text validation arguments are valid for the following Action functions: **web_browser**, **web_element**, **web_list**, **web_text_link**, **web_table**, and **web_text_area**.

> **Note:** You can only use the same type of text verification once per step (for example, **ContainsText** twice). If you want to check for multiple texts, separate them into several steps. You can, however, use different verifications in the same step (for example, **ContainsText** =; **DoesNotContainText**). In this case, all conditions have to be met in order for the step to pass.

In the following example, the verification arguments check that we were not directed from www.acme.com to the French version of the website, acme.com/fr.

```
web_browser("www.acme.com",
        ACTION,
        "Navigate=http://www.acme.com/",
    LAST);
web_browser("Verify",
        VERIFICATION,
        "ContainsText=Go to Acme France",
        "DoesNotContainText=acme.com in English",
    LAST);
```

### Saving a Java script value to a parameter

The **EvalJavaScript** argument lets you evaluate Java Script on the Web page.

Suppose you want to click on a link which has the same name as the page title. The following example evaluates the document title and uses it in the next web_text_link function.

```
web_browser("GetTitle",
    ACTION,
    "EvalJavaScript=document.title;",
    "EvalJavaScriptResultParam=title",
    LAST);
web_text_link("Link",
    DESCRIPTION,
    "Text={title}",
    LAST);
```

### Working with custom descriptions

Suppose you want to randomly click a link that belongs to some group. For example, on **mysite.com** you want to randomly select a country. Regular description matching will not allow this type of operation. However, using a custom description argument, you can identify the group with an attribute that is common to all the links in the group.

Using the custom description argument, you specify any attribute of the element, even those that are not predefined for that element. During replay, the Vuser searches for those attributes specified in the DESCRIPTION section. Replay will not fail on any unknown argument in the DESCRIPTION section.

For example, to find the following hyperlink:

```
<a href="yahoo.com" my_attribute="bar">Yahoo</a>
```

use:

```
web_text_link("yahoo",
    DESCRIPTION,
    "Text=yahoo",
    "my_attribute=bar",
    LAST);
```

In the following example, since all the relevant links have the same class name, newmerc-left-ct, you can perform a random click using the following code:

```
web_text_link("Click",
               DESCRIPTION,
               "Class=newmerc-left-ct",
               "Ordinal=random",
               LAST);
```

The following functions do not support the custom description arguments: **web_browser**, **web_map_area**, **web_radio_group**, and **web_reg_dialog**.

### Copying text to the clipboard

VuGen lets you copy text from a browser to the clipboard. This functionality is available in both the Page view and Page Source view. For details on how to copy the text to the clipboard, see "Work with Snapshots" on page 297.

## Ajax Click and Script Troubleshooting and Limitations

This section describes troubleshooting and limitations for click-and-script protocols.

> **Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.
>
> We recommend that you migrate your existing Ajax Click & Script scripts to another Web protocol such as **TruClient - Web** or **Web - HTTP/HTML**.

For details on these Web protocols, see "Web - HTTP/HTML Protocol" on page 701 or go to the TruClient Help Center (select the relevant version).

**Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

### Replay Issues

#### GUI object not found

*Does the error occur at the beginning of the second iteration?*

If the error occurs at the beginning of the second iteration's Action section, it is probably the result of a starting page that was present for the first iteration, but missing for the second one. If the last page in an action does not contain the links and buttons that were available at the start of the iteration, then the next iteration will fail. For example, if the first page has a text link **Book A Flight**, make sure to navigate to the appropriate page, so that the same link will be visible at the end of the business process.

*Is it a text link containing non-ASCII characters?*

If the problem occurs with non-ASCII characters, you should instruct VuGen to covert the data to a suitable character set.

#### Enable Data Conversion on Windows Machines

1. Select **Replay > Runtime Settings** and select the **Internet Protocol > Preferences** node.
2. Click **Options** to open the Advanced Options dialog box.
3. Locate **Charset Conversions by HTTP** in the Web (Click & Script) > General options, and set it to **Yes**.

#### Enable UTF-8 conversion for Linux Machines

1. Select **Replay > Runtime Settings** and select the **Internet Protocol > Preferences** node.
2. Click **Options** to open the **Advanced Options** dialog box.
3. Locate **Convert from/to UTF-8** in the General options and set it to **Yes**

Alternatively, view the list of alternatives that are displayed when a link is not found. Enter the displayed text as-is, such as hex escape sequences \xA0 or any other non-standard format.

*Can you run the same sequence of actions twice in the application?*

In some cases, you can only perform a certain process once, such as deleting a user from the database. Replay will fail after the first iteration, because the action is no longer valid. Verify that your business process can be repeated in the application more than once with the same data.

*Were the image properties 'Id', 'Name' and 'Alt' empty?*

In the **Step Navigator**, double click on the previous image step to open its properties. If the **Id**, **Name**,

and **Alt** properties are empty, provide further identification of the image, such as its file name in the **Src** property.

Alternatively, you add an **Ordinal** argument to specify the occurrence number of the image on that page. The **Ordinal** argument uniquely identifies each image on the page where all other identification arguments are not unique. For more information, see the Function Reference (**Help > Function Reference**).

*Did the step's description change?*

Check the Output pane for a list of the objects in the problematic step. In some cases, the object description changes slightly from run to run.

There are several solutions:

- If the new value is stable, open the Script View and manually modify the value of the step's DESCRIPTION argument(s).
- If the description changes from run to run, you can use a regular expression in the DESCRIPTION argument(s). For more information, see the Function Reference (**Help > Function Reference**).
- Alternatively, replace the problematic object description property, such as Name, with the Ordinal property For more information, see the Function Reference (**Help > Function Reference**).

#### Replay failure

If the replay fails at a particular step, check the step description. VuGen may have interpreted a single space as a double space. Make sure that there are no incorrect double spaces in the string.

#### Replay snapshots

Replay snapshots may differ from the actual Web page.

### Memory Issues

#### Out of memory error in JavaScript

Increase the JavaScript memory in the runtime settings.

### Increase the JavaScript Memory Size

1. Select **Replay > Runtime Settings** and select the **Internet Protocol > Preferences** node.
2. Click **Options** to open the Advanced Options dialog box.
3. Locate the **Memory Management JavaScript Runtime Memory Size (Kb)** and **Memory Management JavaScript Stack Memory Size (Kb)** options.
4. Increase the memory sizes to 512Kb or higher.

   #### VuGen displays JavaScript errors

   If VuGen displays JavaScript errors in the Output pane, enable IE (Internet Explorer) script errors in order to verify that the Javascript itself does not contain errors.

**Show Script Errors**

1. Open Internet Explorer.
2. Select **Tools > Internet Options** and click the **Advanced** tab.
3. Under **Browsing**, select the **Display a notification about every script error** check box.
4. Rerun the application in IE. If IE displays script errors, then there is a problem with the JavaScript application. If it is not possible to fix the application, you can safely ignore the corresponding replay errors.

   **Problems following parameterization**

   If you encounter problems only after you have parameterized values, verify that the values are valid for your application. Perform business process with the value of the parameter and verify that the application accepts it.

**Supported Environments**

- ActiveX objects and Java applets are only supported on Windows platforms.
- Not supported for Macromedia Flash or VB Script.
- Click & Script protocols do not support pop-up windows.

# Citrix Protocol

## Citrix Protocol - Overview

Citrix Vuser scripts emulate Citrix ICA protocol communication between a Citrix client and server. VuGen records all activity during the communication and creates a Vuser script.

Before you can create scripts for the Citrix protocol, you must set up and configure your Citrix environment to work with LoadRunner.

After the system is properly configured, VuGen generates Citrix functions when you perform actions on the remote server. Each function begins with a **ctrx** prefix. These functions emulate the analog movements of the mouse and keyboard. For example:

```
ctrx_mouse_click(44, 318, LEFT_BUTTON, 0, CTRX_LAST);
```

In addition, you can use other **ctrx** functions to synchronize the replay of the actions, by waiting for specific windows to open.

VuGen also allows you to record a Citrix Web interface session. In this case, the Citrix client is installed, but your interface is a browser instead of a client interface. To record Citrix Web interface sessions, you must perform a multi-protocol recording for Citrix and Web Vusers. In multi-protocol mode, VuGen generates functions from both Citrix and Web protocols during recording.

### See also:

- For more information about the syntax and parameters, see the Function Reference (**Help > Function Reference**).

## Set Up Your Citrix Environment

Before creating a Citrix script, you need to properly configure your Citrix client and server.

1. **Prerequisite:** Ensure that you are working with supported versions of your Citrix client and server as listed in the System Requirements.
2. Set up your Citrix ICA clients.
   a. **Install the client on the VuGen and Load Generator machines.** To run your script, you must install a Citrix client on the VuGen machine each Load Generator machine. If you do not have a client installed, you can download one from the Citrix website www.citrix.com under the **Downloads** section.
   b. **Verify the required VuGen Citrix settings**. For some Citrix versions, the **TCP/IP** Citrix connection option does not work. In that case, use the **TCP/IP+HTTP** Citrix connection option:

      In the VuGen Recording Options dialog box, select **Citrix > Login**. In the **Connection** area, select **TCP/IP+HTTP** as the **Network Protocol**. For more details, see "Citrix > Login Recording Options" on page 165.
3. Set up your Citrix Web Interface clients. (Only when connecting to Citrix server via Web Interface)
   a. **DEP.** Before recording, Turn off Microsoft DEP on the VuGen machine. This is done in different ways on different operating systems.
   One way to do this on Vista, Windows 7, and Windows 2008 is as follows:
      i. From the cmd shell, run **bcdedit.exe/set nx AlwaysOff**.
      ii. Restart the computer.
   b. **Security Software.** If possible, disable anti-malware and other security or antivirus software. Alternatively, add an exception to ignore **vugen.exe**, **runcitrixclient.exe** (a LoadRunner process) and **wfica32.exe** (the Citrix client process). Otherwise, your security software may suspect LoadRunner's Web recording engine and prevent LoadRunner from launching your application.
   c. **Disable the desktop toolbar**. The Citrix administrator should disable the desktop toolbar.
   There are several ways to do this. Add `ConnectionBar=0` to the **default.ica** file, or follow the method described in: https://support.citrix.com/article/CTX138928. (Relevant only for published desktops, not published applications)
   d. **Internet Explorer Settings**.
      i. When working with Internet Explorer 8 or later, ensure that the **Enable SmartScreen Filter** option (**Internet Options > Advanced**) is not selected.
      ii. When recording on the Citrix Web Interface from a Windows server operating system, **Internet Explorer Enhanced Security** (IE ESC) must be disabled. This option is set in different locations in different operating systems.

For example, in Windows 2008 server, you set this option from the Server Manager, using the Configure IE ESC option.

e. **VuGen Settings.**

i. Make sure that VuGen is set to create scripts containing only explicit URLs.

In the VuGen Recording Options dialog box, select **General > Recording** and click the HTML Advanced button. In the Script type area, select: **A script containing explicit URLs only**.

ii. Enable the Citrix server SessionToken correlation rules:

In the VuGen Recording Options dialog box, select **Correlations > Rules**, expand the **Citrix_XenApp** node and select the relevant token option for the Citrix version you are using. (If you are not sure which option to select, you can select all of them.)

4. Set your Citrix client to work in non-seamless mode.

LoadRunner can create Citrix scripts only in non-seamless mode (client opens within a Citrix ICA window and not as a local application).

In the Citrix client, select **Preferences > Session Settings** and set **Window size** to **No preference**.

This setting is saved across Citrix settings for the same user. Alternatively , add `TWIMode=Off` in the **default.ica** file on the server.

5. Configure the Citrix server.

a. **Session Disconnect.** By default, when a client times out or disconnects from the Citrix server, the session remains open for a defined time period. However, beginning a run in a Citrix session that has an unpredictable state can cause your test to fail.

Therefore, the Citrix server administrator should configure the Citrix server to end (reset) the client session when a client disconnects for any reason.

b. **Multi-Session Support.** If you are going to run more than one Citrix Vuser on a load generator, ensure that the Citrix server is configured to enable multiple sessions per user/client.

6. Install the LoadRunner Citrix agent on the Citrix server. (Optional)

The LoadRunner Citrix Agent provides an additional level of object-oriented information about the Citrix published application or desktop that enables you to improve the functionality and robustness of your Citrix script. It also provides some extra enhancements over the normal Citrix functionality. For details, see "Agent for Citrix Presentation Server - Overview" on the next page.

a. If you are upgrading the agent, make sure to uninstall the previous version before installing the next one.

b. If your Citrix server requires administrator permissions to install software, log in as an administrator to the server.

c. Ensure that Microsoft DEP is fully disabled. (DEP must be disabled when using the agent.)

d. Locate the installation file, **SetupCitrixAgent.exe**, on your installation disk in the **Additional Components\Agent for Citrix Server** folder.

e. Follow the installation wizard to completion.

f. To enable logging for the agent, set **DebugEnabled=1** in the `[General]` section of CtrxAgent.ini file. The log file will be created in either the Agent's bin\ folder or in session's %TEMP% folder, depending on user permissions.

> **Note:**
>
> - After installation, the agent will be active only for LoadRunner invoked Citrix sessions—it will not be active Citrix sessions started without LoadRunner.
> - To uninstall, from **Add or remove a program**, remove **Software Agent for Microsoft Terminal Server**.
> - The Citrix agent was designed to be cross-compatible, allowing you to use one version of LoadRunner and yet another version of the agent. It is recommended that you use the most recent version of the agent.

## Agent for Citrix Presentation Server - Overview

The Agent for Citrix Presentation Server, or Citrix Agent, is an optional utility that you can install on the Citrix server. It provides enhancements to the normal Citrix functionality. The following sections describe these enhancements.

It is provided in the product's installation disk and you can install it on any supported Citrix server. For more information, see "Install the LoadRunner Citrix agent on the Citrix server. (Optional)" on the previous page For details on supported Citrix server versions, see the System Requirements.

### Object Detail Recording

If the Citrix Agent is installed on the Citrix server, then when you record a Citrix script, VuGen records specific information about the active object instead of general information about the action. For example, VuGen generates **Obj Mouse Click** and **Obj Mouse Double Click** steps instead of the **Mouse Click** and **Mouse Double Click** that it generates without the agent.

The following example shows the same mouse-click action recorded with and without the agent installation. Note that with an agent, VuGen generates ctrx_obj_xxx functions for all of the mouse actions, such as click, double-click, and release.

```
/* Without Agent Installation */
ctrx_mouse_click(573, 61, LEFT_BUTTON, 0, test3.txt - Notepad);
/* With Agent Installation */
ctrx_obj_mouse_click("<text=test3.txt - Notepad class=Notepad>" 573,
         61, LEFT_BUTTON, 0, test3.txt - Notepad=snapshot21, CTRX_LAST);
```

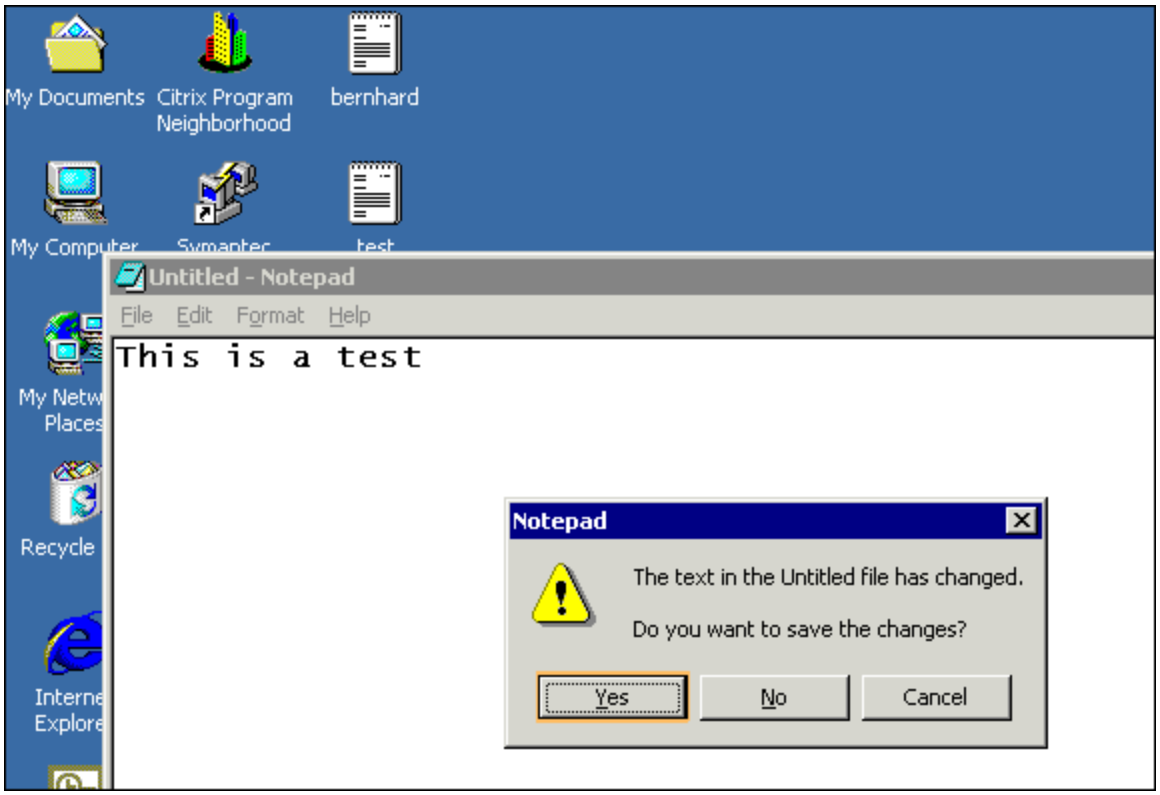
In the example above, the first argument of the **ctrx_obj_mouse_click** function contains the text of the window's title and the class, Notepad. Note that although the agent provides additional information about each object, Vusers only access objects by their window name and the object coordinates.

### Active Object Recognition

The Citrix Agent lets you see which objects VuGen detects in the client window . This includes all Windows Basic Objects such as edit boxes, buttons, and item lists in the current window.

To see the detected objects, you move your mouse through the snapshot. VuGen highlights the borders of the detected objects as the mouse passes over them.

In the following example, the **Yes** button is one of the detected objects.




### Expanded Right-Click Menu

When you click within a snapshot, you can insert several functions into the script using the right-click menu. When no agent is installed, you are limited to the **Insert Mouse Click**, **Insert Mouse Double Click**, **Insert Sync on Bitmap** , and **Insert Get Bitmap Value**. If you are using a 256-color set, the **Insert Sync on Bitmap** and **Get Bitmap Value** steps are not available from the right-click menu.

If the Citrix Agent is installed on the Citrix server, the following additional options are available from the right-click menu of the Citrix window in focus:

- **Obj Get Info** and **Sync on Obj Info**. Provide information about the state of the object: ENABLED, FOCUSED, VISIBLE, TEXT, CHECKED, and LINES.
- **Insert Sync on Obj Info**. Instructs VuGen to wait for a certain state before continuing. This is generated as a **ctrx_sync_on_obj_info** function.
- **Insert Obj Get Info**. Retrieves the current state of any object property. This is generated as a **ctrx_**

**get_obj_info** function.

- **Insert Sync on Text** and **Get Text**.

For more information on the above-mentioned synchronization steps, see "Citrix - Automatic Synchronization" on page 487 and "Citrix - Manual Synchronization" on page 489.

These commands are interactive—when you insert them into the script, you are prompted to mark the object or text area in the snapshot.

In the following example, the **ctrx_sync_on_obj_info** function provides synchronization by waiting for the Font dialog box to come into focus.

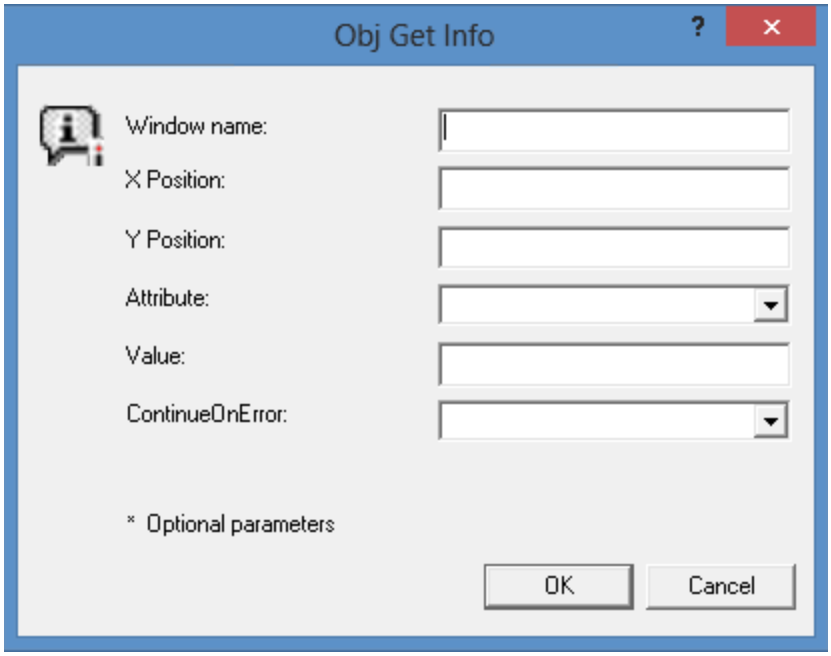
```
ctrx_sync_on_obj_info("Font", 31, 59, FOCUSED, "TRUE", CTRX_LAST);
```

Utilizing VuGen's ability to detect objects, you can perform actions on specific objects interactively, from within the snapshot.

### Insert a Function Interactively Using the Agent Capabilities

1. Click at a point within the Step Navigator to insert the new step. Make sure that a snapshot is visible.
2. Click within the snapshot.
3. To mark a bitmap:
   a. Select it and choose **Insert Sync on Bitmap** from the right-click menu.
   b. Mark the required area.
   c. Click **OK** in the Sync on Bitmap dialog box. VuGen inserts the step into the script after the currently selected step.
4. For all other steps, move your mouse over snapshot objects to determine which items are active—VuGen highlights the borders of active objects as the mouse passes over them.

   When the object is highlighted, right-click and select one of the Insert commands. A dialog box opens with the step's properties.




Set the desired properties and click **OK**. VuGen inserts the step into your script.

### Text Retrieval

With the agent installed, VuGen lets you save standard text to a buffer. Note that VuGen can only save true text—not a graphical representation of text in the form of an image.

You save the text using the **Get Text** step either during or after recording.

For additional details, see "Citrix - Manual Synchronization" on page 489.

## Citrix Recording Tips

Before recording a Citrix Vuser script, make sure your environment is set up as described in "Set Up Your Citrix Environment" on page 479, and then consider these guidelines:

### Plan before recording

Make sure you have a well-defined business process planned, and run through it before recording it in VuGen.

### Avoid production environments if possible

Try to load test Citrix applications which are restricted (contained) to a few Citrix servers in a Citrix development/test “farm” rather than to load test in a live Citrix production environment.

### Consider creating your first test using single-protocol script before testing the Web interface

Consider using a single-protocol Citrix ICA script to load test the ICA elements of your application before expanding the test to address the Web interface. Once you have stabilized the ICA steps of your

script, it may be easier to troubleshoot any problems that arise in your multi-protocol script.

### Record into appropriate sections

Record the connection process into the **vuser_init** section, and the closing process into the **vuser_end** section. This will prevent you from performing iterations on the connecting and disconnecting. For more information about recording into sections, see "Vuser Script Sections" on page 155.

### Ensure a clean session

When recording a session, make sure to perform the complete business process, starting with the connection and ending with the cleanup. End your session at a point from where you could start the entire process from the beginning. Do not leave any client or application windows open.

### Use explicit clicks

When opening expanded menu options, click explicitly on each option—do not depend on the expanding menu. For example, when choosing **Start > All Programs > Microsoft Word**, be sure to click on the line **All Programs**.

### Do not resize windows

To ensure exact reproduction of recorded actions, avoid moving or resizing windows while recording. If it is absolutely necessary to change the size or position of a window, double-click on the relevant **Sync on Window** step in the **Step Navigator** and modify the window's coordinates. This will often, but not always give the desired replay results.

### Make sure resolution settings are consistent

To ensure successful bitmap synchronization, make sure that the resolution settings match. On the recording machine, check the settings of the Citrix client, the Recording Options, and the runtime settings. On the load generators, check the settings of the Citrix client, and make sure that they are consistent between all load generators and recording machines. If there is an inconsistency between the resolutions, the server traffic increases in order to make the necessary adjustments.

### Add Manual Synchronization Points

If it is necessary to wait for an event during recording, such as the opening of an application, add manual synchronization points, such as **Sync on Text** (if using the Citrix agent) or **Sync on Bitmap**. For details, see "Citrix - Manual Synchronization" on page 489.

### Use Classic Windows Style

For **Sync on Bitmap** steps, record windows in the "classic" windows style—not the XP style. To do this:

1. Click in the desktop area.
2. Select **Properties** from the right-click menu.
3. Click the Theme tab.
4. From the **Theme** list, select **Windows Classic**.
5. Click **OK**.

### Modify or disable DEP

DEP (Data Execution Prevention) is a security feature included in Windows. It can interfere with some of the Citrix Agent's functionality during record and replay, and may cause Internet Explorer to hang on the VuGen machine. If you experience unusual behavior during recording, modify the DEP settings.

1. Open **Start > Control Panel > System and Security > System > Advanced system settings**. The System Properties dialog box opens.
2. Select the **Advanced** tab, click the **Settings** button in the **Performance** section.
3. In the Performance Options dialog box, click the **Data Execution Prevention** tab. Select the first option, **Turn on DEP for essential Windows programs and services only**.

   If you cannot change this option, click **Add**. Browse to the client program, for example IEXPLORE.EXE and click **Open** to add the application to the exception list.
4. If neither of the above options are possible, try to disable DEP completely.
   a. Open a command prompt.
   b. Run the following command: **bcdedit.exe /set {current} nx AlwaysOff**
   c. Reboot the machine.
   d. Verify that the settings took effect by running the following at the command line: **BCDEdit /enum**
   e. Verify that the last line shows **nx AlwaysOff**.

### Disable Active X

If the application containing the Citrix session is a native Citrix binary file, disable AcitveX controls in order to prevent execution of the Citrix Online plugin. To disable ActiveX, in Internet Explorer go to **Tools > Internet Options >Security** tab. Click **Custom level** and under **Run ActiveX controls and plug-ins** select **Disable**. Make sure that ActiveX is disabled for the Internet Zone to which the Citrix WebInterface site belongs.

If you cannot change above settings, try to suppress the loading of the Citrix ActiveX object into the browser's process by setting the **SuppressCitrixOcxEnabledto** flag to true in **LR\config\bbhook.ini**.

### Disable Client Updates

Disable client updates when prompted by the Citrix client. This will prevent forward compatibility issues between VuGen and newer Citrix clients that are not currently supported. (For details on supported versions, see the System Requirements.)

For more help, see "Citrix - Troubleshooting and Limitations" on page 495.

## Citrix Synchronization

Synchronization refers to waiting for windows and objects to become available before executing an action. This is necessary when recording Citrix scripts because, for example, if a step in a script opens a window, and the next step performs an action in that window, the second step cannot be implemented

until the window opens. In order to ensure that VuGen does not replay the script incorrectly, it automatically generates functions that synchronize the script by waiting for windows or objects to become available. In addition, you can add synchronization functions manually.

To choose the most suitable synchronization mechanisms for a particular use-case scenario, it is crucial to understand the benefits and drawbacks of the different synchronization mechanisms:

- **Bitmap synchronization (Exact level).** The fastest, most reliable, and least consuming synchronization operation. Should be used by default.
- **Bitmap synchronization (non-Exact levels).** Consumes additional resources of the load generator machine. In general, this mechanism is not recommended unless you specifically know that you need it for a particular purpose.
- **_obj_ sync functions.** Might be a reasonably good way to improve script stability but requires Agent presence on the server side, which cannot always be achieved. Also creates additional traffic to the server (this should not be an issue but should be at least considered).
- **_text.** Same as _obj_. Might not be supported for some applications or technologies.
- **_ocr_.** Use with caution! Consumes significant amounts of hardware resources on the load generator machine.

Exact-level bitmap synchronization should cover most use-case scenarios. Other synchronization mechanism usage should be properly motivated.

For more information about **_text** and **_ocr** functions, see the relevant Synchronization Functions in the Function Reference.

### See also:

- "Citrix - Automatic Synchronization" below
- "Citrix - Manual Synchronization" on page 489

## Citrix - Automatic Synchronization

During recording, VuGen automatically generates steps that help synchronize the Vuser's replay of the script.

### Sync on Window

The **Sync on Window** step instructs the Vuser to wait for a specific event before resuming replay. The available events are **Create** and **Active**. The Create event waits until the window is created. The Active event waits until the window is created and then activated (in focus). Usually VuGen generates a function with a Create event. If, however, the next instruction is a keyboard event, VuGen generates a function with an Active event.

In the Editor, the corresponding function call to the **Sync on Window** step is **ctrx_sync_on_window**.

### Sync on Obj Info

The **Sync on Obj Info** step instructs the Vuser to wait for a specific object property before resuming replay. The available attributes are **Enabled**, **Visible**, **Focused**, **Text**, **Checked**, **Lines**, or **Item**. The Enabled, Visible, Focused, and Checked attributes are boolean values that can receive the values **true** or **false**. The other attributes require a textual or numerical object value.

A primary objective of this step is to wait for an object to be in focus before performing an action upon it.

VuGen automatically generates **sync_on_obj_info** steps when the Citrix agent is installed and the Use Citrix Agent Input in Code Generation option is enabled in the Recording Options. By default, this Recording option is enabled. For more information, see "Citrix > Code Generation Recording Options" on page 165.

```
ctrx_sync_on_obj_info("Run=snapshot9", 120, 144, TEXT, "OK",
                                          CTRX_LAST);
```

### Sync on Text

A text synchronization step, **Sync on Text**, instructs the Vuser to wait for a text string to appear at the specified position before continuing. When replaying **Sync on Text**, the Vuser searches for the text in the rectangle whose modifiable coordinates are specified in the step's properties.

> **Note:**
>
> - The maximum allowable length of a text string is 255 characters.
> - The Citrix agent does not support this function on Windows 10, Windows Server 2016, or later versions of Windows.

By default, automatic text synchronization is disabled. However, with the Citrix Agent installed (see "Agent for Citrix Presentation Server - Overview" on page 481), you can instruct VuGen to automatically generate a text synchronization step before each mouse click or double-click. After this option is enabled, it applies both to newly recorded scripts and also to existing scripts if you choose the **Record > Regenerate Script** option. For more information, see "Citrix > Code Generation Recording Options" on page 165.

In the Editor, the corresponding function call to the **Sync on Text** step is **ctrx_sync_on_text_ex**.

The following segment shows a **ctrx_sync_on_text_ex** function that was recorded during a Citrix recording with the Citrix Agent installed and text synchronization enabled.

```
ctrx_sync_on_window ("ICA Seamless Host Agent", ACTIVATE, 0, 0,391,224, "snapshot1",
CTRX_LAST);
ctrx_sync_on_text_ex (196, 198, 44, 14, "OK", "ICA Seamless Host Agent=snapshot2",
CTRX_LAST);
```

```
ctrx_obj_mouse_click ("<class=Button text=OK>", 196, 198, LEFT_BUTTON, 0, "ICA
Seamless Host Agent=snapshot2", CTRX_LAST);
```

For more information on this function, see the Function Reference (**Help > Function Reference**).

See "Citrix - Additional Ways to Synchronize Your Script" on the next page for additional information.

## Citrix - Manual Synchronization

You can manually add synchronization steps both during and after recording. A common use of this capability is where the actual window did not change, but an object within the window changed. Since the window did not change, VuGen did not detect or record a **Sync on Window**.

For example, if you want the replay to wait for a specific graphic image in a browser window, you insert manual synchronization. Or, if you are recording a large window with several tabs, you can insert a synchronization step to wait for the new tab's content to open.

> **Note:** You can also instruct LoadRunner to insert automatic synchronization steps during your recording session. For details, see "Citrix - Automatic Synchronization" on page 487.

### Synchronize manually during recording

To add synchronization during recording, you use the floating toolbar. The **Sync on Bitmap** and **Sync on Text** functions enable you to mark an area or text that needs to be visible within the client window before resuming replay.

- To insert a **Sync on Bitmap** step, click the **Insert Sync on Bitmap** button on the toolbar and mark a rectangle around the desired area.
- To insert a **Sync on Text** step (Citrix Agent required), click the **Insert Sync on Text** button on the toolbar and mark a rectangle around the desired text.

### Synchronize manually after recording

You can also add synchronization after the recording session. To add a synchronization step, right-click in the Snapshot pane, and select a synchronization option:

- **Insert Sync on Bitmap.** Replay waits until the marked bitmap appears. You mark a rectangle around the desired area.
- **Insert Sync on Bitmap (by coordinates).**This option is identical to the one above, except that it allows you to manually enter coordinates for the synchronization area.
- **Insert Sync on Obj Info.** Waits until an object's attributes have the specified values (agent installations only).
- **Sync on Text.** Waits until the specified text is displayed (agent installations only).

## Citrix - Additional Ways to Synchronize Your Script

In addition to automatic and manual synchronization in Citric Vuser scripts, you can add certain other steps that affect the synchronization indirectly:

### Setting a Delay

The **ctrx_set_waiting_time** step sets a waiting time for the other Citrix synchronization functions. This setting applies to all functions that follow it within the script. For example, if your **ctrx_sync_on_window** steps are timing out, you can increase the default timeout from 60 seconds to 180 seconds.

To insert this step, insert a **ctrx_set_waiting_time** step from the **Steps** toolbox.

### Checking if a Window Exists or Closed

The **ctrx_win_exist** step checks if a window is visible in the Citrix client. By adding control flow statements, you can use this function to check for a window that does not always open, such as a warning dialog box. In the following example, **ctrx_win_exist** checks whether a browser was launched. The second argument indicates how long to wait for the browser window to open. If the window did not open in the specified time, it double-clicks the application's icon to open it.

```
if (!ctrx_win_exist("Welcome",6, CTRX_LAST))
   ctrx_mouse_double_click(34, 325, LEFT_BUTTON, 0, CTRX_LAST)
```

To insert this step, insert a **ctrx_win_exist** step from the **Steps** toolbox.

Another useful application for this step is to check if a window has been closed. If you need to wait for a window to close, you should use a synchronization step such as **ctrx_unset_window**.

For detailed information about these functions, see the Function Reference (**Help > Function Reference**).

### Waiting for a Bitmap Change

In certain cases, you do not know what data or image will be displayed in an area, but you do expect it to change. To emulate this, you can use the **ctrx_sync_on_bitmap_change** function. Right-click in the snapshot, and select **Insert Sync on Bitmap** from the right-click menu. VuGen inserts the step or function at the location of the cursor.

The syntax of the functions is as follows:

**Example:**

```
ctrx_sync_on_bitmap (x_start, y_start, width, height, hash, CTRX_LAST);
ctrx_sync_on_bitmap_change (x_start, y_start, width, height, [initial_wait_time,]
[timeout,] [initial_bitmap_value,] CTRX_LAST);
```

The following optional arguments are available for **ctrx_sync_on_bitmap_change**:

- **initial_wait_time**. When to begin checking for a change.

- **timeout**. The amount of time in seconds to wait for a change to occur before failing.
- **initial_bitmap_value**. The initial hash value of the bitmap. Vusers wait until the hash value is different from the specified initial bitmap value.

In the following example, the recorded function was modified and assigned an initial waiting time of 300 seconds and a timeout of 400 seconds.

```
ctrx_sync_on_bitmap_change(93, 227, 78, 52,
                           300,400, "66de3122a58baade89e63698d1c0d5dfa",CTRX_LAST);
```

> **Note:** If you are using **ctrx_sync_on_bitmap**, make sure that the screen settings in the Controller, Load Generator machine, and VuGen are the same. Otherwise, VuGen may be unable to find the correct bitmaps during replay. For information on how to configure the client settings, see"Recording Options" on page 164.

### Failed Bitmap Synchronization Dialog Box

This dialog box enables you to decide what to do when a bitmap synchronization fails.

| | |
|---|---|
| **To access** | This dialog box opens automatically during replay when there is a mismatch between the record snapshot and the replay snapshot in a bitmap synchronization step. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Continue** | Accept the mismatch and use both the original and new snapshots as a basis for comparison between screens during future replays. If replay returns either one of the bitmaps, the Vuser will not fail. |
| **Recording Snapshot** | A view of the recording snapshot. |
| **Replay Snapshot** | A view of the replay snapshot. |
| **Stop** | Consider the mismatch between the snapshots to be an error. This error will be handled like all other errors and halt the execution by default. Alternatively, you can specify **Continue on Error** for a specific function as described in "Citrix - Troubleshooting and Limitations" on page 495. |

## Citrix Replay Tips

Before replaying a Citrix Vuser script, consider these guidelines:

## Wildcards

You can use wildcards (*) in defining window names. This is especially useful where the window name may change during replay, by its suffix or prefix.

In the following example, the title of the **Microsoft Internet Explorer** window was modified with a wildcard.

```
ctrx_mouse_click(573, 61, LEFT_BUTTON, 0,
"Welcome to MSN.com - Microsoft Internet Explorer");ctrx_mouse_click(573, 61, LEFT_
BUTTON, 0,
"* - Microsoft Internet Explorer");
```

For more information, see the Function Reference (**Help > Function Reference**).

## Run as a Process—not a Service

Since the Citrix protocol is GUI-based, it relies on several settings which are imperative for enabling interactivity. The VuGen script was recorded in an interactive session configured with specific screen settings, ClearType options, keyboard layouts, and so forth. If you run the test as a service, which by default, uses the SYSTEM account, the settings will most likely be different. Any mismatch in the above settings may result in a failed replay. Therefore, you should run the test as a process.

## Modify the Citrix client window size for replay

Before replaying your script, you can modify the client window size using the **Citrix > Configuration > Citrix client resolution** runtime setting.

However, if your script contains synchronization calls, changing the resolution may introduce synchronization errors during replay, and your script may fail to run.

**Recommended:** Keep the default **As recorded** value.

## Enable Think Time

For best results, ensure that think time is enabled in the runtime settings. Think time is especially relevant before the **ctrx_sync_on_window** and **ctrx_sync_on_bitmap** functions, which require time to stabilize.

## Set Initialization Quota

To prevent overloading by multiple Vusers while connecting, set an initialization quota of 4 to 10 Vusers (depending on the capacity of the server) or apply ramp-up initialization using the Scheduler.

## Set a Bitmap Polling Delay

To prevent a false failure in bitmap synchronization, set the **Bitmap polling delay** in the runtime settings **Citrix > Synchronization** view, to a non-zero value. A recommended value is 1000 msecs.

### Use Exact Tolerance

It is recommended to always use **Exact** tolerance for synchronization. Set the **Default Image Sync Tolerance** in the runtime settings **Citrix > Synchronization** view, to **Exact**. Setting this option to **Non-exact** is not effective for slight changes, such as a difference in the **ClearType** settings.

### Set Consistency Between Machines

If you intend to replay the script on another machine, make sure that the following items are consistent between the record and replay machines: Window Size (resolution), Window Colors, System Font, ClearType, and the other Default Options settings for the Citrix client. These settings affect the hash value of bitmaps, and inconsistencies may cause replay to fail. To view the Citrix client settings, right-click an item from the Citrix program group and select **Application Set Settings** or **Custom Connection Settings**. (Note that the remote session on the Citrix server inherits the ClearType settings of the local machine.)

### Increasing the Number of Vusers per Load Generator Machine

Load Generator machines running Citrix Vusers may be limited in the number of Vusers that can run, due to the graphic resources available to that machine, also known as the GDI. To increase the number of Vusers per machine, you can open a terminal server session on the machine to act as an additional load generator.

The GDI count is operating system-dependent. For example, the actual GDI (Graphics Device Interface) count for a heavily loaded machine using LoadRunner is approximately 7,500. The maximum available GDI on most Windows operating systems is 16,384.

For more information on creating a terminal server session, see "LoadRunner Terminal Services Manager" on page 1051Terminal Services Overview.

> **Note:** By default, sessions on a terminal server use a 256-color set. If you intend to use a terminal session for load testing, make sure to record on machines with a 256-color set.

## Citrix Debugging Tips

When your test does not run as expected, try these debugging tips:

#### Single Client Installation

If you are unsuccessful in recording any Citrix actions, verify that you have only one Citrix client installed on your machine. To verify that only one client is installed, open the Add/Remove Programs dialog box from the Control Panel and make sure that there is only one entry for a Citrix client.

#### Add Breakpoints

Add breakpoints to your script in VuGen to help you determine the problematic lines of code.

### Synchronize Your Script

If replay fails, you may need to insert synchronization functions into your script to allow more time for the desired windows to come into focus. Although you can manually add a delay using **lr_think_time**, we recommend that you use one of the synchronization functions discussed in "Citrix - Automatic Synchronization" on page 487.

### Regenerate Your Script

During recording, VuGen saves all of the agent information together with the script. By default, it also includes this information in the script, except for the **ctrx_sync_on_text** steps. If you encounter text synchronization issues, regenerate the script to include the text synchronization steps.

In addition, if you disabled the generation of agent information in the recording options, you can regenerate the script to include them.

To regenerate a script, select **Record > Regenerate Script** and select the desired options. For more information about regenerating scripts, see "Regenerate a Vuser Script" on page 239.

### Continue on Error

You can instruct Vusers to continue running even after encountering an error, such as not locating a matching window. You specify Continue on Error for individual steps.

This is especially useful where you know that one of two windows may open, but you are unsure of which. Both windows are legal, but only one will open.

To indicate Continue on Error:

1. In the **Step Navigator**, right-click on the step and select **Show Arguments**. In the **Continue on Error** box, select the **CONTINUE_ON_ERROR** option.
2. In **Script view**, locate the function and add **CONTINUE_ON_ERROR** as a final argument, before **CTRX_LAST**.

This option is not available for the following functions:

- **ctrx_key**
- **ctrx_key_down**
- **ctrx_key_up**
- **ctrx_type**
- **ctrx_set_waiting_time**
- **ctrx_save_bitmap**
- **ctrx_execute_on_window**
- **ctrx_set_exception**

### View the Extended Log

You can view additional replay information such as:

- the total amount of GDI handles being used
- a list of the running Citrix processes (concentr.exe, receiver.exe, wfica32.exe, wfcrun32.exe) with their PIDs and user names (if the process is not running under the LoadRunner user)
- the Citrix client name
- incompatibility warnings

To view these details, enable Extended logging in the runtime settings ( F4 Shortcut key) **Log** tab. You can view this information in the **Output** pane or in the **output.txt** file in the script's folder.

### Save a Snapshot Bitmap

During recording, the bitmaps generated for the **ctrx_sync_on_bitmap** function are saved under the script's **data** folder. The bitmap name has the format of **hash_value.bmp**. If synchronization fails during replay, the generated bitmap is written to the script's output folder, or if you are running it in a scenario, to wherever the output files are written. You can examine the new bitmap to determine why synchronization failed.

### Show Vusers

To show Vusers during a scenario, enter the following in the Vuser command line box: **-lr_citrix_vuser_view**. In the Controller, open the Vuser Details dialog box and click **More** to expand the dialog box. Note that this will affect the scalability of the test, so this should only be done to examine a problematic Vuser's behavior.

To reduce the effect on the script's scalability, you can show the details for an individual Vuser by adding the Vuser's ID at the end of the command line: **-lr_citrix_vuser_view <*VuserID*>**.

To open multiple Vusers, place a comma-separated list of IDs after the command line. Do not use spaces, but you may use commas or dashes. For example, 1,3-5,7 would show Vusers 1,3,4,5, and 7, but would not show Vuser 2, 6, or any Vuser with an ID higher than 7.

When recording with XenApp using a Citrix and Web multi-protocol script, manual modifications may be required to ensure proper recording.

## Citrix - Troubleshooting and Limitations

### General Limitations

- The Citrix registry patch is installed when you record or replay a Citrix Vuser script for the first time. In rare situations, the error log will indicate that it could not be installed. In this case, try to install the registry patch manually. The patch can be found at: **<installation_folder>\dat\Enable_Citrix_API.reg**. To install the registry patch, double-click **Enable_Citrix_API.reg** on the relevant machine.

> **Note:** The Citrix client is 32-bit software. Therefore, on a 64-bit OS, install this registry patch under HKLM\SOFTWARE\Wow6432Node\ , and not under HKLM\SOFTWARE\ . To do this, launch Enable_Citrix_API.reg from the 32-bit file manager or modify it before launching it

> from Windows Explorer."

- Running Citrix Vusers on virtual machines may adversely affect performance due to the sharing of physical resources.
- Text recognition may not work correctly for overlapped windows on Windows 2012 servers.
- On Windows 8.1, in replay, the Start menu may not appear after clicking on the Start button.

  **Workaround:** Add another **ctrx_mouse_click** function into the script below the recorded **ctrx_mouse_click** or **ctrx_obj_mouse_click** functions.
- The Citrix agent does not provide support for Java applications on x86 operating systems.
- If an ICA script succeeds in VuGen, but fails when running on a Load Generator, check the display on the Load Generator machine. If it displays a Citrix dialog box entitled, ICA Client File Security, then, in the Access section of the dialog box, select **Full Access** and **Never ask me again for any application**. Click **OK** to apply your changes.

> **Note:** You can also set these options in **WebICA.ini**. For details, see: http://support.citrix.com/article/CTX568194

- The Citrix agent cannot capture text from Java-based applications or from Internet Explorer 9 and later.
- The Citrix agent cannot capture text from certain Java controls with overlapping text boxes, such as dropdown and combo boxes.
- The recording window size option does not work properly with the Plugin for Hosted Applications 11. The size of the client window is captured, but the server screen resolution is not. This is a Citrix Client limitation and may be fixed in future Citrix Client versions.

  **Workaround**: When recording, set the window size equal to the local screen resolution. When replaying/load testing, set the VuGen or Load Generator screen resolution to equal the resolution used when the script was recorded. To verify the recorded resolution, view the Window property in the **<Script Folder>\default.cfg** file.
- The Citrix Connection Center may prevent record and replay of Citrix ICA scripts, if it is running in a different user session on the same machine.

  **Workaround:** Close all instances of the **concenter.exe** process for all users. To prevent the Citrix Connection Center from starting automatically, set the *ConnectionCenter* registry key to an empty value. This key can be found at:

  **For 32-bit systems:** `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run`
  **For 64-bit systems:** `HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\Curr`
  - On a machine with Citrix Receiver 14.4 installed, it is not possible to connect to the Citrix server during record/replay for a single-protocol Citrix script, if connection details have been specified manually in **Recording Options > Citrix > Login**. (This may also apply for later versions of Citrix Receiver.)
    There is no issue with Citrix Receiver 14.4 for Citrix single-protocol scripts with ICA files, or Citrix multi-protocol scripts.

- On a machine with Citrix Receiver 13.0-14.6 installed, **ctrx_mouse_move()** does not work.

  **Workaround:** Do one of the following:

  - Change the script to use an alternative UI flow. That is, replace mouse-move operations with other operations, such as keyboard operations.
  - Upgrade the Citrix Receiver client (both locally and on the Load Generator machine) to Citrix Receiver version 4.7. Then modify the following Windows registry keys to enable the solution:

| OS | Registry keys |
|---|---|
| 32-bit | **Key location:** LOCAL_MACHINE\SOFTWARE\Citrix\ICA Client\CCM |
| 64-bit | ○ **Key location:** LOCAL_MACHINE\SOFTWARE\Wow6432Node\Citrix\ICA Client\CCM<br>○ **Name:** AllowMouseEvent<br>○ **Type:** REG_DWORD<br>○ **Value:** 1 |

## Effects and Memory Requirements of Citrix Agent

When you run Citrix Vusers with the agent installed, each Vuser runs its own process of **ctrxagent.exe**. This results in a slight reduction in the number of Vusers that can run on the server machine (about 7%).

When the agent is installed, the memory requirements per Citrix Vuser is approximately 4.35 MB. To run 25 Vusers, you would need 110 MBs of memory.

## Random Failures of Functions Accessing Citrix Agent

Communication between Citrix Server and Citrix client-side software is directed via Citrix ICA Virtual Channels. This is a bi-directional connection for the exchange of packet data.

Each Vuser opens its own instance of the Citrix Agent on the server side, and, respectively, its own virtual channel. Citrix Virtual Channels may become unreliable under high load. As a consequence, functions that rely on Citrix Agent API (ctrx_get_text(), ctrx_sync_on_obj_info() etc.) may fail randomly.

**Workaround:** Use a TCP channel for communication with the Citrix Agent. Set the following flags:

`TCPChannel=1` in the `[CITIRX]` section of the script's **default.cfg** configuration file,

`TcpChannelEnabled=1` in the `[ChannelConfig]` section of the **CtrxAgent.ini** file.

Note that for **MinPortValue** and **NumPorts** flags in CtrxAgent.ini, the agent tries to find a free port and enumerates NumPorts ports starting from MinPortValue. If you have firewall software on the Citrix server or load generator, make sure to configure it to allow connections on these ports.

## Citrix Agent will not start

If the Citrix Agent does not start, check that corresponding keys are present in the registry.

In order to be launched during session initialization, Citrix Agent's installer writes it to registry. For servers, it adds it under HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon, and for client machines under HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run.

Also, make sure the Citrix agent's installation folder, usually **C:\Program Files (x86)\Micro Focus\Agent for Citrix Server**, is set to "Read and Execute" and not only "Read".

## Unexpected Disconnection

If you experience "unexpected disconnect" errors, try the following:

- If you suspect this is due to a network issue, you can try the Citrix Session Reliability feature (this can be enabled by the Citrix administrator on the server side). When Session Reliability is enabled, Citrix Client reconnects to the server when the network connection is restored without need for user re-authentication, i.e., transparently for LoadRunner.
- Sometimes the "unexpected disconnect" error may be caused by discrepancy of the script and server timeout settings. Consider the following scenario: The script executes some synchronization function, for example, **ctrx_sync_on_window()**, and waiting time is quite long, say, 180 seconds. The script does not perform any action like mouseclicks or keypresses while it is waiting for the window to appear, and the server disconnects the session when Idle Session Timeout (2 minutes by default) is exceeded. As a result, an "unexpected disconnect" message appears in the replay log. If you get "unexpected disconnect" at the synchronization step, it is recommended to check waiting time value in the script, and session timeouts on the server.

  Another workaround for unexpected disconnect at the synchronization step, is to enable User Activity Simulation - **Runtime settings > Citrix > Synchronization > Enable user activity simulation**. If the feature is turned on, LoadRunner will simulate user activity on a Citrix server over the specified time period and in this way prevent a disconnect.
- It may be a result of connecting to a session that already exists on the server. When a Vuser enters an existing session, it cannot receive Windows events from a Citrix ICA object. This is a limitation of the Citrix software. To prevent this, ask the Citrix administrator to configure sessions on the Citrix server to be terminated immediately after disconnect or log off. In the VuGen script side, make sure to add a **ctrx_logoff()** function at the end of the script (in the vuser_end section).

  To minimize the risk of entering an existing session, Citrix Agent tries to close the session on the server when communication with the client machine is lost. This functionality is available in Citrix Agent version 12.51 and later, and enabled by default. To disable it, set **LogoffSessionOnExit=0** in CtrxAgent.ini.

## Citrix Receiver—Security Warning

The Citrix client may prompt you with a warning "An online application is attempting to access files in your computer". This dialog box blocks the replay because it requires user intervention.

**Workaround:** To prevent this, configure the registry on the Citrix client machine to allow it to silently access local drives, as described in http://support.citrix.com/article/CTX124921.

## Failed to get session from client

This error occurs when the Citrix registry patch (LR\dat\Enable_Citrix_API.reg) is not installed

**Workaround:**Make sure the AllowSimulationAPI key is present in the above registry and not set to 0, as it enables Citrix ICO functionality. Note that in 64-bit operating systems, these keys should reside under the **HKLM\Software\Wow6432Node**, node, since the Citrix client is a 32-bit application.

## Citrix Error 13 "Unsupported Function"

The Citrix Error 13 is a general error code that usually refers to an error for which Citrix do not provide a specific code. This error is most common in Performance Center and BPM environments where Citrix processes (wfica32.exe, wfcrun32.exe, concentr.exe, receiver.exe...) are running in sessions other than that of the **mdrv** process.

**Workaround:** Use TaskManager or ProcessExplorer to find and kill all of these processes.

## Citrix Error 70, Client Error 1030 "Protocol driver error"

This error may occur for several reasons: network issues, proxy configuration, and so forth. Often, it occurs when you are running a Citrix+Web multi-protocol script recorded against a secured (https) Web Interface site, and the certificate required by this site is missing on the Load Generator machine.

**Workaround:** Try to open the published application from the Web Interface on the problematic machine. Look at the log file %APPDATA%\ICAClient\wfcwin32.log and search for “SSL Error 61”. If you find this text, it is clearly a certificate issue. For example,

```
09-18-2014 10:28:55:380 Calculator MUCFARMEXT01: SSL Error 61: You have not chosen to
trust "AddTrust External CA Root", the issuer of the server's security certificate.
```

Compare certificates on the Load Generator and VuGen machines and install the missing one. Make sure that the attributes match—do not rely on a matching certificate name only. You must also check also other attributes such as “Expiration date”.

## Capturing Empty Text

In certain Windows 7 installations, VuGen is unable to capture the actual text during recording. Instead it captures empty text.

**Workaround:**

1. Open **Start > Control Panel > System and Security > System > Advanced system settings**. The System Properties dialog box opens.
2. Select the **Advanced** tab and click the **Settings** button in the **Performance** section.
3. In the Performance Options dialog box, click the **Visual Effects** tab.
4. Clear the check box adjacent to the last option, **Use visual styles on windows and buttons**.

### See also:

- For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

# Click and Script Protocols

> **Note:** From LoadRunner version 12.00 and later, Web (Click & Script) is only supported for replay—not recording.

## Click and Script Troubleshooting and Limitations

This section describes troubleshooting and limitations for click-and-script protocols.

> **Note:** Some of the items below apply to specific click-and-script protocols only.

#### Replay Issues

#### GUI object not found

Does the error occur at the beginning of the second iteration?

If the error occurs at the beginning of the second iteration's Action section, it is probably the result of a starting page that was present for the first iteration, but missing for the second one. If the last page in an action does not contain the links and buttons that were available at the start of the iteration, then the next iteration will fail. For example, if the first page has a text link **Book A Flight**, make sure to navigate to the appropriate page, so that the same link will be visible at the end of the business process.

Is it a text link containing non-ASCII characters?

If the problem occurs with non-ASCII characters, you should instruct VuGen to covert the data to a suitable character set.

#### Enable Data Conversion on Windows Machines

1. Select **Replay > Runtime Settings** and select the **Internet Protocol > Preferences** node.
2. Click **Options** to open the Advanced Options dialog box.
3. Locate **Charset Conversions by HTTP** in the Web (Click & Script) > General options, and set it to **Yes**.

#### Enable UTF-8 conversion for Linux Machines

1. Select **Replay > Runtime Settings** and select the **Internet Protocol > Preferences** node.
2. Click **Options** to open the **Advanced Options** dialog box.
3. Locate **Convert from/to UTF-8** in the General options and set it to **Yes**

Alternatively, view the list of alternatives that are displayed when a link is not found. Enter the displayed text as-is, such as hex escape sequences \xA0 or any other non-standard format.

Can you run the same sequence of actions twice in the application?

In some cases, you can only perform a certain process once, such as deleting a user from the database. Replay will fail after the first iteration, because the action is no longer valid. Verify that your business process can be repeated in the application more than once with the same data, without recording again.

Were the image properties 'Id', 'Name' and 'Alt' empty?

In the **Step Navigator**, double click on the previous image step to open its properties. If the **Id**, **Name**, and **Alt** properties are empty, provide further identification of the image, such as its file name in the **Src** property.

Alternatively, you add an **Ordinal** argument to specify the occurrence number of the image on that page. The **Ordinal** argument uniquely identifies each image on the page where all other identification arguments are not unique. For more information, see the Function Reference (**Help > Function Reference**).

Did the step's description change?

Check the Output pane for a list of the objects in the problematic step. In some cases, the object description changes slightly from run to run.

There are several solutions:

- If the new value is stable, open the Script View and manually modify the value of the step's DESCRIPTION argument(s).
- If the description changes from run to run, you can use a regular expression in the DESCRIPTION argument(s). For more information, see the Function Reference (**Help > Function Reference**).
- Alternatively, replace the problematic object description property, such as Name, with the Ordinal property For more information, see the Function Reference (**Help > Function Reference**).

Did the page load completely during recording?

During recording, it is best to wait for the page to load completely before doing the next step. If you did not wait for all of the pages to load, record the script again.

### Replay failure

If the replay is failing at a particular step, check the step description. VuGen sometimes reads a single space as a double space. Make sure that there are no incorrect double spaces in the string.

## Miscellaneous Issues

### Out of memory error in JavaScript

Increase the JavaScript memory in the Runtime settings.

### Increase the JavaScript Memory Size

1. Select **Replay > Runtime Settings** and select the **Internet Protocol > Preferences** node.
2. Click **Options** to open the Advanced Options dialog box.
3. Locate the **Memory Management JavaScript Runtime Memory Size (Kb)** and **Memory Management JavaScript Stack Memory Size (Kb)** options.
4. Increase the memory sizes to 512Kb or higher.

   **VuGen displays JavaScript errors**

   If VuGen displays JavaScript errors in the Output pane, enable IE (Internet Explorer) script errors in order to verify that the Javascript itself does not contain errors.

### Show Script Errors

1. Open Internet Explorer.
2. Select **Tools > Internet Options** and click the **Advanced** tab.
3. Under **Browsing**, select the **Display a notification about every script error** check box.
4. Rerun the application in IE. If IE displays script errors, then there is a problem with the JavaScript application. If it is not possible to fix the application, you can safely ignore the corresponding replay errors.

   **Problems following parameterization**

   If you encounter problems only after you have parameterized values, verify that the values are valid for your application. Perform business process with the value of the parameter and verify that the application accepts it.

   **Problems with applications that utilize styling actions**

   If the client-side scripts of the application use a lot of styling activities, you should record the script again after enabling the **Record rendering-related property values** option. This enables the recording of additional DOM objects.

### Enable the "Record rendering-related property values" Option

1. Select **Recording > Recording Options** and select the **GUI Properties > Advanced** node.
2. Select the **Record rendering-related property values** check box.

Re-record the Vuser script.

# COM/DCOM Protocol

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

# COM/DCOM Protocol Overview

> **Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

Each COM function begins with an **lrc** prefix. You can configure the programming language in which to create a Vuser script as either C or Visual Basic.

Each COM/DCOM Vuser script contains the following:

- Interface pointer and other variable declarations in the interfaces.h file.
- Function calls that you can define in the vuser_init, actions or vuser_end sections of the Vuser file.
- A user.h file containing the translation of the Vuser script into low level calls.

# COM/DCOM Technology Overview

> **Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

This section provides an outline of COM technology. This should be enough to get you started with COM Vuser scripts. See Microsoft Developer's Network (MSDN) and other documentation for further details.

COM (Component Object Model) is a technology for developing reusable software components ("plug-ins"). DCOM (Distributed COM) allows use of COM components on remote computers. Microsoft transaction servers (MTS), Visual Basic and Explorer all use COM/DCOM technology. Thus, the application you are testing may use COM technology indirectly, even though you don't know it. You will probably have to include some, but certainly not all, of the COM calls made by your application in the Vuser script.

## Objects, Interfaces, and Type Libraries

COM objects are binary code modules. Each COM object implements one or more interfaces that allow client programs to communicate with it. You need to know about these interfaces in order to follow the COM calls in the Vuser scripts. Type libraries, used as a reference for accessing COM interface methods and parameters, contain descriptions of COM objects and interfaces. Each COM class, interface, and type library is identified by a Global Unique Identifier (GUID).

## COM Interfaces

A COM interface provides a grouped collection of related methods. For example, a **Clock** object may have **Clock**, **Alarm** and **Timer** interfaces. Each interface has one or more methods. For example the **Alarm** interface may have **AlarmOn** and **AlarmOff** methods.

An interface may also have one or more properties. Sometimes, the same function may be performed by calling a method or by setting or getting the value of a property. For example, you can set the **Alarm Status** property to **On** or call the **AlarmOn** method.

A COM object may support many interfaces. The **IUnknown** interface is implemented by all components and is used to find out about other interfaces. Many components also implement the **IDispatch** interface, which exposes all other interfaces and methods of the object, allowing implementation of COM automation in scripting languages.

### COM Class Context and Location Transparency

COM objects can run on the same machine as the client application, or on a remote server. COM objects that an application creates may be in a local library, a local process or a remote machine ("Remote Object Proxy"). The location of the COM object, known as the "Context," can be transparent to the application. Most users apply the Vusers to check the load on remote servers. Therefore, objects accessed by Remote Object Proxy are usually the most relevant for these tests.

### COM Data Types

COM also provides several special data types, including safe arrays, BSTR strings and variants. You may need to use these data types for debugging, parameterization and similar tasks.

## COM/DCOM Vuser Script Structure

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

VuGen COM Vuser scripts are structured in a special way to meet the needs of COM interfaces.

#### Interface Methods

Calls to interface methods have the following names and syntax conventions:

```
lrc_<interface name>_<method name>(instance,...);
```

Note that the instance is always the first parameter passed.

The vendors of the respective COM components usually supply documentation for the interface functions.

#### Interface Pointers

The interface header file defines the interface pointers, as well as other variables, that can be used in the script. Each interface has an Interface ID (IID) which uniquely identifies the interface.

The format of the interface definition is:

```
<interface type>*<interface name> = 0; //"{<IID of the interface type>}"
```

In the following example, the interface type is IDispatch, the name of the interface instance is IDispatch_0, and the IID of IDispatch type is the long number string:

```
IDispatch* IDispatch_0= 0;//"{00020400-0000--C000-000000000046}"
```

### Vuser Script Statements

The COM Vuser script consist of code that creates object instances, retrieves interface pointers and calls the interface methods. Each user action may generate one or more COM calls. Each COM call is coded by VuGen as a group of statements. Each such group is contained in a separate scope enclosed in braces. Several different statements prepare for the main call by assigning values and performing type conversions. For example, the group of calls needed to create an object may look like this:

```
{
GUID pClsid = lrc_GUID("student..1");
IUnknown * pUnkOuter = (IUnknown*)NULL;
unsigned long dwClsContext = lrc_ulong("7");
GUID riid = IID_IUnknown;
lrc_CoCreateInstance(=;pClsid, pUnkOuter, dwClsContext, =;riid, (void**)=;IUnknown_0,
CHECK_HRES);
}
```

### Error Checking

Each COM method or API call returns an error value. VuGen will set a flag to check or not to check errors during replay, depending upon whether the call succeeded during the original recording. The flag appears as the last argument of the function call and has these values:

| | |
|---|---|
| **CHECK_HRES** | This value is inserted if the function passed during recording and errors should be checked during replay. |
| **DONT_CHECK_HRES** | This value is inserted if the function failed during recording and errors should not be checked during replay. |

## COM Sample Vuser Scripts

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

This section shows examples of how VuGen emulates a COM client application. It is divided up into the basic COM script operations. Each type of operation is done within a separate scope.

### Instantiation of the Object

To use a COM object, the application must first instantiate it and get a pointer to an interface of that object.

### VuGen does the following to instantiate an object

1. VuGen calls lrc_GUID to get a unique ProgID for the object, to be stored in pClsid:

   ```
   GUID pClsid = lrc_GUID("student..1");
   ```

   **pClsid** is the unique global CLSID of the object, which was converted from the **student.student.1** ProgID.

2. If the unknown interface pointer is a pointer to an aggregated object, VuGen retrieves the pointer to that object, or else it sets it to NULL:

   ```
   IUnknown * pUnkOuter = (IUnknown*)NULL;
   ```

3. VuGen sets the contexts of the object to be created:

   ```
   unsigned long dwClsContext = lrc_ulong("7");
   ```

   **dwClsContext** contains the context of the object (in process, local, remote or combinations of these.)

4. VuGen sets a variable to hold the requested interface ID, which is IUnknown in this case:

   ```
   GUID riid = IID_IUnknown;
   ```

   **riid** contains the interface ID of the **IUnknown** interface.

5. After the input parameters are prepared, a call to lrc_CoCreateInstance creates an object using the parameters defined in the preceding statements. A pointer to the IUnknown interface is assigned to output parameter IUnknown_0. This pointer is needed for subsequent calls:

   ```
   lrc_CoCreateInstance(=;pClsid, pUnkOuter, dwClsContext, =;riid,
   (void**)=;IUnknown_0, CHECK_HRES);
   ```

   The input parameters were prepared and explained above. Since the call succeeded, VuGen sets error checking on during the user simulation by inserting the **CHECK_HRES** value. The call returns a pointer to the IUnknown interface in **IUnknown_0**, that can be used in subsequent calls.

### Retrieving an Interface

After creating an object, VuGen has access only to the **IUnknown** interface . VuGen will use the **IUnknown** interface for communicating with the object. This is done using the **QueryInterface** method of the **IUnknown** standard interface. The first parameter in a VuGen method call is the interface instance. In this case it is the **IUnknown_0** pointer set previously by **CoCreateInstance**. The

**QueryInterface** call requires as input the ID of the interface to be retrieved, and returns a pointer to the interface designated by that ID.

### Get the Interface

1. First, VuGen sets a parameter, riid, equal to the ID of the Istudent interface:

   ```
   GUID riid = IID_Istudent;
   ```

   A call to QueryInterface assigns a pointer to the Istudent interface to output parameter Istudent_0 if the Istudent object has such an interface:
2.

   ```
   lrc_IUnknown_QueryInterface(IUnknown_0, =;riid, (void**)=;Istudent_0, CHECK_
   HRES);
   ```

### Using an Interface to Set Data

This example shows the how the interface was used to set data. Suppose that in the application, the user supports entering a name. This would activate a method for setting the name. VuGen recorded this in two statements: one statement was used for setting up the name string and the second one set the name property.

### Set up the Entire Function Call

1. First, VuGen sets a variable (Prop Value) equal to the string. The parameter is of type BSTR, a string type used in COM files:

   ```
   BSTR PropValue = lrc_BSTR("John Smith");
   ```

   In subsequent stages, you will probably parameterize this call, replacing "John Smith" with a parameter, so that different names are used each time the Vuser script is run.

2. Next, VuGen calls the Put_Name method of the Istudent interface to enter the name:

   ```
   lrc_Istudent_put_name(Istudent_0, PropValue, CHECK_HRES);
   ```

### Using an Interface to Return Data

Returning data from an application is different than entering the data, because you might want to store these values and use them as inputs in subsequent calls for parameterization.

### The following is an example of what VuGen may do when the application retrieves data

1. Create a variable of the appropriate type (in this case a BSTR) that will contain the value of the property.

```
BSTR pVal;
```

2. Get the value of the property, in this case a name, into the **pVal** variable created above, using the get_name method of the **Istudent** interface in this example.

```
lrc_Istudent_get_name(Istudent_0, =;pVal, CHECK_HRES);
```

3. VuGen then generates a statement for saving the values.

```
//lrc_save_BSTR("param-name",pVal);
```

The statement is commented out. You can remove the comments and change <param-name> to a variable with a meaningful name to be used for storing this value. VuGen will use the variable to save the value of **pVal** returned by the previous call. You can then use the variable as a parameterized input in subsequent calls to other methods.

### The IDispatch Interface

Most COM objects have specific interfaces. Many of them also implement a general-purpose interface called **IDispatch**, which VuGen translates in a special way. IDispatch is a "superinterface" that exposes all of the other interfaces and methods of a COM object. Calls to the **IDispatch:Invoke** method from VuGen scripts are implemented using **lrc_Disp** functions. These calls are constructed somewhat differently from calls to other interfaces.

The **IDispatch** interface **Invoke** method can execute a method, it can get a property value, or it can set a value or reference value for a property. In the standard **IDispatch:Invoke** method these different uses are signaled in a **wflags** parameter. In the VuGen implementation they are implemented in different procedure calls that invoke a method or put or get a property.

For example, a call to IDispatch to activate the GetAgentsArray method may look like this:

```
retValue = lrc_DispMethod1((IDispatch*)IDispatch_0, "GetAgentsArray", /*locale*/1033,
LAST_ARG, CHECK_HRES);
```

The parameters in the above call are:

| | |
|---|---|
| **IDispatch_0** | This is the pointer to the **IDispatch** interface returned by a previous call to the **IUnknown:Queryinterface** method. |
| **GetAgentsArray** | This is the name of the method to invoke. Behind the scenes, VuGen will get the ID of the method from the name. |
| **1033** | This is the language locale. |

| | |
|---|---|
| **LAST_ARG** | This is a flag to tell the **IDispatch** interface that there are no more arguments. |
| **CHECK_HRES** | This flag turns on checking of HRES, since the call succeeded when it was recorded. |

In addition, there might be another parameter, OPTIONAL_ARGS. This signals that in addition to any standard parameters, VuGen is sending some optional arguments. Each optional argument consists of a pair giving the ID or name of the argument and its value. For example, the following call to lrc_ DispMethod passes optional arguments "#3" and "var3":

```
{
        GUID riid = IID_IDispatch;
        lrc_IOptional_QueryInterface(IOptional_0, =;riid, (void**)=;IOptional_0,
CHECK_HRES);
    }
    {
        VARIANT P1 = lrc_variant_short("47");
        VARIANT P2 = lrc_variant_short("37");
        VARIANT P3 = lrc_variant_date("3/19/1901");
        VARIANT var3 = lrc_variant_scode("4");
        lrc_DispMethod((IDispatch*)IOptional_0, "in_out_optional_args",
/*locale*/1024, =;P1, =;P2, OPTIONAL_ARGS, "#3", =;P3, "var3", =;var3, LAST_ARG,
CHECK_HRES);
}
```

The different **lrc_Disp** methods that use the **IDispatch** interface are detailed in the Function Reference (**Help > Function Reference**).

### Type Conversions and Data Extraction

As shown in the above example, many COM parameters are defined as variants. To extract these values, VuGen uses a number of conversion functions, derived from the equivalent COM functions. The full list is given in the Function Reference (**Help > Function Reference**). Previously, we showed how the **lrc_ DispMethod1** call was used to retrieve an array of name strings:

```
VARIANT retValue = lrc_variant_empty();
retValue = lrc_DispMethod1((IDispatch*)IDispatch_0, "GetAgentsArray", /*locale*/1033,
LAST_ARG, CHECK_HRES);
```

The following example now shows how VuGen gets the strings out of **retValue**, which is a variant that will be read as an array of strings.

First, VuGen extracts the BSTR array from the variant:

```
BstrArray array0 = 0;
array0 = lrc_GetBstrArrayFromVariant(=;retValue);
```

With all the values in array0, VuGen provides you with code that you can use to extract the elements from the array for later use in parameterization, as in the example below:

```
//GetElementFrom1DBstrArray(array0, 0); // value: Alex
//GetElementFrom1DBstrArray(array0, 1); // value: Amanda
....
```

VuGen has numerous type conversion functions and functions for extracting conventional types from variants. These are detailed in the Function Reference (**Help > Function Reference**).

# Database Protocols

## Database Protocols Overview

Suppose that you have a database of customer information that is accessed by customer service personnel located throughout the country. You use Database Vusers to emulate the database server servicing many requests for information.

A Database Vuser can:

- Connect to the server
- Submit an SQL query
- Retrieve and process the information
- Disconnect from the server

### Supported protocols

The following database protocols are supported:

- Oracle Two-Tier
- ODBC

The resulting script contains LRD functions that describe the database activity.

### Prerequisites

| Protocol | Recording requirements | |
|---|---|---|
| | **32-bit applications only** | **32- and 64-bit applications** |
| **Oracle Two-Tier** | • 32-/64-bit operating system<br>• 32-bit Oracle client<br>• 32-bit AUTs | • 64-bit operating system<br>• 32 and 64-bit Oracle client<br>• 32 and/or 64-bit AUTs |

| | Recording requirements | |
|---|---|---|
| **Protocol** | **ODBC**<br>• 32-/64-bit operating system<br>• 32-bit ODBC driver<br>• 32-bit AUTs<br>For recording 32- and 64-bit AUTs:<br>• 32-/64-bit operating system<br>• 64-bit ODBC driver<br>• | |

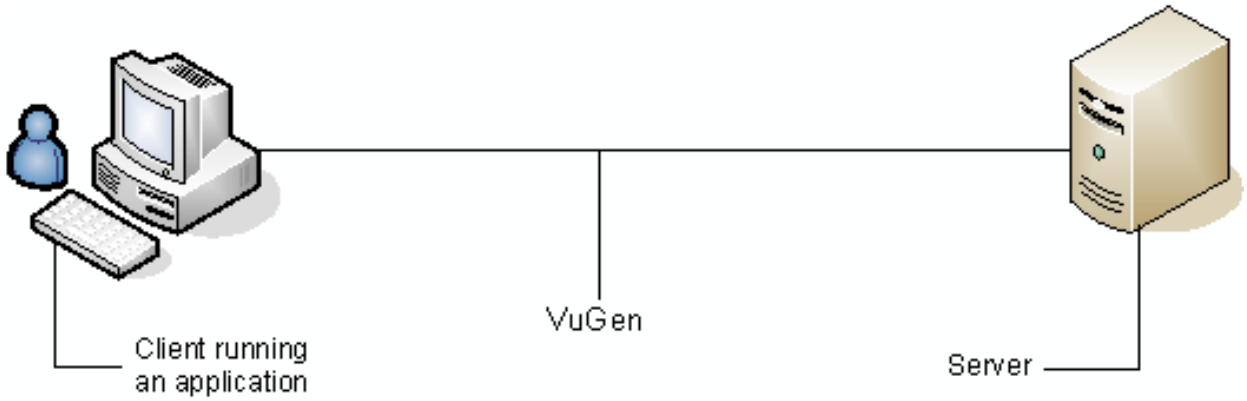
| Protocol | Recording requirements | |
|---|---|---|
| | | 4-bit AUTs |
| Replay on Linux machines requires unixODBC v.2.3.1 or later | | |

### See also:

- "VuGen Database Recording Technology" below
- "Database Grids" on the next page

## VuGen Database Recording Technology

VuGen creates Database Vuser scripts by recording all the activity between a database client and a server. VuGen monitors the client end of the database and traces all the requests sent to and received from the database server.




Like all other Vusers created using VuGen, Database Vusers communicate with the server without relying on client software. Instead, each Database Vuser executes a script that executes calls directly to server API functions.

You create Database Vuser scripts in a Windows environment using VuGen. Once you create a script, you can assign it to Vusers in both Windows and Linux environments.

Users working in a Linux only environment can create Database Vuser scripts through programming using VuGen templates as the basis for a script. For information about programming Database Vuser scripts on Linux, see "Create and Run Scripts in Linux" on page 886.

## Database Grids

When you record or replay a Vuser script, the data that is returned by each query is displayed in a data grid. In a Vuser script, the existence of a data grid is indicated by a **GRID** statement. VuGen displays data grids in either the Data Grids pane or the Snapshot pane.

### Correlate a value in a data grid

Display the data grid in the Snapshot pane, right-click in a cell inside the data grid, and select **Create Correlation**.

### Save the data in a data grid to a file

Display the data grid in the Snapshot pane or the Data Grids pane, right-click in any cell in the data grid, and select **Save Grid To File**.

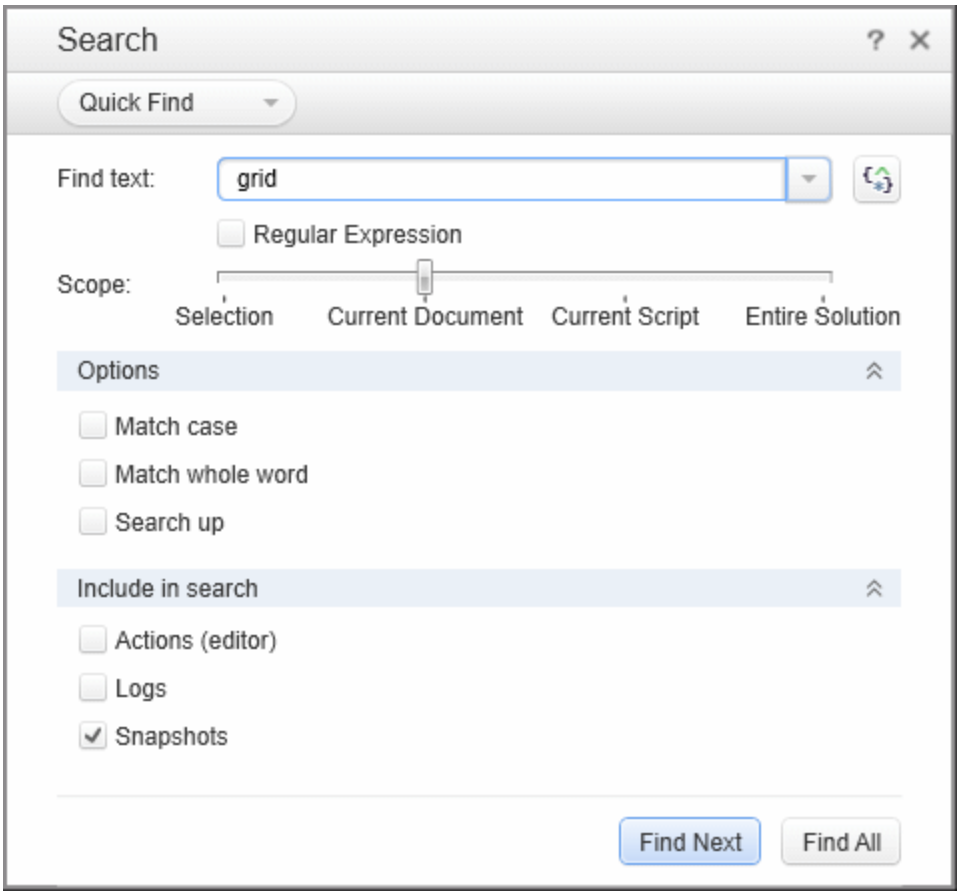
### Copy the text from a cell in a data grid to the clipboard

Display the data grid in the Snapshot pane or in the Data Grids pane, right-click in the cell in the data grid, and select **Copy Selection**.

### Search for data inside a data grid

1. Display the data grid in the Snapshot pane, and click **Search > Quick Find** to open the Search dialog box.

2. Click **Include in Search**, and then select the **Snapshots** check box.




### See also:

- "Work with Snapshots" on page 297
- "Snapshot Pane" on page 85

## Handling Database Errors

You can control how database Vusers handle errors when you run a database Vuser script. By default, if an error occurs during script execution, the script execution is terminated. To change the default behavior, you can instruct the Vuser to continue when an error occurs. You can apply this behavior in different ways as described below.

### Globally Modifying Error Handling

You can change the way that Vusers handle errors by issuing an LRD_ON_ERROR_CONTINUE or LRD_ON_ERROR_EXIT statement. By default, a Vuser aborts the script execution when it encounters any type of error—database, parameter related, and so on. To change the default behavior, insert the following line into your script:

```
LRD_ON_ERROR_CONTINUE;
```

From this point on, the Vuser continues script execution, even when an error occurs.

You can also specify that the Vuser continue script execution when an error occurs only within a segment of the script. For example, the following code tells the Vuser to continue script execution even if an error occurs in the lrd_stmt or lrd_exec functions:

```
LRD_ON_ERROR_CONTINUE;
lrd_stmt(Csr1, "select..."...);
lrd_exec(...);
LRD_ON_ERROR_EXIT;
```

Use the LRD_ON_ERROR_CONTINUE statement with caution, as significant and severe errors may be missed.

### Locally Modifying Error Handling

You can set error handling for a specific function by modifying the severity level. Functions such as **lrd_stmt** and **lrd_exec,** which perform database operations, use severity levels. The severity level is indicated by the function's final parameter, **miDBErrorSeverity**. This parameter tells the Vuser whether or not to continue script execution when a database error occurs (error code 2009). The default, 0, indicates that the Vuser should abort the script when an error occurs.

For example, if you reference a table that does not exist, the following database statement fails and the script execution terminates.

```
lrd_stmt(Csr1, "insert into EMP values ('Smith',301)\n", -1, 1 /*Deferred*/,
        1 /*Dflt Ora Ver*/, 0);
```

To tell a Vuser to continue script execution, even when a database operation error occurs for that function, change the statement's severity parameter from 0 to 1.

```
lrd_stmt(Csr1, "insert into EMP values ('Smith',301)\n", -1, 1 /*Deferred*/,
        1 /*Dflt Ora Ver*/, 1);
```

When the severity is set to 1 and a database error occurs, a warning is issued. Note that the severity level set for a particular function applies only to that function.

## Debugging Database Applications

The following tips apply to database applications only (such as ODBC):

### Generating Debugging Information

> **Note:** You can now set options to view most of the information described in this section using VuGen's user interface.

VuGen contains an inspector "engine." You can force VuGen recorder to create "inspector" output by editing *<install_dir>\config\vugen.ini* as follows:

```
[LogMode]
EnableAscii=ASCII_LOG_ON
```

When this option is enabled, VuGen creates a file, vuser.asc in the Data folder at the end of the recording. Note that this option should be used for debugging purposes only. This output file can become very large (several MB) and have serious effects on machine performance and disk space.

For cases like ODBC-based applications, it is possible to configure the ODBC Administrator (located in the Windows Control Panel) to provide a similar trace output. Open the ODBC options, and select `Trace ODBC calls' to ON. Similarly the ODBC Developer Kit provides a Spy utility for call tracing.

To enable further debug information, add the following section to the *<install_dir>\config\vugen.ini*:

```
[INSPECTOR]
TRACE_LEVEL=3
TRACE_FILENAME=c:\tmp\sqltrace.txt
```

The file (sqltrace.txt) will include useful internal information regarding the hooking calls made during recording. The trace_level is between 1 and 3, with 3 representing the most detailed debug level. Note that in VuGen versions 5.02 and higher, you can set the trace level from the user interface.

### Examining Compiler Information

You can view information about each stage of code generation, preprocessing and compilation to determine the source of any errors.

### Code Generation Information

Look at the vuser.log file under the Data folder. This file, which contains a log of the code generation phase, is automatically created at the end of every lrd recording (i.e. all database protocols).

The following is an example of a log file:

```
lrd_init: OK
lrd_option: OK
lrd_option: OK
lrd_option: OK
Code generation successful
lrd_option: OK
```

```
lrd_end: OK
```

If any of the messages are not OK or successful, then a problem occurred during the code generation.

#### Preprocessing and Compilation Information

During runtime, VuGen displays information about both the preprocessing and compilation processes.

## Database Protocols - Troubleshooting and Limitations

This section describes troubleshooting and limitations for database protocols.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

### Troubleshooting all database protocols

#### Evaluating Error Codes

When a Vuser executes an LRD function, the function generates a return code. A return code of 0 indicates that the function succeeded. For example, a return code of 0 indicates that another row is available from the result set. If an error occurs, the return code indicates the type of error. For example, a return code of 2014 indicates that an error occurred in the initialization.

There are four types of return codes, each represented by a numerical range:

| Type of Return Code | Range |
|---|---|
| Informational | 0 to 999 |
| Warning | 1000 to 1999 |
| Error | 2000 to 2999 |
| Internal Error | 5000 to 5999 |

For more detailed information on the return codes, see the Function Reference (**Help > Function Reference**).

You can evaluate the return code of an LRD function to determine if the function succeeded. The following script segment evaluates the return code of an lrd_fetch function:

```
static int rc;
rc=lrd_fetch(Csr15, -13, 0, 0, PrintRow4, 0);
if (rc==0)
    lr_output_message("The function succeeded");
else
```

```
    lr_output_message("The function returned an error code:%d",rc);
```

## Two-tier Database Scripting Tips

The following section offers solutions for two-tier database scripts.

### Question 1: Why does the script fail when it is data driven, while the same values work with the application itself?

Answer: The failure may be a result of trailing spaces in your data values. Even though the data values that you type directly into the GUI are probably truncated, you should manually eliminate them from your data file. Tab-delimited files can hide trailing spaces and therefore obscure problems. In general, comma-delimited files are recommended. You can view the files in Excel to see if things are correct.

### Question 2: Why does an SQL error of an invalid cursor state occur on the second iteration?

Answer: The **lrd_close_cursor** function may not have been generated or it may be in the *end* section instead of the *action* section. You will need to add a cursor close function or move it from the *end* section to make the script iterate successfully.

Opening a new cursor may be costly in terms of resources. Therefore, we recommend that you only open a cursor once in the *actions* section during the first iteration. You can then add a new parameter that contains the iteration number as a string by using the Iteration Number type. Call this parameter *IterationNum*. Then, inside the *actions* section replace a call to open a new cursor, for example,

```
lrd_open_cursor(=;Csr1, Con1, 0);
```

with

```
if (!strcmp(lr_eval_string("<IterationNum>"), "1"))
lrd_open_cursor(=;Csr1, Con1, 0);
```

### Question 3: How can I fix code produced by VuGen that will not compile because of data declarations in the *vdf.h* file?

Answer: The problem, most likely, is an SQL data type that is not supported by VuGen. For Microsoft SQL, you can often work around this issue by replacing the undefined error message in *vdf.h* with "DT_SZ" (null terminated string). Although this is not the actual datatype, VuGen can compile the script correctly. Please report the problem and send the original script to customer support.

### Question 4: What is the meaning of LRD Error 2048?

Answer: VuGen is failing because it is trying to bind a variable with a longer length than what was allocated during recording. You can correct this by enlarging the variable definition in *vdf.h* to receive a longer string back from the database. Search this file for the unique numeric identifier. You will see its

definition and length. The length is the third element in the structure. Increase this length as required and the script will replay successfully.

For example, in the following script, we have:

```
lrd_assign(=;_2_D354, "<ROW_ID>", 0, 0, 0);
```

In *vdf.h*, we search for *_2_D354* and find

```
static LRD_VAR_DESC _2_D354 = {
LRD_VAR_DESC_EYECAT, 1, 10, LRD_BYTYPE_ODBC,
{0 ,0, 0}, DT_SZ, 0, 0, 15, 12};
```

We change it to:

```
static LRD_VAR_DESC _2_D354 = {
LRD_VAR_DESC_EYECAT, 1, 12, LRD_BYTYPE_ODBC,
{0,, 0}, DT_SZ, 0, 0, 15, 12};
```

The complete definition of LRD_VAR_DESC appears in *lrd.h*. You can find it by searching for `typedef struct LRD_VAR_DESC`.

### Question 5: How can I obtain the number of rows affected by an `UPDATE`, `INSERT` or `DELETE` when using ODBC and Oracle?

Answer: You can use **lrd** functions to obtain this information. For ODBC, use **lrd_row_count**. The syntax is:

```
int rowcount;
.
.
.
lrd_row_count(Csr33, =;rowcount, 0);
```

Note that **lrd_row_count** must immediately follow the pertinent statement execution.

For Oracle you can use the fourth argument of **lrd_exec**.

```
lrd_exec(Csr19, 1, 0, =;rowcount, 0, 0);
```

If you are using Oracle's OCI 8, you can use the fifth argument of **lrd_ora8_exec**.

```
lrd_ora8_exec(OraSvc1, OraStm3, 1, 0, =;uliRowsProcessed, 0, 0, 0, 0, 0);
```

### Question 6: How can I avoid duplicate key violations?

Answer: Occasionally, you will see a duplicate key violation when performing an Insert. You should be able to find the primary key by comparing two recordings to determine the problem. Check whether this or earlier `UPDATE` or `INSERT` statement should use correlated queries. You can use the data dictionary in order to find the columns that are used in the violated unique constraint.

In Oracle you will see the following message when a unique constraint is violated:

```
ORA-00001: unique constraint (SCOTT.PK_EMP) violated
```

In this example `SCOTT` is the owner of the related unique index, and `PK_EMP` is the name of this index. Use SQL*Plus to query the data dictionary to find the columns. The pattern for this query is:

```
select column_name from all_ind_columns where index_name = '<IndexName> and index_
owner = '<IndexOwner>';
select column_name from all_ind_columns where index_name = 'PK_EMP' and index_owner =
'SCOTT';
```

Since the values inserted into the database are new, they might not appear in earlier queries, but they could be related to the results of earlier queries, such as one more than the value returned in an earlier query.

You can use the Query Analyzer to find out which columns used by the key or index. The pattern for this query is:

```
select C.name
    from sysindexes A, sysindexkeys B, syscolumns C
    where C.colid = B.colid and C.id = B.id and
    A.id = B.id and A.indid = B.indid
    and A.name = '<IndexName>' and A.id = object_id('<TableName>')
select C.name
    from sysindexes A, sysindexkeys B, syscolumns C
    where C.colid = B.colid and C.id = B.id and
    A.id = B.id and A.indid = B.indid
    and A.name = 'IX_newtab' and A.id = object_id('newtab')
```

If you still encounter problems, be sure to check the number of rows changed for Updates and Inserts for both recording and replay. Very often, an `UPDATE` fails to change any rows during replay, because the `WHERE` clause was not satisfied. This does not directly result in an error, but it causes a table not to be properly updated, and can cause a later `SELECT` to select the wrong value when correlating the query.

Also verify that there are no problems during multi-user replay. In certain instances, only one user will successfully perform an `UPDATE`. This occurs with Siebel, where it is necessary to manually write a loop to overcome the problem.

**Question 7: The database does not appear to be modified after replaying a script which should have modified the database.**

Answer: Through the user application's UI, check if the updated values appear when trying to see the current data accessible to the application. If the values have not been updated, you need to determine they were not changed. Possibly, an UPDATE statement changed one or more rows when the application was recorded, and did not change any during replay.

Check these items:

- **Verify statement.** If there is a WHERE clause in the UPDATE statement, verify that it is correct.
- **Check for correlations.** Record the application twice and compare the UPDATE statements from each of the recordings to make sure that the necessary correlations were performed.
- **Check the total number of rows.** Check the number of rows that were changed after the UPDATE. For Oracle, this information is stored in the fourth parameter of **lrd_exec**. For ODBC, use **lrd_row_count** to determine the number of rows updated. You can also add code to your script that prints the number of rows that were updated. If this value is 0, the UPDATE failed to modify the database.
- **Check the SET clause.** Check the SET clause of the UPDATE statement. Make sure that you correlated any necessary values here instead of hard-coding them. You can see this by comparing two recordings of the UPDATE.

It certain cases, the UPDATE works when replaying one Vuser, but not for multiple Vusers. The UPDATE of one Vuser might interfere with that of another. Parameterize each Vuser so that each one uses different values during the UPDATE, unless you want each Vuser to update with the same values. In this case try adding retry logic to perform the UPDATE a second time.

**Question 8: How do I avoid the unique column name error when replaying a statement recorded with an Oracle Application. For example:**

```
lrd_stmt(Csr9, "SELECT UOM_CODE, UOM_CODE, DESCRIPTION FROM "
"MTL_UNITS_OF_MEASURE "
"WHERE NVL(DISABLE_DATE, SYSDATE + 1) > "
"SYSDATE ORDER BY UOM_CODE", -1, 1, 1, 0);
```

The following error message was issued:

```
"lrdo.c/fjParse: "oparse" ERROR return-code=960, oerhms=ORA-00960:
ambiguous column naming in select list".
```

Answer: Change the statement by adding an alias to at least one of the non-unique columns, thereby mapping it to a new unique name. For example:

```
lrd_stmt(Csr9,"SELECT UOM_CODE,UOM_CODE second, DESCRIPTION FROM"
```

```
"MTL_UNITS_OF_MEASURE "
"WHERE NVL(DISABLE_DATE, SYSDATE + 1) > "
"SYSDATE ORDER BY UOM_CODE", -1, 1, 1, 0);
```

## Troubleshooting Oracle 2-Tier Vusers

This section contains a list of common problems that you may encounter while working with Oracle Vusers, and suggested solutions.

**ORA-20001 and ORA-06512**

Errors ORA-20001 and ORA-06512 appear during replay when the lrd_stmt contains the pl/sql block: fnd_signon.audit_responsibility(...)

This statement fails during replay because the sign-on number is unique for each new connection.

Solution

In order to solve this problem you need to use the new correlation tool for the sign-on number. This is second assigned value in the statement.

After you scan for possible values to correlate, highlight the value of the second lrd_assign_bind() for the failed statement. Note that the values in the "correlated query" window may not appear in the same order as the actual recorded statements.

The grid containing the substitution value should appear after the lrd_stmt which contains the pl/sql block: fnd_signon.audit_user(...).

> **Note:** Since the sign-on number is unique for every connection, you need to use correlation for each new connection that you record.

Example of Solution

The following statement failed in replay because the second value, "1498224" is the unique sign-on number for every new connection.

```
lrd_stmt(Csr6, "begin fnd_signon.audit_responsibility
(:s,:l,:f,:a,:r,:t,:p)"
"; end;", -1, 1, 1, 0);
lrd_assign_bind(Csr6, "s", "D", =;s_D216, 0, 0, 0);
lrd_assign_bind(Csr6, "l", "1498224", =;l_D217, 0, 0, 0);
lrd_assign_bind(Csr6, "f", "1", =;f_D218, 0, 0, 0);
lrd_assign_bind(Csr6, "a", "810", =;a_D219, 0, 0, 0);
```

```
lrd_assign_bind(Csr6, "r", "20675", =;r_D220, 0, 0, 0);
lrd_assign_bind(Csr6, "t", "Windows PC", =;t_D221, 0, 0, 0);
lrd_assign_bind(Csr6, "p", "", =;p_D222, 0, 0, 0);
lrd_exec(Csr6, 1, 0, 0, 0, 0);
```

The sign-on number can be found in the lrd_stmt with "fnd_signon.audit_user". The value of the first placeholder "a" should be saved. The input of "a" is always "0" but the output is the requested value.

**Modified code:**

```
lrd_stmt(Csr4, "begin fnd_signon.audit_user(:a,:l,:u,:t,:n,:p,:s); end;",
-1, 1, 1, 0);
lrd_assign_bind(Csr4, "a", "0", =;a_D46, 0, 0, 0);
lrd_assign_bind(Csr4, "l", "D", =;l_D47, 0, 0, 0);
lrd_assign_bind(Csr4, "u", "1001", =;u_D48, 0, 0, 0);
lrd_assign_bind(Csr4, "t", "Windows PC", =;t_D49, 0, 0, 0);
lrd_assign_bind(Csr4, "n", "OraUser", =;n_D50, 0, 0, 0);
lrd_assign_bind(Csr4, "p", "", =;p_D51, 0, 0, 0);
lrd_assign_bind(Csr4, "s", "14157", =;s_D52, 0, 0, 0);
lrd_exec(Csr4, 1, 0, 0, 0, 0);
lrd_save_value(=;a_D46, 0, 0, " saved_a_D46");
Grid0(17);
lrd_stmt(Csr6, "begin fnd_signon.audit_responsibility
(:s,:l,:f,:a,:r,:t,:p)"
"; end;", -1, 1, 1, 0);
lrd_assign_bind(Csr6, "s", "D", =;s_D216, 0, 0, 0);
lrd_assign_bind(Csr6, "l", " <saved_a_D46>", =;l_D217, 0, 0, 0);
lrd_assign_bind(Csr6, "f", "1", =;f_D218, 0, 0, 0);
lrd_assign_bind(Csr6, "a", "810", =;a_D219, 0, 0, 0);
lrd_assign_bind(Csr6, "r", "20675", =;r_D220, 0, 0, 0);
lrd_assign_bind(Csr6, "t", "Windows PC", =;t_D221, 0, 0, 0);
```

```
lrd_assign_bind(Csr6, "p", "", =;p_D222, 0, 0, 0);
lrd_exec(Csr6, 1, 0, 0, 0, 0);
```

**Working with large numbers**

Large numbers (NUMBER data type) sometimes appear in different format in the GRID and in the ASCII file. This difference makes it more difficult to identify numbers while searching for values to save for correlation.

For example, you could have a value appear as 1000003 in the grid, but as 1e+0006 in the Recording Log (ASCII file).

Workaround

If you have an error during replay and the correlation tool cannot locate the value in previous results, look for this value in the other format in grid.

**ORA-00960**

This error may occur with non-unique column names. For example:

```
lrd_stmt(Csr9, "SELECT UOM_CODE, UOM_CODE, DESCRIPTION FROM "
"MTL_UNITS_OF_MEASURE "
"WHERE NVL(DISABLE_DATE, SYSDATE + 1) > "
"SYSDATE ORDER BY UOM_CODE", -1, 1, 1, 0);
```

In this case you receive the following error:

```
"lrdo.c/fjParse: "oparse" ERROR return-code=960, oerhms=ORA-00960:
ambiguous column naming in select list".
```

Workaround

Change the statement by adding an alias to at least one of the non-unique columns, thus mapping it to a new unique name. For example:

```
lrd_stmt(Csr9,"SELECT UOM_CODE,UOM_CODE second, DESCRIPTION FROM"
"MTL_UNITS_OF_MEASURE "
"WHERE NVL(DISABLE_DATE, SYSDATE + 1) > "
"SYSDATE ORDER BY UOM_CODE", -1, 1, 1, 0);
```

Alternate Workaround: remove ORDER BY from the lrd statement.

**ORA-2002**

Error 2002 appears when you try to use an unopened cursor. It occurs when you replay a user more than one iteration and you recorded into more than one section of the script.

Specifically, if a cursor is opened in the vuser_init section and closed in the Actions section, then you will encounter this error on the second iteration if you try to use the cursor. This is because it was closed but not re-opened.

For example: You have *lrd_open_cursor* in the vuser_init section and *lrd_close_cursor* in the Actions section. If you replay this user more than one iteration, you are going to get an error in the second iteration because you try using an unopened cursor (it was closed in the first iteration, but not re-opened in the second).

Workaround

The easiest way to solve this is to move the **lrd_close_cursor** or/and **lrd_close_connection** of the problem cursor to the *vuser_end* section.

**Database Protocols (lrd)**

Replay of recorded asynchronous operations is not supported.

**Wrong Client Version**

You may receive an error message when running the wrong Oracle client version:

```
"Error: lrdo_open_connection: "olog" LDA/CDA return-code_019: unable to
allocate memory in the user side"
```

Workaround

You need to modify the library information in the *lrd.ini* file, located in the product's *bin* folder. This file contains the settings that indicate which version of database support is loaded during recording or replay. The file contains a section for each type of host.

For example, the following section of the *lrd.ini* file is for Oracle on Windows NT:

[ORACLE_WINNT]

```
805=lrdo32.dll+ora805.dll
816=lrdo32.dll+oci.dll
815=lrdo32.dll+oraclient8.dll
804=lrdo32.dll+ora804.dll
803=lrdo32.dll+ora803.dll
73=lrdo32.dll+ora73.dll
72=lrdo32.dll+ora72.dll
71=lrdo32.dll+orant71.dll
```

These settings indicate that Vusers should use the **ora805.dll** library if the client uses Oracle 8.0.5, oci.dll for Oracle 8.1.6, and so on.

# Flex (RTMP/AMF) Protocol

## Flex Overview

> **Note:** This topic applies to Flex Vuser scripts only.

The Flex Vuser protocol emulates communication between a client server application that uses the Flex collection of technologies.

### What is Flex?

Flex is a collection of technologies that provides developers with a framework for building RIAs (Rich Internet Applications) based on the Flash Player.

RIAs are lightweight online programs that provide users with more dynamic control than with a standard Web page. Like Web applications built with Ajax, Flex applications are generally more responsive, because the application does not need to load a new Web page every time the user performs an action. However, unlike working with Ajax, Flex is independent of browser implementations such as JavaScript or CSS. The framework runs on Adobe's cross-platform Flash Player.

Flex applications consist of many MXML and ActionScript files. They are compiled into a single SWF movie file which can be played by the Flash player installed on the client's browser.

> **Note:** For Flex applications working with SOAP data, use the **Web Services** Vuser protocol.

### Flex Technologies

The following tables describe the specific technologies that are supported by the VuGen recording engine.

| Technology | Description |
|---|---|
| AMF0 | Action Message Format |
| AMF3 | Action Message Format - Compressed format |
| RTMP | Real Time Message Protocol: Messaging and streaming over TCP |
| RTMPS | Real Time Message Protocol: Messaging and streaming over TCP/SSL |
| RTMPT | Real Time Message Protocol Tunneled: Messaging and streaming over HTTP |

VuGen supports the following development solutions:

| Development Platforms | Description |
|---|---|
| BlazeDS | Open Source Remoting and Messaging solution |
| GraniteDS | Open source development and integration solution for building Flex applications. |
| LiveCycle | Adobe development and integration solution for building Flex applications. |

## Learn more about developing a Flex Vuser script

| Topic | Description |
|---|---|
| Creating a Flex Vuser script | See VuGen's generic documentation about creating Vuser scripts ("Creating Vuser Scripts - Overview" on page 136). |
| Recording | In addition to the generic documentation about recording Vuser scripts ("Recording - Overview" on page 154), see:<br>• "Record a Flex Script" on page 539<br>• "Setting the Flex Recording Mode" on page 540<br>• "RTMP/RTMPT Streaming" on page 531<br>• "RTMP Tunneled" on page 538<br>You may need to configure recording options for your Flex script:<br>• "Flex > RTMP Recording Options" on page 182<br>• "Flex > Externalizable Objects Recording Options" on page 183 |
| Correlating | In addition to the generic VuGen documentation on correlating Vuser scripts ("Correlation Overview" on page 251), see:<br>"Flex Correlations" on page 544 |
| Replaying | In addition to the generic VuGen documentation about replaying Vuser scripts ("Replay Overview" on page 295), see:<br>• "Query an XML Tree" on page 546<br>You may need to configure runtime settings for your Flex script:<br>• **Flex > RTMP** view<br>• **Flex > Externalizable** view<br>• **Flex > Configuration** view |

| Topic | Description |
|---|---|
| Debugging | See the generic documentation about debugging Vuser scripts ("Debugging Overview" on page 328).<br>• "Externalizable Objects in Flex Scripts" on page 542<br>• "Serialize Flex Scripts" on page 545 |
| Viewing Test Results | See the generic documentation about viewing test results ("Replay Summary Pane" on page 131). |

## Recording Flex Scripts

When you record a Flex application, VuGen generates Flex Vuser script functions that emulate the application. The following tables describe the functions that are supported by the Flex protocol.

### AMF

VuGen's **Flex** protocol lets you create scripts that emulate Flex applications working with AMF0 and AMF3.

| Function Name | Description |
|---|---|
| **flex_amf_call** | Sends an AMF request. |
| **flex_amf_define_envelope_header_set** | Defines a set of envelope headers. |
| **flex_amf_define_header_set** | Defines a set of AMF headers. |
| **flex_login** | Logs on to a password-protected Flex application. |
| **flex_logout** | Logs off of a password-protected Flex application. |
| **flex_ping** | Checks if a Flex application is available. |
| **flex_remoting_call** | Invokes one or more methods of a server-side Remote object (RPC). |

#### AMF Example 1:

In the following example, **flex_ping** checks for the availability of a service. The **flex_remoting_call** function invokes the service remotely.

**Example:**

```
flex_ping("1",
    "URL=http://<HOST>/weborb.aspx",
```

```
        "Snapshot=t6.inf",
        LAST);
    flex_remoting_call("getProductEdition::GenericDestination",
        "URL=http://testlab1/weborb30/console/weborb.aspx",
        "Snapshot=t1.inf",
        INVOCATION,
        "Target=/2",
        "Operation=getProductEdition",
        "Destination=GenericDestination",
        "DSEndpoint=my-amf",
        "Source=Weborb.Management.LicenseService",
        "Argument=<arguments/>",
        LAST);
```

### AMF Example 2:

In the following AMF0 example, the **flex_amf_call** function accesses a gateway and sends message to the server.

**Example:**

```
flex_amf_call("EchoAny",
       "Gateway=http://<host>/gateway.aspx",
       "Snapshot=t1.inf",
       "IsParseResponse=No",
       MESSAGE,
       "Method=EchoAMF.EchoAMF.EchoAny",
       "TargetObjectId=/1",
       BEGIN_ARGUMENTS,
             "<boolean>true</boolean>",
       END_ARGUMENTS,
       LAST);
```

### AMF Example 3:

In the following AMF3 example, the **flex_remoting_call** function sends the server an AMF call that can be serialized.

**Example:**

```
          flex_remoting_call(
          "product::getProductsByName",
           URL=http://<HOST>:<PORT>/amf;jsessionid={CorrelationParameter}",
           "Snapshot=t1.inf",
           "IsParseResponse=No",
           INVOCATION,
           "Target=/2",
```

```
            "Operation=getProductsByName",
            "Destination=product",
            "DSEndpoint=my-amf",
            "DSId=8E3759E5-E51A-3906-0EAB-6119CD1E26BF",
            "Argument="
                  "<arguments>"
                        "<string>A</string>"
                  "</arguments>",
            LAST);
```

## RTMP Functions

| Function Name | Description |
|---|---|
| **flex_rtmp_connect** | Connects a client to an RTMP server and sets connection options. |
| **flex_rtmp_disconnect** | Disconnects a client from an RTMP server. |
| **flex_rtmp_receive_stream** | Receives streaming data from an RTMP server. |
| **flex_rtmp_receive** | Receives responses from an RTMP server. |
| **flex_rtmp_send** | Sends a request to an RTMP server. |

### RTMP Example

In the following example, **flex_rtmp_receive** receives data.

```
            flex_rtmp_receive("recv_step0",
            "ConnectionID=19",
            "Snapshot=tRTMP44.inf",
            CHANNEL,
            "ChunkStreamID=2",
            CHANNEL,
            "ChunkStreamID=2",
            LAST);
```

## RTMP Tunneled Functions

| Function Name | Description |
|---|---|
| **flex_rtmp_tunneled_connect** | Connects a client to an RTMP server over HTTP. |
| **flex_rtmp_tunneled_disconnect** | Disconnects a client from session over HTTP with an RTMP server. |
| **flex_rtmp_tunneled_send** | Sends a request to an RTMP server over HTTP. |

### RTMP Tunneled Example

In the following example, **flex_rtmp_tunneled_send** sends an RTMP tunneled request.

```
flex_rtmp_tunneled_send(
 "send_step0",
 "SessionID=0",
 "Snapshot=t30.inf",
 MESSAGE,
 "DataType=command message amf3",
 "ChunkStreamID=3",
 "MessageStreamID=0",
 "Argument="
       "<arguments>"
                          ...
       "</arguments>",
 LAST);
```

For detailed syntax information about all of the Flex functions, see the Function Reference (**Help > Function Reference**).

## RTMP/RTMPT Streaming

VuGen's Flex protocol supports record and replay of streaming data for both the RTMP and RTMPT protocols. You can record using either the regular recording mode or the simplified recording mode. The simplified mode enables VuGen to generate a single function in place of the multiple functions that are generated when the regular recording mode is used.

VuGen also supports RTMPS and RTMPTS in which the streaming data is sent over SSL.

When you use the simplified mode to record, the following occurs:

- For an RTMP-based stream: VuGen generates a single **flex_rtmp_receive_stream** step in place of many **flex_rtmp_receive** and **flex_rtmp_send** steps.
- For an RTMPT-based stream: VuGen generates a single modified **flex_rtmp_tunneled_send** step in place of many **flex_rtmp_tunneled_send** steps.

The single generated **flex_rtmp_receive_stream** or **flex_rtmp_tunneled_send** step makes the Vuser script more readable (by eliminating multiple lines of code), and makes the script replay more reliable. It is recommended that you use the simplified mode for recording your Vuser scripts, unless the Vuser activity includes asynchronous behavior, as described below.

The simplified recording mode is the default recording mode for streaming. To activate the simplified mode, in the Recording Options dialog box, click **Flex > RTMP** and select **Generate single step for RTMP/T stream handling**.

The differences between the simplified and regular recording modes are listed below:

| | Simplified mode | Regular mode |
|---|---|---|
| (Recording option) Generate single step for RTMP/T stream handling check box | Selected | Not selected |
| Functions generated | RTMP: Generates **flex_rtmp_receive_stream** functions.<br>RTMPT: Generates **flex_rtmp_tunneled_send** functions. | Generates **flex_rtmp_receive** and **flex_rtmp_send** steps. |
| Number of functions generated per stream | One | Multiple |
| Supports asynchronous behavior | No | Yes |
| Default Mode | Yes | No |

**Note:** The simplified recording mode is supported for Flash Media Server versions 3.5 and 4.

## Synchronous Vuser behavior

The **flex_rtmp_receive_stream** and **flex_rtmp_tunneled_send** functions that are generated when the simplified recording mode is selected are synchronous functions. This means that no other Vuser functions can be executed while either of these functions is executing. For example, consider a Vuser script that includes a **flex_rtmp_receive_stream** function that streams a video for 5 minutes. During the 5 minute period during which the video is streaming, the Vuser will not be able to perform any other actions, such as clicking the **Pause** button or skipping to a different location in the video. Clicking a button while a video is streaming is an example of asynchronous behavior.

Although a single generated step makes script replay more reliable, it is not able to replay asynchronous actions (such as pause and seek) that you may have performed while recording the script. The single generated step also does not replay the automatic requests that the client performs when Dynamic Stream is in use. If it is important to replay these asynchronous actions, you must record the Vuser script using the regular recording mode - not the simplified recording mode - and then manually modify the generated script as described below.

## Modifying scripts to replay asynchronous user actions

If your Vuser script must be able to replay asynchronous actions that are performed while a streaming action is executed, you must record the Vuser script using the regular recording mode - not the simplified recording mode - and then manually modify the generated script. The modified script will include a combination of single streaming steps and the more verbose steps that are generated with regular recording.

> **Note:** In this section, we will use the term **required user actions** to refer to the actions that must be performed while a video is streamed.

To create a script that can replay asynchronous behavior, first you record the script using the regular recording mode - not the simplified recording mode. Thereafter, identify the **flex_rtmp_send** steps that represent required user actions. Then replace the steps between the required user actions with single streaming functions. See the sections below for details.

> **Note:** The modification procedure differs slightly between RTMP and RTMPT steps.

## Modifying recorded Flex RTMP steps

When you use the regular recording mode, VuGen generates **flex_rtmp_receive** and **flex_rtmp_send** steps for all communication with the server. This ensures that user actions such as pause and seek, as well as automatic requests that the client performs when Dynamic Stream is in use, are included in the script. However, this method also captures less-necessary lines of code that are difficult to read and may not be reliable during replay of streaming actions.

> **Note:** To activate the regular recording mode, clear the **Generate single step for RTMP/T stream handling** option in the **Flex > RTMP** pane of the Recording Options dialog box.

Follow the instructions below to remove the unnecessary **flex_rtmp_receive** and **flex_rtmp_send** steps from your script.

1. Search for the **flex_rtmp_send** step that contains the initial play argument. For example:

```
flex_rtmp_send("send_step2",
"ConnectionID=10",
"Snapshot=tRTMP6.inf",
MESSAGE,
...
MESSAGE,
...
"Argument=<arguments><string>play</string><number>0</number><null/>"
...
LAST);
```

2. Delete or comment out the **flex_rtmp_receive** steps that occur during streaming. For example:

```
//This is the start of the stream:
flex_rtmp_receive("recv_step2",
        "ConnectionID=10",
        "Snapshot=tRTMP7.inf",
        CHANNEL,
        "ChunkStreamID=2",
        CHANNEL,
        "ChunkStreamID=2",
        CHANNEL,
        "ChunkStreamID=4",
        CHANNEL,
        "ChunkStreamID=2",
        LAST);
flex_rtmp_receive("recv_step3",
        "ConnectionID=10",
        "Snapshot=tRTMP8.inf",
        CHANNEL,
        "ChunkStreamID=5",
        CHANNEL,
        ...
```

3. Remove the **flex_rtmp_send** steps that are not related to the required user actions, such as "user control message" types. For example:

```
flex_rtmp_send("send_step3",
        "ConnectionID=10",
        "Snapshot=tRTMP9.inf",
        MESSAGE,
        "DataType=user control message",
        "EventType=set buffer length",
        "MessageStreamID=1",
        "BufferLength=100",
```

```
        LAST);
```

4. When you find a **flex_rtmp_send** step that represents a required user action, do the following:
   a. Manually add a **flex_rtmp_receive_stream** step before the send step.

      - Make sure that the **ConnectionID** argument has the same value as the steps you removed above it.

      - The **Snapshot** argument is not relevant for the manually added step.

      - You can use the **ContinueToNexStepAfter = <msec>** argument to control the minimum play duration of the stream to download before continuing to the next step.

   b. Determine the **flex_rtmp_send** steps that represent the required user actions. These will likely include arguments such as **pauseRaw**, **pause**, **seek** and **play2** (for Dynamic Stream). For example:

```
flex_rtmp_send("send_step5",
"ConnectionID=10",
"Snapshot=tRTMP62.inf",
MESSAGE,
"DataType=command message amf3",
"ChunkStreamID=8",
"MessageStreamID=1",
"Argument=<arguments><string>pauseRaw</string><number>0</number><null/>"
"<boolean>true</boolean><number>12000</number></arguments>",
LAST);
```

   c. Determine whether there are some extra **flex_rtmp_send** steps that you can remove. For example, if you dragged a button to seek in the stream, subtle jerks in the motion may be recorded as separate pause and seek actions. In these cases, may not need all of them. Keep only those that describe the desired operations.

   d. Identify the **flex_rtmp_receive** step that indicates that the server has received the end of the user action. For example:

```
//This is the confirmation from the server on the "seek" command.
flex_rtmp_receive("recv_step55",
"ConnectionID=10",
"Snapshot=tRTMP68.inf",
CHANNEL,
```

```
"ChunkStreamID=2",
CHANNEL,
"ChunkStreamID=2",
LAST);
```

5. Repeat steps 2 - 4 for each set of unnecessary receive data and required user actions in your script.

For additional details on **flex_rtmp_receive_stream** including a complete example, see the Function Reference (**Help > Function Reference**).

### Modifying recorded Flex RTMPT steps

When you use the regular recording mode, VuGen generates a **flex_rtmp_tunneled_send** step for all communication with the server. This ensures that user actions such as pause and seek, as well as automatic requests that the client performs when Dynamic Stream is in use, are included in the script. However, this method also captures less-necessary lines of code that are difficult to read and may not be reliable during replay of streaming actions.

> **Note:** To activate the regular recording mode, clear the **Generate single step for RTMP/T stream handling** check box in the **Flex > RTMP** pane of the Recording Options dialog box.

Follow the instructions below to remove the unnecessary steps from your script.

1. Search for the **flex_rtmp_tunneled_send** step that contains the initial play argument. For example:

```
flex_rtmp_tunneled_send("send_step2",
"SessionID=1",
"Snapshot=t36.inf",
MESSAGE,
...
MESSAGE,
...
"Argument=<arguments><string>play</string><number>0</number><null/>"
...
LAST);
```

2. Remove **flex_rtmp_tunneled_send** steps that are not related to required user actions, such as "user control message" types. For example:

```
flex_rtmp_tunneled_send("send_step3",
```

```
    "SessionID=10",
    "Snapshot=t15.inf",
    MESSAGE,
    "DataType=user control message",
    "EventType=set buffer length",
    "MessageStreamID=1",
    "BufferLength=100",
    LAST);
```

3. When you find a **flex_rtmp_tunneled_send** step that represents a required user action, do the following:
   a. Add a **ContinueToNexStepAfter = <msec>** argument to the previous step. The **ContinueToNexStepAfter = <msec>** argument controls the minimum play duration of the stream to download before continuing to the next step. For example:

```
flex_rtmp_tunneled_send("send_step2",
"SessionID=1",
"Snapshot=t36.inf",

//Read the stream until at least 15 seconds of media have been downloaded
"ContinueToNexStepAfter = 15000",
MESSAGE,
...
MESSAGE,
...
"Argument=<arguments><string>play</string><number>0</number><null/>"
...
LAST);
```

   b. Determine the **flex_rtmp_tunneled_send** steps that represent the required user actions. These will typically include arguments such as **pauseRaw**, **pause**, **seek** and **play2** (for Dynamic Stream). For example:

```
flex_rtmp_tunneled_send("send_step5",
"SessionID=10",
```

```
"Snapshot=t16.inf",
MESSAGE,
"DataType=command message amf3",
"ChunkStreamID=8",
"MessageStreamID=1",
"Argument=<arguments><string>pauseRaw</string><number>0</number><null/>"
"<boolean>true</boolean><number>12000</number></arguments>",
LAST);
```

c. Determine whether there are extra **flex_rtmp_tunneled_send** steps that you can remove. For example, if you dragged a button to seek in the stream, subtle jerks in the motion may be recorded as separate pause and seek actions. In these cases, you may not need all of them. Keep only those that describe the desired operations.

4. Repeat steps 2 - 3 for each set of unnecessary send data and required user actions in your script.

For additional details on the **flex_rtmp_tunneled_send** function, including a complete example, see the Function Reference (**Help > Function Reference**).

### Live Streaming

VuGen's Flex protocol supports Adobe's Live Streaming. If VuGen detects a live stream while you record a Vuser script, VuGen adds '**ContinueToNextStepAfter**' and '**ContinueMode**' arguments to the generated **flex_rtmp_receive_stream** or **flex_rtmp_tunneled_send** function. These additional arguments enable the live stream to be accurately replayed. For details on these arguments, see the Function Reference (**Help > Function Reference**).

> **Note:** The default value of the generated **ContinueToNextStepAfter** argument is the length of time (in milliseconds) for which the video was streamed while the Vuser script was recorded.

#### RTMP Tunneled

VuGen supports the recording of RTMP Tunneled steps in Flex application which are split into the following step types:

- **Messaging support**. The Flex protocol supports enhanced record and replay of messaging and has been verified for Adobe LiveCycle Data Services ES2 Version 3.1.
- **Streaming support**. The Flex protocol supports enhanced record and replay of streaming. For details, see "RTMP/RTMPT Streaming" on page 531.

  When you record a Flex stream, by default, VuGen generates a single **flex_rtmp_tunneled_send** step in place of many **flex_rtmp_tunneled_send** steps. This step makes your script more readable (eliminating tens or hundreds of lines) and makes the replay more reliable.

> **Note:** The new **flex_rtmp_tunneled_send** step is generated when the **Generate single RTMP/T step** option is selected in the **Flex:RTMP** pane of the **Recording Options** dialog box. Although this step makes the script more reliable, it does not replay certain actions you may perform while recording your script, such as pause and seek. It also does not replay the automatic requests that the client performs when Dynamic Stream is in use.

If it is important to replay these actions, you can clear the **Generate single RTMP/T step** option in the **Flex> RTMP** pane of the **Recording Options** dialog box, which causes LoadRunner to generate the steps for all of the raw streaming data.

However, to ensure proper replay, you must manually modify the generated script as described in "RTMP/RTMPT Streaming" on page 531.

The above functionality has been verified for Flash Media Server versions 3.5 and 4.

- **Externalizable objects**. VuGen supports externalizable objects over RTMP Tunneled. For details, see "Externalizable Objects in Flex Scripts" on page 542.
- **User Data Points**. VuGen generates a number of new data points that provide more useful information for analysis.
- The Flex RTMP-Tunneled protocol supports manual correlation using **web_reg_save_param_xpath API**.

For additional details on **flex_rtmp_tunneled_send** including a complete example, see the Function Reference (**Help > Function Reference**).

## Record a Flex Script

> **Note:** You can generate a Flex Vuser script by analyzing an existing network traffic file (capture file). This method may be useful for creating Vuser scripts that emulate activity on mobile applications. For details, see "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723.

1. Create a new script or open an existing script.

   Select **New Script and Solution > Flex protocol**. For details, see "Creating or Opening Vuser Scripts" on page 136.

2. Configure the recording options. The recording options affect the way that a Vuser script is generated after recording or regenerating the script.

   - **Recording options > Flex > RTMP > Generate single step for RTMPT/Streaming**

     This option, selected by default, enables VuGen to create a single step while recording a stream. However, when you create a single step, certain actions are not replayed, such as pause and seek. If you want to be able to replay these actions, disable the option.

   - **Recording Options > Flex > Configuration > Use External JVM**

     If you are using an external Java Virtual Machine select this option and configure the path of the JVM in the value field.

- **Recording Options > Flex > Configuration > Use GraniteDS**

  Select this option if you are using GraniteDS as a sever side Data Service configuration.

- **Recording Options > Flex > Externalizable Objects**

  This option enables you to specify additional .jars that are required to record your script. For details, see "Externalizable Objects in Flex Scripts" on page 542.

- **Recording Options > HTTP Properties > Advanced**

  Make sure that the **Save snapshot resources locally** option is enabled, as local snapshots are required for data loading. For details, see "HTTP Properties > Advanced Recording Options" on page 195. For concept and user interface details, see "Recording Options" on page 164.

3. Initialize the recording session.

   When creating a new script, this occurs automatically. To manually start recording, click **Start Record** on the VuGen toolbar, complete the Start Recording dialog box, and then click **Start Recording**. VuGen's floating toolbar appears, VuGen opens your application and begins recording your actions. For user interface details, see "Start Recording Dialog Box" on page 240. For details on the script sections into which you can record, see "Vuser Script Sections" on page 155.

4. Perform business processes on your application.

   Perform the desired business processes that you wish to record. The floating toolbar allows you to insert transactions, rendezvous points, and comments. You can also use the floating toolbar to specify into which section of the script to record. For user interface details, see "Floating Recording Toolbar" on page 244.

   Click **Stop** on the floating toolbar when you are finished recording.

5. Regenerate the script to determine if all the steps were parsed correctly. For details, see "Code Generation in the Flex Protocol" on the next page.

## Setting the Flex Recording Mode

You can instruct VuGen how to generate a script from a Flash Remoting session using the Flex and Web Protocols.

### Example

Use Web HTTP technology to generate **web_custom_request** functions with the Flash Remoting information.

**Example:**

```
web_url("flash",
    "URL=http://<HOST>:<PORT>/flash/",
    "Resource=0",
    "RecContentType=text/html",
    "Referer=",
    "Snapshot=t1.inf",
    "Mode=HTML",
    EXTRARES,
```

```
    "Url=movies/XMLExample.swf", "Referer=", ENDITEM,
    "Url=movies/JavaBeanExample.swf", "Referer=", ENDITEM,
    LAST);
web_link("Sample JavaBean Movie Source",
    "Text=Sample JavaBean Movie Source",
    "Snapshot=t2.inf",
    EXTRARES,
    "Url=XMLExample.swf", "Referer=", ENDITEM,
    "Url=JavaBeanExample.swf", "Referer=", ENDITEM,
    LAST);
web_custom_request("gateway",
    "URL=http://<HOST>:<PORT>/flashservices/gateway",
    "Method=POST",
    "Resource=0",
    "RecContentType=application/x-amf",
    "Referer=",
    "Snapshot=t3.inf",
    "Mode=HTML",
    "EncType=application/x-amf",
    "BodyBinary=\\x00\\x00\\x00\\x01\\x00\\x10amf_server_debug\\x01
                \\x00\\x00\\x00`\\x03\\x00\ncoldfusion\\x01\\x01\\x00
                \namfheaders\\x01\\x00\\x00\\x03amf\\x01\\x00\\x00
                \\x0Bhttpheaders\\x01\\x00\\x00\trecordset\\x01\\x01
                \\x00\\x05error\\x01\\x01\\x00\\x05trace\\x01\\x01
                \\x00\\x07m_debug\\x01\\x01\\x00\\x00\t\\x00\\x01
                \\x00/flashgateway.samples.FlashJavaBean.testDocument
                \\x00\\x02/1\\x00\\x00\\x004\n\\x00\\x00\\x00\\x01
                \\x0F\\x00\\x00\\x00*
                <TEST message=\"test\"><INSIDETEST /></TEST>",
        LAST);
```

# Code Generation in the Flex Protocol

## Code Generation Notification

If a Flex or Java over HTTP script encounters an error during the code generation phase, VuGen issues a warning. This warning appears in the Errors pane, when the **Warnings** button is selected, and the **Define Available Categories** filter is set to **All** or **Code Generation Notification**.The list of warnings displays details about each error, as well as recommended actions for resolving the problem. Follow the recommended actions and regenerate the script.

If the error is related to externalizable objects in a Flex script, see "Externalizable Objects in Flex Scripts" on the next page.

To manually open the Errors pane at any time, select **View > Errors**.

## Parsing Responses in Flex Scripts

When generating a Flex script, VuGen attempts to parse responses for any of the following steps:

- flex_amf_call
- flex_remoting_call
- flex_login
- flex_logout
- wflex_ping

If the parsing fails, the following attribute is dynamically added to the step:

IsParseResponse = No

This instructs VuGen not to parse the responses for that step during script replay. Every time you regenerate the script, VuGen will attempt to parse again, and will set this parameter to false if it fails. If needed, you can delete this line, or set the value to = 'Yes' to force VuGen to parse responses for that step during replay.

Additionally, you can manually add the attribute and set the value to 'No' in a generated script, even if the parse is successful, as it may enhance replay performance.

## Externalizable Objects in Flex Scripts

When recording a Flex application, information is usually passed between the client and server using known serialization methods (AMF). If this is the case, VuGen creates a **flex_amf_call** and both the request and response are parsed.

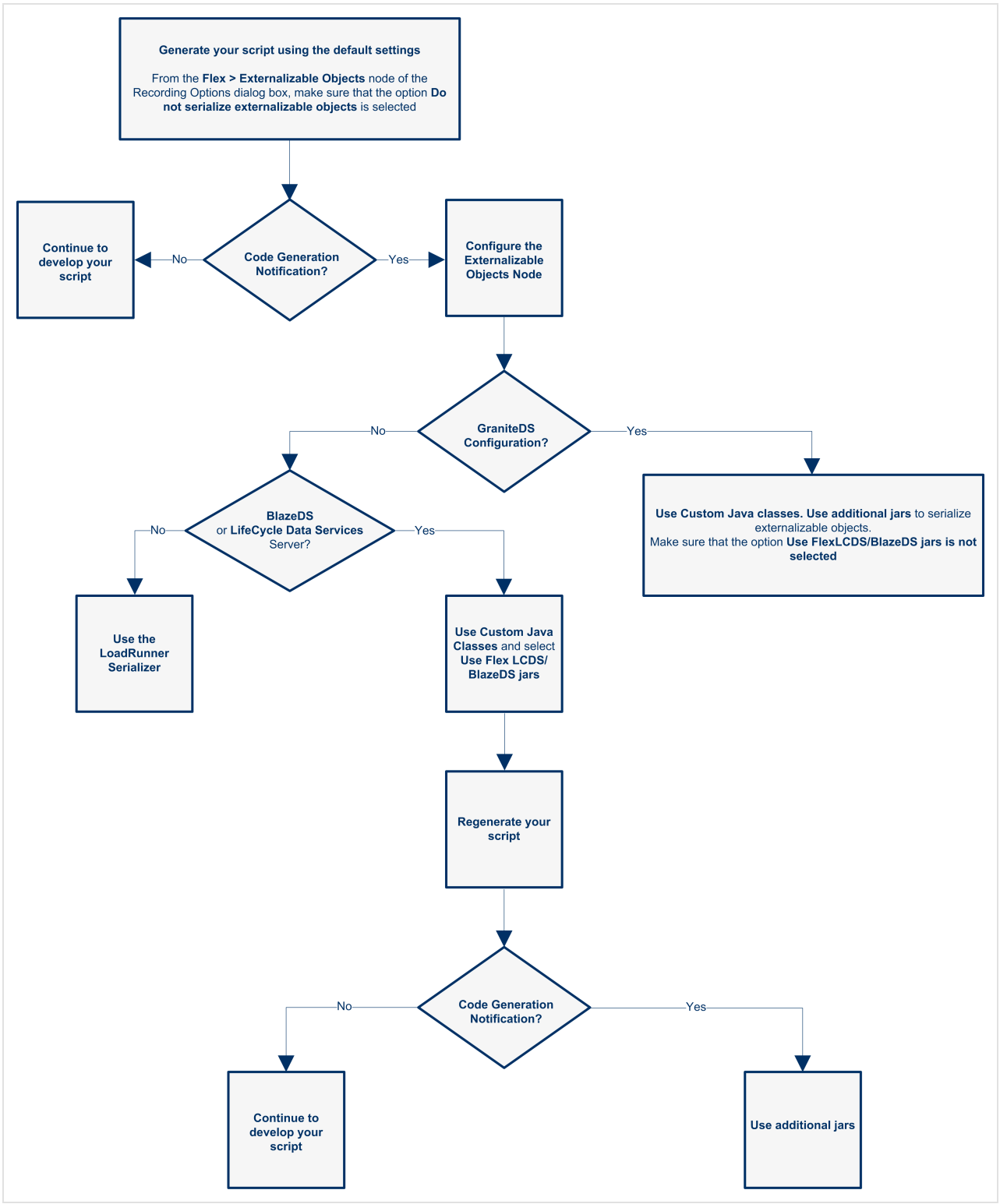
However, when a given AMF object uses a custom serialization method (externalizable), VuGen automatically issues a warning. This warning displays details about the exception, as well as recommended actions for resolving it.

The following are some examples of the exceptions that the generated script may include when an AMF object uses a custom serialization method:

- **Request and response not parsed.** This exception is automatically displayed in the Errors pane when the **Warnings** button is selected, and the **Define Available Categories** filter is set to **All** or **Code Generation Notification**. Details about the exception are listed, as well as recommended actions. For details, see "Errors Pane" on page 95.
- **Request parsed but response is not parsed.** VuGen generates a `IsParseResponse=No` statement. Additionally, VuGen issues a warning that is automatically displayed in the Errors pane when the **Warnings** button is selected, and the **Define Available Categories** filter is set to **All** or **Code Generation Notification**. The list of warnings displays details about the exception, as well as recommended actions. For details, see "Errors Pane" on page 95.

For details on configuring the **Recording Options > Flex > Externalizable Objects** Node, see "Flex > Externalizable Objects Recording Options" on page 183.

The following flowchart illustrates the steps to resolve externalizable objects in Flex scripts:


Generate your script using the default settings
From the Flex > Externalizable Objects node of the Recording Options dialog box, make sure that the option Do not serialize externalizable objects is selected
Code Generation Notification?
No
Continue to develop your script
Yes
Configure the Externalizable Objects Node
GraniteDS Configuration?
No
Yes
Use Custom Java classes. Use additional jars to serialize externalizable objects.
Make sure that the option Use FlexLCDS/BlazeDS jars is not selected
BlazeDS or LifeCycle Data Services Server?
No
Use the LoadRunner Serializer
Yes
Use Custom Java Classes and select Use Flex LCDS/ BlazeDS jars
Regenerate your script
Code Generation Notification?
No
Continue to develop your script
Yes
Use additional jars

### See also:

- "Flex > Externalizable Objects Recording Options" on page 183
- "Serialize Flex Scripts" on the next page

## Flex Correlations

VuGen supports correlation in Flex scripts.

Support for correlations applies to the following Flex steps:

- flex_login
- flex_logout
- flex_ping
- flex_amf_call
- flex_remoting_call
- flex_rtmp_tunnelled_connect
- flex_rtmp_tunnelled_send

Flex correlation includes integration with the following features:

- **Correlations rules**

  **DSid**, **jsessionid**, and **RTMPT ID**
- **Design Studio**
- **Manual correlation** using the API **web_reg_save_param_xpath**.

  For general information, see "Correlating Values" on page 254.

  For task details, see "Correlate Scripts Using Design Studio" on page 268.

## Flex Snapshots

Vuser scripts based on the Flex protocol utilize VuGen's Snapshot pane.

- For details on how to work with the Snapshot pane, see "Work with Snapshots" on page 297.
- For details on the Snapshot pane UI, see "Snapshot Pane" on page 85.

In addition, a new snapshot has been designed to show Flex data in several views:

- Raw Data

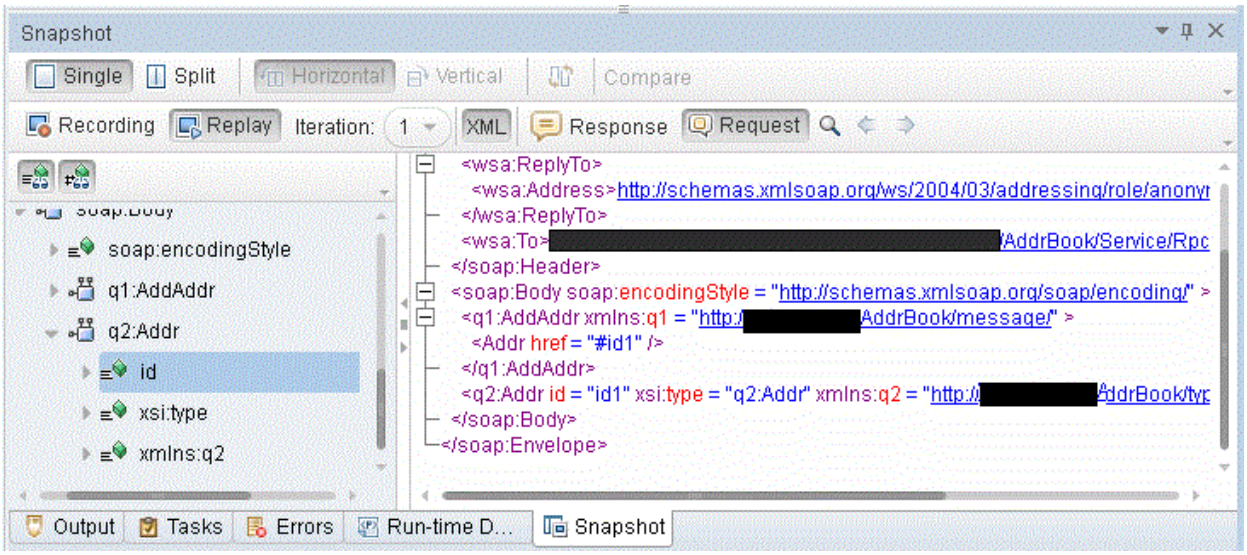
  The data received from the server that has not been formatted or parsed in any way.
- Response Body

  Data entity received from the server
- Request Body

  Data entity sent to the server
- Headers

If the Response Body and the Request Body are in XML format, the data can be displayed as:

- Text
- Hex
- XML




# Serialize Flex Scripts

## External Java Serializer

You can use the Java classes from the Flex server to serialize AMF messages in your script. This process has been simplified so that you need to include the application JAR files only if the AMF objects implement an externalizable interface.

1. In the **Recording Options > Flex > Externalizable Objects** node, select **Serialize objects using** and select **Custom Java Classes** from the drop-down menu.
2. Add the relevant files by using the **Add all classes in folder** or **Add JAR or Zip file** buttons. Add the following files:
   a. **For Adobe BlazeDS or Adobe LCDS**, add the following JAR files:
      - flex-messaging-common.jar
      - flex-messaging-core.jar
   b. Regenerate the script and note any errors. Open the recording options dialog box using the **Generation Options** button and add the necessary application JAR files.
3. Ensure that the added files exist in the same location both on the VuGen machine and on all load generators.

   For details, see "Externalizable Objects in Flex Scripts" on page 542.

**Notes and Limitations for the Java Serializer:**

- Supported JDK versions: 1.6 and earlier.
- Supported servers: Adobe BlazeDS and Adobe Livecycle DS.
- Microsoft .NET classes are not supported.
- During code generation VuGen performs a validity test of the request buffers by verifying that the buffer can be read and written using the provided jars. Failure in this validity test indicates that the classes are incompatible with VuGen.

### Built-in Serializer

You can attempt to serialize externalizable objects using the built-in serializer. Ensure that you have saved all open scripts because this option may result in unexpected errors or invalid steps.

1. Save all open scripts in VuGen.
2. In the **Recording Options > Flex > Externalizable Objects** node, select **Serialize objects using** and select **LoadRunner AMF serializer**.

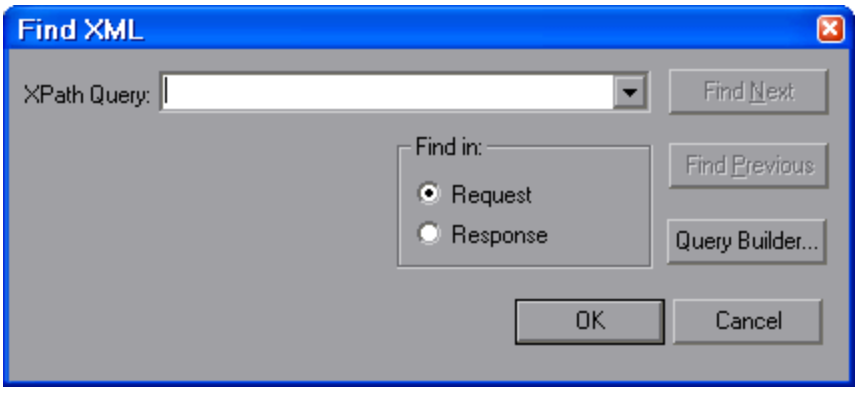
## Query an XML Tree

VuGen provides a Query Builder that lets you create and execute queries on the XML.

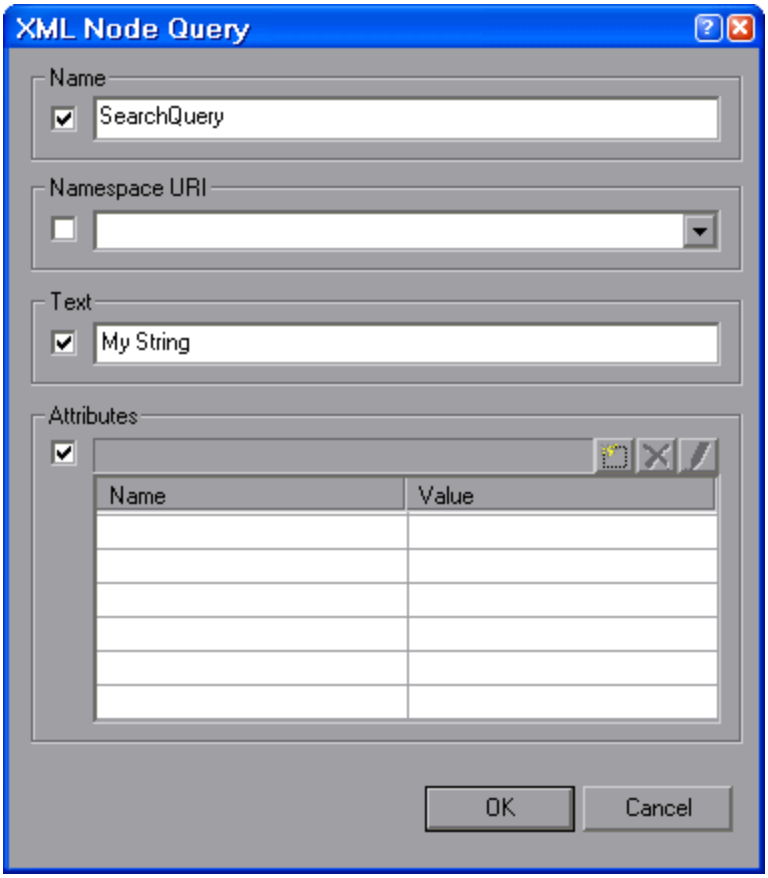
VuGen displays the XML code in an expandable tree. You can perform a query on your XML document, and search for a specific Namespace URI, value, or attribute. Note that all queries are case-sensitive.

**Perform a query**

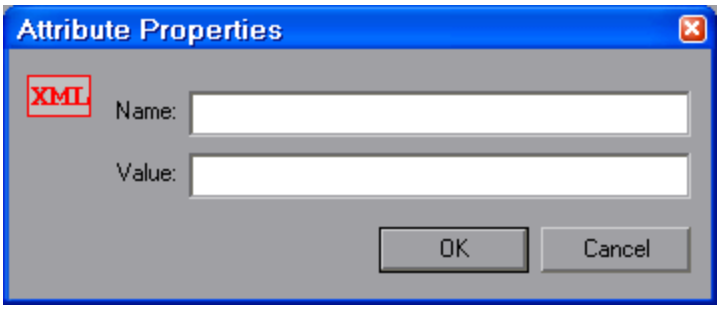
1. In the Snapshot pane, select the node that you want to search. Click the **Find XML** button. The Find XML dialog button opens.



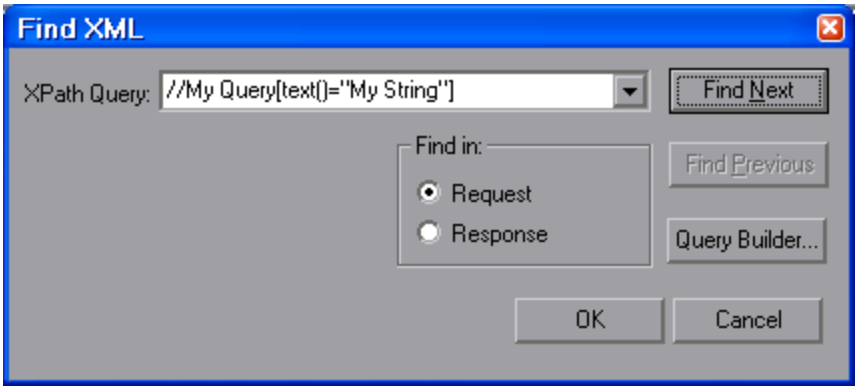

2. Select **Request** or **Response**. Enter an XPath query and click **OK**. To formulate a query, click **Query Builder** button. The XML Node Query dialog box opens.
3. Enable one or more items for searching.




4. Enable the **Name** section to search for the name of a node or element.
5. Enable the **Namespace URI** section to search for a namespace.
6. Enable the **Text** section to search for the value of the element indicated in the Name box.
7. Enable the **Attributes** section to search for an attribute.
8. Enter the search text in the appropriate boxes. To add an attribute, click the **Add** button. The Attribute Properties box opens. Enter an attribute name and value. Click **OK**.




9. Click **OK** in the XML Node Query dialog box. VuGen places the text of the query in the Find XML box.




10. Click **Find Next** to begin the search.

## Troubleshooting and Limitations for Flex

This section describes troubleshooting and limitations for the Flex Protocol.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

- A Flex script cannot be generated using an external Java Virtual Machine (JVM) version 1.4 or lower.
- The Flex protocol does not support the **Generate snapshot on error** option (**Replay > Runtime Settings > General > Miscellaneous > Error Handling**).
- Proxy recording is not supported for the Flex protocol.
- If your script contains more than one **flex_RTMP_tunneled_connect** step, with the same gateway parameter, you must insert a disconnect step for the previous **flex_RTMP_tunneled_connect** step before you connect again. For example:

```
("connect_step0",
"SessionId=0",
"Gateway=http://123.123.123.123:1935",
…
LAST);
…
Flex_rtmp_tunneled_disconnect("disconnect_step0",
"SessionId=0")
flex_rtmp_tunneled_connect("connect_step1",
"SessionId=1",
```

```
"Gateway=http://123.123.123.123:1935",
…
LAST);
…
>Flex_rtmp_tunneled_disconnect("disconnect_step0",
"SessionId=1")
```

- If a subsequent **flex_rtmp_tunneled_connect** command has the same gateway parameter as the previous **flex_trtmp_tunneled_connect** step and the **flex_rtmp_tunneled_disconnect** step is omitted, the script will pause indefinitely.

### GraniteDS (Data Services)

- If you have modified the granite-config.xml, copy it to the **<Installation_folder>\dat** directory.
- When switching between BlazeDS and GraniteDS parsing (**Recording Options > Flex > Configuration**), VuGen must be restarted.
- LoadRunner cannot serialize both GraniteDS and BlazeDS/LCDS messages in the same script.
- All limitations that apply to AMF3 parsing, also apply to externalizable objects over RTMP.
- If the **Generate flex_rtmp_receive_stream step** option is enabled, all transactions, comments, and rendezvous points that you add from the Recording toolbar are added to the script after the **flex_rtmp_receive_stream** step in your script.
- Web diagnostics is not supported for RTMP and RTMPT steps (even when the breakdown is enabled).
- You cannot replay two RTMPT steps at once.

# Java Protocols

VuGen supports the following protocols for testing Java applications:

- "Java Record Replay Protocol" on page 567
- "Java Vuser (Manual) Protocol" on page 583
- "Java over HTTP Protocol" on page 586

The following topics describe general information about working with all Java protocols.

# Set Up the Java Environment

In order to successfully work with any Java protocol, you must install a supported JDK on the VuGen machine.

## Setting up the Java environment

Install JDK/JRE and configure runtime settings for the Java protocol, as described in the following steps.

### To set up the Java environment:

1. Install JDK or JRE depending on your needs.

| | |
|---|---|
| JDK | Replay<br>Compile |
| JRE | Code generation<br>Record |

For information about supported versions, see"JDK support matrix" on the next page.

> **Note:**
> - JRE alone is insufficient for running a script, since running a script involves compilation and replay.
> - To compile and replay the Java Vuser scripts, you must have a complete JDK installation.
> - It is preferable not to have more than one JDK or JRE installation on a machine. If possible, uninstall all unnecessary versions.

2. In your Java protocol script configure the runtime settings:
   a. In **Java Environment > Java VM**, specify the JDK:

      You can instruct VuGen to **use internal logic to locate the JDK** (default) to locate the SDK, or you can specify which SDK to use.

      Using internal logic (Java 8 and later), when more than one JDK is installed on the machine, VuGen attempts to locate the JDK in the environment variables and selects the first one defined there.

      The first of the following to succeed is used:

      - Uses the LR_JAVA_HOME environment variable. You create this variable if you want to use it to select the JDK.
      - Uses the same JDK used for recording.
      - Searches for a java.exe using the default Windows application search algorithm.

   b. **Java Record Replay** and **Java Vuser** scripts: To replay with a 64-bit JDK, select

**Miscellaneous > Replay script with 64-bit**.

c. All extra files such as jars, classes, and so on, used by the Java Vuser must be accessible by the script by doing one of the following:

- Add the files to the script's **Extra Files** folder: In the Solution Explorer pane, right click the **Extra Files** node and select **Add Files to Script**. The files are copied to the script's folder.
- In the script's runtime settings, in **Java Environment > Classpath**, set the classpath to the Java classes. These files are then used as a reference, but are not contained in the script's folder.

When the environment is set up, you can enhance or modify your Java script with standard Java code using JDK libraries or custom classes. VuGen utilizes the standard Java compiler, javac.exe, to compile the script. Once the script is successfully compiled you can incorporate it into a LoadRunner scenario or Business Process Monitor configuration.

### JDK support matrix

The following JDK versions are required for script replay with Java protocols.

| | Version 6 | Version 7 | Version 8 | Version 9 | Version 10 |
|---|---|---|---|---|---|
| **32-bit** (for Java over HTTP, Java Record Replay & Java Vuser protocols) | Oracle JDK | Oracle JDK | Oracle JDK | Common OpenJDK Distributions | Common OpenJDK Distributions |
| **64-bit** (for Java Record Replay & Java Vuser protocols) | n/a | n/a | Oracle JDK | Oracle JDK/Common OpenJDK Distributions | Oracle JDK/Common OpenJDK Distributions |

For more information about supported versions, see System Requirements.

## Java Protocol Programming Tips

When programming a Java Vuser script, you can paste ready-made code segments into scripts or import ready-made classes in order to invoke their methods. If Vusers need to run as threads under the Controller (for scalability reasons), you need to make sure that all of the imported code is thread-safe.

Thread-safety is often difficult to detect. A Java Vuser may run flawlessly under VuGen and under the Controller with a limited number of Vusers. However, problems may then occur with a large number of Vusers. Code that is not thread-safe is usually the result of static class member usage as shown in the following example:

```
import lrapi.*;
public class Actions
```

```
{
   private static int iteration_counter = 0;
    public int init() {
        return 0;
    }
    public int action() {
        iteration_counter++;
        return 0;
    }
    public int end() {
        lr.message("Number of Vuser iterations: "+iteration_counter);
        return 0;
    }
}
```

When you run one Vuser, the **iteration_counter** member determines the number of iterations that were executed. When multiple Vusers run together as threads on a single virtual machine, the static class member **iteration_counter** is shared by all threads, resulting in an incorrect counting. The total number of all Vusers iterations is counted.

If code is known to be non thread-safe and you still want to import it into your script, you can run the Vusers as processes. For more information on running Vusers as threads or processes, see "Runtime Settings" on page 301.

When you run a basic Java Vuser script, it usually consists of a single thread—the main thread. Only the main thread can access the Java Vuser API. If a Java Vuser spawns secondary worker threads, using the Java API may cause unpredictable results. Therefore, we recommend that you use the Java Vuser API only in the main thread. Note that this limitation also affects the **lr.enable_redirection** function.

The following example illustrates where the LR API may and may not be used. The first log message in the execution log indicates that the value of flag is false. The virtual machine then spawns a new thread set_thread. This thread runs and sets flag to true, but will not issue a message to the log, even though the call to lr.message exists. The final log message indicates that the code inside the thread was executed and that flag was set to true.

```
boolean flag = false;
public int action() {
   lr.message("Flag value: "+flag);
   Thread set_thread = new Thread(new Runnable();{
        public void run() {
           lr.message("LR-API NOT working!");
           try {Thread.sleep(1000);} catch(Exception e) {}
           flag = true;
        }
   });
   set_thread.start();
   try {Thread.sleep(3000);} catch(Exception e) {}
   lr.message("Flag value: "+flag);
```

```
    return 0;
}
```

**Note:**

Although .NET-based and Java protocols support creating threads, we recommend that you do not use background threads in real load testing scenarios because:

- Threads can degrade tests scalability.
- Threads can affect performance measurements.
- The utility functions' behavior is undetermined if called from any thread except the Vuser main thread which runs the vuser_init, Action, and vuser_end actions. This applies to all functions named lr*.

## Manually Insert Java Methods

You can use pre-defined Java packages within a Vuser script. The packages can be included in specific archive files or in the archive files contained in designated folders.

You can customize the function generation settings by modifying the recording options. For more information, see "General > Script Recording Options" on page 188.

**To insert Java functions:**

1. Click within your script at the desired point of insertion.
2. Click **Java Function** in the toolbar. The Insert Java Function dialog box opens. Here you can add functions to your Vuser script.

   Packages, classes, methods, and other objects are represented by icons as follows:

| Icon | Item | Example |
|---|---|---|
| | Package | java.util |
| | Class | public class Hashtable extends java.util.Dictionary implements java.lang.Cloneable, java.io.Serializable |
| | Interface Class | public interface Enumeration |
| | Method | public synchronized java.util.Enumeration keys () |
| | Static Method | public static synchronized java.util.TimeZone getTimeZone |

| Icon | Item | Example |
|---|---|---|
| | Constructor Method | public void Hashtable () |

3. Under **Locations**, VuGen lists the paths and archives defined in the CLASSPATH environment variable

   To add a path, click **Select Folder** and then locate and highlight the required folder.

   To add an archive (**jar** or **zip**), click **Select Library** and then locate and highlight one or more archive files.

   Repeat for each path or archive you want to add.
4. Select or clear the check boxes to the left of each item in the **Locations** list. If an item is selected, its members will be listed in the **Packages** list below.
5. Click **Packages** to close the Locations area and view the available packages.

> **Tip:** Click the arrows to the left of each item in the navigator, to expand or collapse the trees.

6. Select an object and click **Insert**. VuGen inserts the object at the location of the cursor in the script. To paste all the methods of a class into your script, select the class and click **Insert**.

   Repeat for all of the desired methods or classes, and then click **Close**.

> **Note:** When you close the Insert Java Function dialog box, VuGen gives you the option to add the newly added locations to the Classpath list in the Java Environment runtime settings.

7. Modify the method parameter.

   If the script generation setting **DefaultValues** is set to **true**, you can use the default values inserted by VuGen.

   If **DefaultValues** is set to **false**, you must add parameters for all methods you insert into the script.

   In addition, modify any return values. For example, if your script generated the following statement:

```
"(String)=LavaVersion.getVersionId();"
```

   replace (String) with a string type variable.
8. Add any necessary statements to your script, such as imports or Java-specific functions, from the Function Reference (**Help > Function Reference**). For details, see "Java Vuser (Manual) Protocol" on page 583.
9. Save the script and run it from VuGen.

# Enhance a Java Script

This topic describes how to enhance custom Java scripts.

## Insert transactions

You define transactions to measure the performance of the server. Each transaction measures the time it takes for the server to respond to specified requests. These requests can be short or complex tasks.

When working with LoadRunner, you can analyze the performance per transaction during and after the scenario run, using online monitor and graphs.

You can also specify a transaction status: lr.PASS or lr.FAIL. You can let the Vuser automatically determine if the transaction was successful, or you can incorporate it into a conditional loop. For example, in your code you can check for a specific return code. If the code is correct, you issue a lr.PASS status. If the code is wrong, you issue an lr.FAIL status.

## Mark a transaction

1. Insert **lr.start_transaction** into the script, at the point where you want to begin measuring the timing of a task.
2. Insert **lr.end_transaction** into the script, at the point where you want to stop measuring the task. Use the transaction name as it appears in the **lr.start_transaction** function.
3. Specify the desired status for the transaction: lr.PASS or lr.FAIL.

```
public int action() {
for(int i=0;i<10;i++)
   {
    lr.message("action()"+i);
  lr.start_transaction("trans1");
    lr.think_time(2);
  lr.end_transaction("trans1",lr.PASS);
   }
    return 0;
}
```

## Insert rendezvous points

To emulate heavy user load on your client/server system, you synchronize Vusers to perform a task at exactly the same moment by creating a rendezvous point. When a Vuser arrives at the rendezvous point, it is held by the Controller until all Vusers participating in the rendezvous arrive.

You designate the meeting place by inserting a rendezvous function (lr.rendezvous) into your Vuser script at the point where you want the Vusers to perform a rendezvous.

```
public int action() {
```

```
  for(int i=0;i<10;i++)
   {
   lr.rendezvous("rendz1");
    lr.message("action()"+i);
    lr.think_time(2);
   }
   return 0;
}
```

## Obtain Vuser information

You can add the following functions to your Vuser scripts to retrieve Vuser information:

| | |
|---|---|
| **lr.get_attrib_string** | Returns a string containing command line argument values or runtime information such as the Vuser ID or the load generator name. |
| **lr.get_group_name** | Returns the name of the Vuser's group. |
| **lr.get_host_name** | Returns the name of the load generator executing the Vuser script. |
| **lr.get_master_host_name** | Returns the name of the machine running the LoadRunner Controller or Business Process Monitor. |
| **lr.get_scenario_id** | Returns the ID of the current scenario. (LoadRunner only) |
| **lr.get_vuser_id** | Returns the ID of the current Vuser. (LoadRunner only) |

In the following example, the lr.get_host_name function retrieves the name of the computer on which the Vuser is running.

```
String my_host = lr.get_host_name();
```

For more information about the above functions, see the Function Reference (**Help > Function Reference**).

## Issue output messages

When you run a scenario, the Controller Output window displays information about script execution. You can include statements in a Vuser script to send error and notification messages to the Controller. The Controller displays these messages in the Output window. For example, you could insert a message that displays the current state of the client application. You can also save these messages to a file.

**Note:** Do not send messages from within a transaction. Doing so lengthens the transaction

execution time and may skew the actual transaction results.

You can use the following message functions in your Vuser script:

| | |
|---|---|
| **lr.debug_message** | Sends a debug message to the Output window. |
| **lr.log_message** | Sends a message to the Vuser log file. |
| **lr.message** | Sends a message to a the Output window. |
| **lr.output_message** | Sends a message to the log file and Output window with location information. |

In the following example, **lr.message** sends a message to the output indicating the loop number:

```
for(int i=0;i<10;i++)
   {
  lr.message("action()"+i);
    lr.think_time(2);
   }
```

For more information about the message functions, see the Function Reference (**Help > Function Reference**).

You can instruct the Vusers to redirect the Java standard output and standard error streams to VuGen's Execution log. This is especially helpful when you need to paste existing Java code or use ready-made classes containing **System.out** and **System.err** calls in your Vuser scripts. In the execution log, standard output messages are colored blue, while standard errors are shown in red.

The following example shows how to redirect specific messages to the standard output and standard error using lr.enable_redirection:

```
lr.enable_redirection(true);
System.out.println("This is an informatory message...");  // Redirected
System.err.println("This is an error message...");  // Redirected
lr.enable_redirection(false);
System.out.println("This is an informatory message...");  // Not redirected
System.err.println("This is an error message...");  // Not redirected
```

**Note:** When you set **lr.enable_redirection** to **true**, it overrides all previous redirections. To restore the former redirections, set this function to **false**.

For additional information about this function, see the Function Reference (**Help > Function Reference**).

## Emulate user think time

The time that a user waits between performing successive actions is known as the think time. Vusers use the lr.think_time function to emulate user think time. In the following example, the Vuser waits two seconds between loops:

```
for(int i=0;i<10;i++)
   {
    lr.message("action()"+i);
  lr.think_time(2);
   }
```

You can use the think time settings as they appear in the script, or a factor of these values. To configure how Vusers handle think time functions, open the runtime settings dialog box. For more information, see "Runtime Settings" on page 301.

For more information about the lr.think_time function, see the Function Reference (**Help > Function Reference**).

## Handle command line arguments

You can pass values to a Vuser script at runtime by specifying command line arguments when you run the script. You insert command line options after the script path and filename in the Controller or Business Process Monitor. There are three functions that allow you to read the command line arguments, and then to pass the values to a Vuser script:

| | |
|---|---|
| **lr.get_attrib_double** | Retrieves double precision floating point type arguments |
| **lr.get_attrib_long** | Retrieves long integer type arguments |
| **lr.get_attrib_string** | Retrieves character strings |

Your command line should have the following format, where the arguments and their values are listed in pairs after the script name:

```
script_name   - argument argument_value   -argument argument_value
```

The following example shows the command line string used to repeat script1 five times on the machine pc4:

```
script1  -host pc4  -loop 5
```

For more information on the command line parsing functions, see the Function Reference (**Help > Function Reference**).

For more information on how to insert the command line options, see "Run a Vuser Script from a Command Prompt" on page 323.

## Correlate Java Scripts

Automatic correlation is not supported in Java protocols, but you can manipulate script objects like any Java object.

For example: Given this business logic class:

```
public class RmiMessage implements Serializable {
    List<Integer> integers;

    public RmiMessage(List<Integer> myIntegers) {
        this.integers = myIntegers;
    }

    public void setIntegers(List<Integer> myIntegers) {
        this.integers = myIntegers;
    }
}
```

And given the following code in Action.java:

```
String _string9 = "com.comp.lr.testing.rmi.RmiMessage __CURRENT_OBJECT =
   {"
        + "java.util.List myIntegers = {"
             + "super = {"
                  + "super = {"
                  + "}"
                  + "int modCount = #0#"
             + "}"
             + "java.lang.Object a[] = {"
                  + "java.lang.Integer a[0] = {"
    + "super = {"
    + "}"
    + "int value = #4#"
```

```
                + "}"
                + "java.lang.Integer a[1] = {"
    + "super = {"
    + "}"
    + "int value = #21#"
                + "}"
           + "}"
        + "}"
     + "}";
RmiMessage _message = (com.comp.lr.testing.rmi.RmiMessagelr.deserialize(_
   string9, 2);
_remote.exampleMethodCall(_message);
```

After the call to lr.deserialize, you have the business logic object, which can then be manipulated.

In this example, correlation is done by setting the integers list:

```
RmiMessage _message = (com.comp.lr.testing.rmi.RmiMessage) lr.deserialize
   (_string9, 2);
_message.setIntegers(Arrays.asList(4,5,7)); // call business logic methods
   here to adjust object to your needs
_remote.exampleMethodCall(_message);
```

## Insert Parameters into Java Scripts

Parameter functions can be added for each response or request body text in a specific location. This location is indicated by a blank line, usually one to two lines below the start of the response or request body.

In the following example of a Java over HTTP script, parameter functions can be added to the blank lines in each requestBody section.

```
//////////////////////    requestBody2.xml    //////////////////////
RemoteInvocation RemoteInvocation_getUsernameList2 =
    (RemoteInvocation) JavaHTTP.readObject(RemoteInvocationBA0);
//INSERT PARAMETERIZATION AND CORRELATION CODE HERE
RemoteInvocationResult RemoteInvocationResult_ArrayList2 =
    (RemoteInvocationResult) JavaHTTP.sendSerialized(RemoteInvocation_getUsernameList2, 2,
"ObjectsDeserializerDefaultImpl",
"OrderService-httpinvoker",
"URL=http://kalimanjaro.devlab.ad:8080/jpetstore/remoting/OrderService-httpinvoker",
new String[]{
        "Method=POST",
        "Resource=0",
        "RecContentType=application/x-java-serialized-object",
        "Referer=",
        "Mode=HTML",
        "EncType=application/x-java-serialized-object",
        LAST});// 2 is the number of the header file, record time response is at file responseBody2.xml

//////////////////////    requestBody3.xml    //////////////////////
RemoteInvocation RemoteInvocation_getCategoryList3 =
    (RemoteInvocation) JavaHTTP.readObject(RemoteInvocationBA1);
//INSERT PARAMETERIZATION AND CORRELATION CODE HERE
RemoteInvocationResult RemoteInvocationResult_ArrayList3 =
    (RemoteInvocationResult) JavaHTTP.sendSerialized(RemoteInvocation_getCategoryList3, 3,
"ObjectsDeserializerDefaultImpl",
"OrderService-httpinvoker_2",
"URL=http://kalimanjaro.devlab.ad:8080/jpetstore/remoting/OrderService-httpinvoker",
new String[]{
        "Method=POST",
        "Resource=0",
        "RecContentType=application/x-java-serialized-object",
        "Referer=",
        "Mode=HTML",
        "EncType=application/x-java-serialized-object",
        LAST});// 3 is the number of the header file, record time response is at file responseBody3.xml
```

# Manually Configure Script Generation Settings

You can customize the way the navigator adds methods to your script.

## jquery.ini file structure

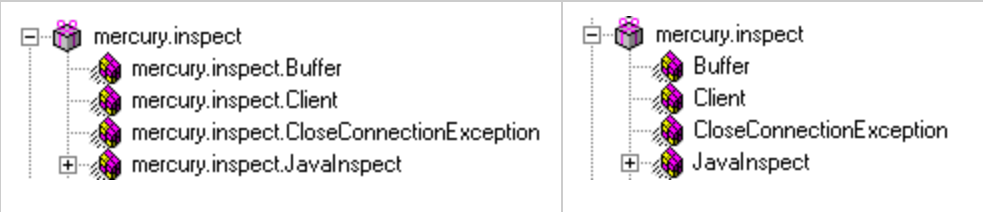
To view the configuration setting, open the **jquery.ini** file in VuGen's dat folder.

```
[Display]
FullClassName=False
[Insert]
AutoTransaction=False
DefaultValues=True
CleanClassPaste=False
```

## Class Name path

The **FullClassName** option displays the complete package and class name in the Java Function navigator. This option does not affect the way the functions are added into the script—it only affects the way the classes are displayed in the navigator. By default, this option is set to false. If your packages have many classes and you are unable to view the package and class names at the same time, you should enable this option.

| FullClassName enabled | FullClassName disabled |
|---|---|




## Automatic transactions

The **AutoTransaction** setting creates a Vuser transaction for all methods. When you enable this option, VuGen automatically encloses all Java methods with **lr.start_transaction** and **lr.end_transaction** functions. This allows you to individually track the performance of each method. This option is disabled by default.

```
lr.start_transaction("get_host_name");

(String) = lr.get_host_name():

lr.end_transaction("get_host_name", lr.AUTO);


lr.start_transaction("isSystemClass");

(boolean) = isSystemClass ((String) "");

lr.end_transaction("isSystemClass", lr.AUTO);
```

## Default parameter values

The **DefaultValues** setting includes default values for all methods you paste into your script. This option is enabled by default and inserts a null for all objects. If you disable this option, you must manually insert parameter values for all functions in the script. The following table illustrates the DefaultValues flag enabled and disabled.

| DefaultValues enabled | DefaultValues disabled |
|---|---|
| `lr.message((String)"");`<br>`lr.think_time((int)0);`<br>`lr.enable_redirection((boolean)false);`<br>`lr.save_data((byte[])null, (String)"");` | `lr.message((String));`<br>`lr.think_time((int));`<br>`lr.enable_redirection((boolean));`<br>`lr.save_data((byte[]), (String));` |

## Class pasting

The **CleanClassPaste** setting pastes a class so that it will compile cleanly: with an instance returning from the constructor, with default values as parameters, and without a need for import statements.

Using this option, you will most likely be able to run your script without any further modifications. If you disable this option (default), you may need to manually define parameters and include import statements.

> **Note:** This setting is only effective when you paste an entire class into your script—not when you paste a single method.

The following segment shows the **toString** method pasted into the script with the **CleanClassPaste** option enabled.

```
_class.toString();
     // Returns: java.lang.String
```

The same method with the **CleanClassPaste** option disabled is pasted as follows:

```
(String) = toString();
```

The next segment shows the **NumInserter** Constructor method pasted into the script with the **CleanClassPaste** option enabled.

```
utils.NumInserter _numinserter = new utils.NumInserter
                ((java.lang.String)"", (java.lang.String)"",
(java.lang.String)""...);
// Returns: void
```

The same method with the **CleanClassPaste** option disabled is pasted as:

```
new utils.NumInserter((String)"", (String)"", (String)"",...);
```

## Run Java Vuser Scripts

Java Vuser scripts differ from C Vuser scripts in that they are first compiled and then executed; C Vuser scripts are interpreted.

VuGen locates the **javac** compiler from within the JDK installation and compiles the Java code inside the script. This stage is indicated by the **Compiling...** status message in the bottom of the VuGen window. If errors occur during compilation, they are listed in the execution log.

To go to the code in your script that caused the error, double-click on the error message containing the line number of the error. Fix the error and run the script again.

If the compilation succeeds, the status message **Compiling...** changes to **Running...** and VuGen begins to execute the script. When you run the script again, VuGen runs the script without recompiling it, provided that no changes were made to the script.

To debug your script further, you can use breakpoints and animated run type execution using the step option.

**Note:** If you are making calls to JNDI extensions within your script, you may encounter problems trying to run your Vusers as **threads**. This happens because JNDI requires each thread to have its own context class loader. In order to run as threads, instruct each Vuser to run with its own context class loader, by adding the following line to the beginning of the **init** section:

```
DummyClassLoader.setContextClassLoader();
```

### See also:

- "Compile and Run a Script as Part of a Package" below
- "Edit and Run Scripts in Eclipse" below

## Compile and Run a Script as Part of a Package

When creating a Java Vuser script, you may need to use methods in other classes in which the class or method is protected. If you try to compile this type of script, you will receive errors in the compilation stage indicating that the methods are inaccessible.

To make sure that your script can access these methods, insert the package name containing these methods at the top of the script, just as you would do in a standard Java program— <package_name>.

**Example:**

This script defines the **my.test** package which consists of a path. Not that the package statement must be the first line in the script, similar to Java (excluding comments).

```
package my.test;
import lrapi.*;
public class Actions
{

}
```

VuGen automatically creates the **my/test** folder hierarchy under the Vuser folder, and copies the **Actions.java** file to **my/test/Actions.java**, allowing it to compile with the relevant package.

## Edit and Run Scripts in Eclipse

You can use supported versions of Eclipse to take advantage of additional tools that enable you to view, edit, and debug your Java Vuser (Java Record Replay, and Java over HTTP) scripts. You can add breakpoints, view variable values, add references, and edit the script using IntelliSense. You can also run the script in a step-by-step mode for debugging.

When you save your script, VuGen creates Java source files in your script's folder. You can open the solution file in Eclipse and view all of its components in the Projects Explorer.

To open the Vuser script in Eclipse, click the **Open Script in Eclipse** button on the VuGen toolbar.

> **Note:**
>
> - If this is your first time using Eclipse from within VuGen, it will automatically install the Eclipse plugin.
> - Before opening a script in Eclipse, you need to set the location of the Eclipse IDE in the **Java** node of the VuGen's Scripting options. If you do not set this value, VuGen prompts you to select its location. For details, see "Scripting Options Tab" on page 119.
>   - For details on supported Eclipse versions, see the System Requirements.

An additional toolbar menu provides access to common VuGen commands, such as Runtime Settings, Parameter List, Run, and Stop.

VuGen also has an add-in for Eclipse developers that allows you to create JUnit tests that can be called directly from the testing application, such as the LoadRunner Controller, without having to open them in VuGen. The add-ins are located in your LoadRunner installation package's **/Additional Components** folder. For details, see "Additional Components" on page 1566.

**To open the Vuser script in Eclipse:**

1. Make sure you have Eclipse 4.2 or higher on your machine, running with JDK 1.7 or later.
2. Set the location of the Eclipse IDE in the **Scripting > Java** node in VuGen's Options dialog box. For details, see "Scripting Options Tab" on page 119.
3. Create a Java script (Java Vuser, Java Record Replay, Java over HTTP, and so forth).
4. Click the **Open Script in Eclipse** button on the VuGen toolbar. If this is your first time using Eclipse from within VuGen, it will automatically install the VuGen Eclipse plugin.
5. Double-click the appropriate section, such as **Actions.java**, to edit the code.
6. Use the **Vuser** menu to define parameters, configure runtime settings, and run the script directly from the Eclipse IDE.

### See also:

- "Creating Vuser Scripts or Unit Tests in Visual Studio or Eclipse" on page 877.

## Debug Java Scripts

> **Note:** This topic applies to the Java Record Replay and Java over HTTP protocols only.

This task describes how to debug Java scripts by comparing the request and response data from the record and replay stages.

1. Add arguments to the VM Param Node

   Select **Replay > Runtime Settings > Java VM** node. In the **Additional VM Parameters** field, enter the following string:

   ```
   -DdumpServerRequests=true -DdumpServerResponses=true
   ```

2. Compare record and replay data

   In the Solution Explorer, right-click the script name and select **Open Script Folder**. The data from the recording phase is in the main folder. The data from the replay phase is in the replay folder.

   The files that follow the format RequestBodyX contain the request data. The files that follow the format ResponseBodyX contain the response data.

   To compare the record and replay data for the purposes of debugging, compare the files with identical names from the recording and replay phases. For example, compare the RequestBody1 file from the main folder (recording phase) to the RequestBody1 file from the replay folder. Normally, the files should be identical. Cases where the files are not identical may indicate problems in the script.

3. Remove arguments before load testing

   Return to the Java VM node and the items you added to the Additional VM Parameters field.

## Notes and Limitations - Java Vuser protocols

This section describes notes and limitations for the Java protocols.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

### Notes

- 64-bit replay is not supported for Java over HTTP protocol. and for Java Record Replay or Java Vuser protocols when using web packages
- 64-bit replay is not supported for Java Record Replay or Java Vuser protocols when using web packages.
- **Java Record Replay and Java over HTTP protocols**: When recording on Internet Explorer 8 using the Java protocols, you must first close all instances of Internet Explorer before LoadRunner opens an Explorer instance for the record session.
- VuGen provides a tool that enables you to convert a script created for Web, into Java. For more information, see "Convert a Web - HTTP/HTML Vuser Script into a Java Vuser Script" on page 713.

> **Note:** By default, Java 7 enables the Java Split Verifier. This prevents Java recording. VuGen

> uses the **-XX:-UseSplitVerifier** key while initializing the JVM during recording, to disable the verifier. This adaptation does not require any user intervention.

- Although .NET-based and Java protocols support creating threads, we recommend that you do not use background threads in real load testing scenarios because:
  - Threads can degrade tests scalability.
  - Threads can affect performance measurements.
  - The utility functions' behavior is undetermined if called from any thread except the Vuser main thread which runs the vuser_init, Action, and vuser_end actions. This applies to all functions named lr*.

### Limitations

- When you run a Java script, the replay status may be **Script Not Run** and some errors may appear in the mdrv.log file. However, due to Java internal architecture, these errors may not be included in the VuGen Output and Errors panes. This occurs when VuGen fails to initialize a Java Vuser, and JVM then terminates the replay process.

  **Workaround:** Look for errors directly in the mdrv.log file. If the entry in the log is due to a memory-related issue, try using different memory options for Java in the runtime settings.

### See also:

- "Notes, Limitations, and Troubleshooting - Java over HTTP" on page 588

# Java Record Replay Protocol

## Supported Java communication protocols

The Java Record Replay (JRR) protocol enables full VuGen functionality when recording a script on a Java application or applet. VuGen creates a script in pure Java and enhances it with Java-specific functions.

VuGen supports a variety of different communication protocols for Java applications:

| Protocol | Description |
|---|---|
| **RMI** | See "Working with RMI" on page 569 |
| **CORBA** | See "Working with CORBA" on page 571 |
| **JACADA** | See "Working with Jacada" on page 573 |
| **JMS** (Java Message Service). | Supports messaging between computers in a network |
| **JDBC** (Java Database Connectivity) | Used for defining how a client may access a database |

VuGen's built-in support for the Java protocols utilizes hook files to define how different classes communicate with each other. By default, VuGen only records client side activity in a script, which then emulates the load on the server. You must edit the hook file to change the actions recorded by VuGen.

**Note:** If you are recording a script that does not use one of the supported protocols above, you must define your own hook file. Otherwise your Vuser script will be empty.

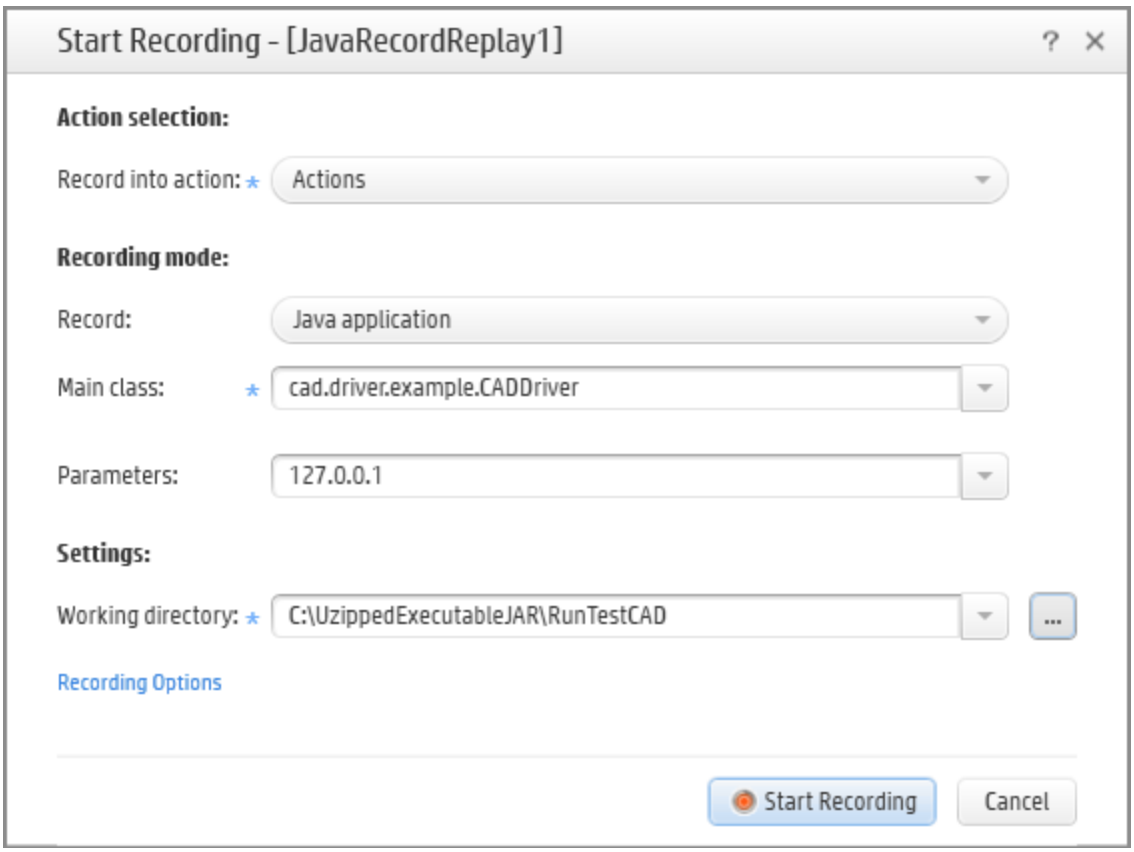
For details, see "Custom Java Filters" on page 577.

## Java Record Replay Protocol Recording Tips

This section gives tips and important information to consider when recording a Java Record Replay Vuser Script.

- Make sure that you have properly installed a JDK on the machine running the Vusers. For details, see "Set Up the Java Environment" on page 550.
- The Java Record Replay protocol can record 32- and 64-bit applications.
- 64-bit replay is not supported when using web packages.
- You can also specify a batch (.bat) file as the application to record.
- Make sure to fill in the correct information in the Start Recording dialog box.

**Note:** The executable's jar file must be unzipped.




- When you load an applet or application from VuGen during recording, it may take several seconds longer than if you were to load it independent of VuGen.
- If you intend to run the Java Vuser script as a thread, ensure your code is thread-safe.

  Although .NET-based and Java protocols support creating threads, we recommend that you do not use background threads in real load testing scenarios because:
  - Threads can degrade tests scalability.
  - Threads can affect performance measurements.
  - The utility functions' behavior is undetermined if called from any thread except the Vuser main thread which runs the vuser_init, Action, and vuser_end actions. This applies to all functions named lr*.

### See also:

- "VuGen Workflow" on page 136
- "Recording - Overview" on page 154

## Working with RMI

VuGen provides full support for the RMI over IIOP protocol.

This section describes the elements of the Java Vuser script that are specific to RMI.

Depending on what you are recording, you can utilize VuGen's RMI recorder to create a script that will optimally emulate a real user for:

- Pure RMI client: Recording a client that uses native JRMP protocol for remote invocations
- RMI over IIOP client: Recording a client application that was compiled using the IIOP protocol instead of JRMP .

RMI does not have constructs. Instead, it uses Serializable Java objects. In RMI there is no specific shutdown section.

The following code example locates a naming registry and utilizes a lookup operation to obtain a specific Java object. You can then perform functions such as **set_sum**, **increment**, and **get_sum** on the object. You must import the RMI classes to access the RMI functions.

**Example:**

```
Import java.rmi.*;
Import java.rmi.registry.*;

public int action() throws Throwable {

    _registry = LocateRegistry.getRegistry("localhost",1099);
    counter = (Counter)_registry.lookup("Counter1");
    counter.set_sum(0);
    counter.increment();
    counter.increment();
    counter.get_sum();
    return lr.PASS;
    }
```

When recording RMI Java, your script may contain several calls to **lr.deserialize**, which deserializes all of the relevant objects. The **lr.deserialize** calls are generated because the object passed to the next invocation could not be correlated to a return value from any of the previous calls. VuGen therefore records its state and calls the **lr.deserialize** function to represent these values during replay.

The deserialization is done before VuGen passes the objects as parameters to invocations. For more information, see "Java Scripts Correlation - Serialization" on page 278.

## Connection timeouts

To set connection timeouts, add Java code to set the properties.

The following example shows setting RMI timeouts to 5 seconds:

**Example:**

```
import lrapi.lr;
public class Actions {
```

```
  public int init() throws Throwable {
    return 0;
  }
  public int action() throws Throwable {
    java.util.Properties properties=System.getProperties();

    properties.put("sun.rmi.transport.tcp.responseTimeout", 5000);
    properties.put("sun.rmi.transport.tcp.readTimeout", 5000);
    properties.put("sun.rmi.transport.connectionTimeout", 5000);
    properties.put("sun.rmi.transport.handshakeTimeout", 5000);

    properties.put("user.script","");
    System.setProperties(properties);
    java.lang.String var_0="rmi://example.com/RmiServer";
    RmiServerIntf var_1=(RmiServerIntf)java.rmi.Naming.lookup(var_0);
    java.lang.String var_2=var_1.getMessage();
    return 0;
  }
  public int end() throws Throwable {
    return 0;
  }
}
```

To set this timeout to all TCP connections:

```
    properties.put("sun.net.client.defaultReadTimeout", 5000);
    properties.put("sun.net.client.defaultConnectTimeout", 5000);
```

## Working with CORBA

**Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

CORBA (Common Object Request Broker Architecture) is a powerful distributed application development architecture.

**Note:** Java 8 and later versions are not supported in CORBA.

### CORBA Application Vendor Classes

Running CORBA applications with JDK1.2 or later might load the JDK internal CORBA classes instead of the specific vendor CORBA classes. To force the virtual machine to use the vendor classes, specify the following java.exe command-line parameters:

**Visigenic 3.4**

```
-Dorg.omg.CORBA.ORBClass=com.visigenic.vbroker.orb.ORB
-Dorg.omg.CORBA.ORBSingletonClass=com.visigenic.vbroker.orb.
     ORBSingleton
```

**Visigenic 4.0**

```
-Dorg.omg.CORBA.ORBClass=com.inprise.vbroker.orb.ORB
-Dorg.omg.CORBA.ORBSingletonClass=com.inprise.vbroker.orb.ORBSingleton
```

**OrbixWeb 3.x**

```
-Dorg.omg.CORBA.ORBClass=IE.Iona.OrbixWeb.CORBA.ORB
-Dorg.omg.CORBA.ORBSingletonClass=IE.Iona.OrbixWeb.CORBA.
     singletonORB
```

**OrbixWeb 2000**

```
-Dorg.omg.CORBA.ORBClass=com.iona.corba.art.artimpl.ORBImpl
-Dorg.omg.CORBA.ORBSingletonClass=com.iona.corba.art.artimpl.
     ORBSingleton
```

## Editing a CORBA Vuser Script

CORBA-specific scripts usually have a well-defined pattern:

- The first section contains the ORB initialization and configuration.
- The next section indicates the location of the CORBA objects.
- The following section consists of the server invocations on the CORBA objects.
- The final section includes a shutdown procedure which closes the ORB.

The pattern is not mandatory and each one of these sections may appear multiple times within a script.

In the following segment, the script initializes an ORB instance and performs a bind operation to obtain a CORBA object. VuGen imports all the necessary classes.

**Example:**

```
import org.omg.CORBA.*;
import org.omg.CORBA.ORB.*;
import lrapi.lr;

public class Actions {

   public int init() throws Throwable {
// Initialize Orb instance...
   MApplet mapplet = new MApplet("http://chaos/classes/", null);
   orb = org.omg.CORBA.ORB.init(mapplet, null);

// Bind to server...
   grid = grid_dsi.gridHelper.bind("gridDSI", "chaos");
   return lr.PASS;
}
```

The `org.omg.CORBA.ORB` function makes the connection to ORB. Therefore, it should only be called once. When running multiple iterations, place this function in the **init** section.

In the following section, VuGen recorded the actions performed on a grid CORBA object.

```
 public int action() throws Throwable {

   grid.width();
   grid.height();
   grid.set(2, 4, 10);
   grid.get(2, 4);
   return lr.PASS;
}
```

At the end of the session, VuGen recorded the shutdown of the ORB. The variables used throughout the entire recorded code appear after the **end** method and before the Actions class closing curly bracket.

```
 public int end() throws Throwable {
        if (lr.get_vuser_id() == -1)
            orb.shutdown();
        return lr.PASS;
}
// Variable section
    org.omg.CORBA.ORB orb;
    grid_dsi.grid ;
}
```

> **Note:** The ORB shutdown statement was customized for this product. This customization prevents a single Vuser's shutdown from shutting down all other Vusers.

## Working with Jacada

The Jacada Interface Server provides an interface layer for mainframe applications. This layer separates the user interface from the application logic in order to insulate the organization from changes in standards and technologies.

> **Note:** Java 8 is not supported in Jacada.

### Recording a Jacada Vuser

VuGen records Jacada's Java thin-client.

To record communication with the Jacada server through the HTML thin-client, use the Web HTTP/HTML type Vuser. For more information, see "Web Protocols (Generic)" on page 729.

## Editing a Jacada Vuser Script

The Actions method of a Java Vuser script using Jacada, has two main parts: properties and body. Use the properties section to retrieve and set the server properties. Once your have the server properties you can connect to the Jacada server.

```
// Set system properties...
       _properties = new Properties(System.getProperties());
       _properties.put("com.ms.applet.enable.logging", "true");
       System.setProperties(_properties);

       _jacadavirtualuser = new cst.client.manager.JacadaVirtualUser();

       lr.think_time(4);
       _jacadavirtualuser.connectUsingPorts("localhost", 1100, "LOADTEST", "", "",
"");
```

The body of the script contains the user actions along with the exception handling blocks for the checkFieldValue and checkTableCell methods.

**Example:**
```
try {
        _jacadavirtualuser.checkFieldValue(23, "S44452BA");
        } catch(java.lang.Exception e) {
             lr.log_message(e.getMessage());
        }

try {
       _jacadavirtualuser.checkTableCell(41, 0, 0, "");
        } catch(java.lang.Exception e) {
             lr.log_message(e.getMessage());
        }
```

The **checkField** method has two arguments: field ID number and expected value. The **checkTableCell** method has four arguments: table ID, row, column, and expected value. If there is a mismatch between the expected value and the received value, an exception is generated.

By default, the try-catch wrapper blocks are commented out. To use them in your script, remove the comment markers.

In addition to the recorded script, you can add any of the Java Vuser API functions. For a list of these functions and information on how to add them to your script, see "Java Vuser (Manual) Protocol" on page 583.

## Replaying Jacada Scripts

Before replay, you must download the **clbase.jar** file from the Jacada server.

All extra files such as jars, classes, and so on, used by the Java Vuser must be accessible by the script by doing one of the following:

- Add the files to the script's **Extra Files** folder: In the Solution Explorer pane, right click the **Extra Files** node and select **Add Files to Script**. The files are copied to the script's folder.
- In the script's runtime settings, in **Java Environment > Classpath**, set the classpath to the Java classes. These files are then used as a reference, but are not contained in the script's folder.

During replay, the Jacada server may return screens from the legacy system in a different order than they appear in the recorded script. This may cause an exception in the replay. For information on how to handle these exceptions, contact Customer support.

## Working with JDBC

VuGen provides full support for the JDBC protocol which is used for defining how a client may access a database.

> **Note:** Support for Java 8 only.

### Supported Databases

The JDBC protocol supports all databases used via standard JDBC interface.

It also has custom support for non-standard usage of following databases:

- Postgres
- MySQL
- Oracle
- MSSQL
- H2

### Recording JDBC Scripts

To record JDBC script use the Java Record Replay protocol.

In Recording Options:

- Under **Recording Protocol > Recorder Options**, select **JDBC** and record as with any Java Record Replay script
- To avoid long and repetitive scripts, configure VuGen to generate loops where necessary by selecting **Recording Protocol > Code Generation Options > Search for loops**

For more information about recording Java Record Replay scripts, see "Java Record Replay Protocol Recording Tips" on page 568.

## Limitations

- Blob and Clob class recording is not supported.
- Advanced non-standard features are not supported.

## Sample Script

```
import lrapi.lr;

public class Actions{

	public int action() throws Throwable{
		_properties1 = System.getProperties();
		_properties1.put("com.comp.lr.java2.record.format", "XML");
		System.setProperties(_properties1);

		_driver1 = new org.h2.Driver();
		_driver2 = org.h2.Driver.load();

		_string1 = "jdbc:h2:~/test";
		_string2 = "user";
		_string3 = lr_unmask("3fbae0ecdbfa75c3");
		_connection1 = java.sql.DriverManager.getConnection(_string1, _string2, _string3);

		_string4 = "SELECT count(1) FROM USERS";
		_preparedstatement1 = _connection5.prepareStatement(_string4);

		_resultset1 = _preparedstatement1.executeQuery();

		_boolean1 = _resultset1.next();

		_int1 = _resultset1.getInt(1);
		_connection1.close();

		return 0;
	}// end of method: public int action() throws Throwable

	public int init() throws Throwable{
		return 0;
	}// end of method: public int init() throws Throwable

	public int end() throws Throwable{
		return 0;
	}// end of method: public int end() throws Throwable

	/* ------ fields ------ */
```

```
        java.util.Properties _properties1;
        org.h2.Driver _driver1;
        org.h2.Driver _driver2;
        java.lang.String _string1;
        java.lang.String _string2;
        java.lang.String _string3;
        java.sql.Connection _connection1;

        java.lang.String _string4;
        java.sql.PreparedStatement _preparedstatement1;
        java.sql.ResultSet _resultset1;
        boolean _boolean1;
        int _int1;
}
```

## Custom Java Filters

This section describes when and how to create a custom hook file for Java method filtering.

### Overview

When recording a Java Record Replay script, the recorder selects which methods are recorded in the Vuser script and which are left out. VuGen uses hooking to filter the methods to include in the script.

VuGen comes with built-in support for the **RMI**, **JMS**, **Jacada**, and **JDBC** protocols. The built-in filters for these protocols are designed to record only the server-related traffic relevant to your testing goals.

If your protocol is not supported, you must define your own custom hook file. Custom Java protocols, proprietary enhancements and extensions to the default protocols, and data abstraction all require a custom filter definition.

> **Note:** If no hook file is defined for your protocol, your Vuser script will be empty.

Creating a custom hook file demands planning and a good understanding of the protocols your application uses. When hooking is implemented correctly, the Vuser script should be well correlated and ready for compilation. For more details on how to select which methods and classes to hook, see "Determining which elements to include" on the next page.

When you record a method, the methods that are called from the recorded method—whether directly or indirectly—are not recorded.

To record a method, VuGen must recognize the object upon which the method is invoked, along with the method's arguments. VuGen recognizes an object if it is returned by another recorded method provided that:

- The construction method of that object is hooked
- The object is a primitive or a built-in object
- The object supports a serializable interface

You can create a custom filter to exclude unwanted methods. When recording a Java application, your script may include calls to methods that do not affect the server, such as calls to a local utility or the GUI interface. These calls are usually not relevant to your testing goals, and it would be correct to filter them out.

Before creating a test, we recommend that you become familiar with your application and determine its primary classes and methods, so that you will know which ones to include in your recording.

If you are not familiar with your application's classes, VuGen allows you to record with a stack trace that logs all of the methods that were called by your application. To record with stack trace set the log level to **Detailed**.

## Determining which elements to include

When designing a custom filter, we recommend that you start by choosing the appropriate built-in filter as a base filter. You can then customize the filter using one of the following approaches:

| | |
|---|---|
| **Top-down** | Include the relevant package and exclude specific classes that are not part of the client-server activity. This is recommended if you are familiar with your application and you can identify a well-defined layer that implements all client-server activity without involving any GUI elements. |
| **Bottom-up** | Use the default filter and refine it by adding individual methods or classes. Use this approach if you cannot identify a well-defined layer or if you are not familiar with your application. Do not add all AUT packages and then try to remove extra component one by one. |

Include or exclude elements as follows:

- If, as a result of your including a class, your script has many unrelated method calls, try modifying the filter to exclude the irrelevant methods.
- If you identify a non-client/server call in your script, exclude its method in the filter.
- During recording, VuGen may detect an unknown input argument, for example, an argument whose construction it had never encountered before. If this argument supports serialization, VuGen serializes it by saving it to a file in a special format. During replay, VuGen reconstructs the argument by deserializing it.
- VuGen serializes objects passed as arguments that were not included by the filter. We recommend that you include this object in the filter in order to track its construction and activity instead of using it in its serialized form. You can identify serialized objects in the script by searching for calls to the **lr.deserialize()** method in your script. For more information see "Java Scripts Correlation - Serialization" on page 278.
- Exclude all activity that involves GUI elements.

- Add classes for utilities that may be required for the script to be compiled.

## Create a custom Java filter

When preparing a script, you may need to customize the filter several times in order to achieve the optimal filter. An optimal filter records the relevant methods without introducing a large number of irrelevant calls to the script.

> **Note:** Rerecording a script after modifying the filters will overwrite all manual changes. Therefore, if you plan to add manual code to your script such as control flow or message statements, make sure to do so after you have a functional script that runs inside VuGen.

**To define a custom hook file:**

1. Create a **user.hooks** file in the VuGen installation classes folder, typically **C:\Program Files (x86)\Micro Focus\Virtual User Generator\classes**. VuGen automatically searches for this file when recording. For structural details about the hook file, see "Hook file structure" on the next page.
2. Open the Recording Options dialog box (Ctrl+F7) and select the **Log Options** node. For **Log Level**, select **Detailed**.
3. Record your application. Click **Start Record** (Ctrl + R) to begin and **Stop** (Ctrl + F5) to end.
4. View the script's steps. If you can determine the business logic from the steps and apply correlation, you may not need to create custom filters. If, however, the script is very long or hard to maintain and correlate, you should customize the script's filter.
5. Try to identify the high-level method in the call that captures or wraps one or more client server calls. You can do this by opening the AUT source files (if they are available) or by viewing a Stack Trace of the script.
6. Set the filter to include the relevant methods. For more information, see "Determining which elements to include" on the previous page.
7. Record the application again. You should always rerecord the application after modifying the filter.
8. Repeat steps 4 through 7 until you get a simple script which can be maintained and correlated.
9. Correlate the script. In order for your test to run properly, you may need to insert a correlation to capture a value and use it at a later point in the script. For more information about the built-in correlation mechanism, see "Java Scripts Correlation - Serialization" on page 278.

> **Caution:**
>
> - Adding custom hooks to the default recorder is a complicated task and should be considered carefully as it has both functional and performance consequences.
> - Do not modify any of the other .hook files as this might damage the VuGen recorder.
> - Incorrect hooking definitions can lead to incorrect scripts, slow recording, and application freeze-up.

## Hook file structure

The following section describes the structure of a typical hook file used when creating a custom filter. For details, see "Create a custom Java filter" on the previous page.

The hook files are structured as .ini files where each section represents a hook definition. Regular expressions are supported in some of the entries. Any entry that uses a regular expression must start with an exclamation point (!).

> **Note:** When you use a filter such as **!.***, the **!** indicates the beginning of a RegExp—not a Regexp negation.

```
[hook-Name]
class      = MyPackage.MyClass
method     = MyMethod
signature  = ()V
ignore_cl =
ignore_mtd =
ignore_tree =
cb_class   = mercury.ProtocolSupport
cb_mtd =
general_cb = true
deep_mode = soft | hard
make_methods_public = true | false
lock = true | false
```

## Structure description

| | |
|---|---|
| **hook-Name** | Specifies the name of this section in the hook file. Hook-Name must be unique across all hook files. A good practice is to give the fully qualified class name and method.<br>For example:<br>`[javax.jms.Queue.getQueueName]` |
| **class** | A fully qualified class name.<br>Regular expressions can be used to include several classes from the same package, a whole package, several packages, or any class that matches a name.<br>The following example filters for any class that starts with **javax.jms** and is followed by at least one character:<br>`class = !javax\.jms\.*` |

| | |
|---|---|
| **method** | The simple name of the method to include.<br>Regular expressions can be used to include more than one method from the class.<br>For example:<br>`method = getQueueName` |
| **signature** | The standard Java internal type signature of the method.<br>To determine the signature of a method, run the command<br>`javap -s class-name`<br>where `class name` is the fully qualified name of the class.<br>Regular expressions can be used to include several methods with the same name, but with different arguments.<br>For example:<br>• `signature = !.*` will match any possible signature, thus causing any method in this class to be recorded into the script regardless of signature.<br>• `signature = !\(Lorg/omg/CORBA/ORB;\).*` will match any signature starting with `(Lorg/omg/CORBA/ORB;)`. |
| **ignore_cl** (optional) | A specific class to ignore from the classes that match this hook.<br>This can be a list of comma-separated class names. Each item in the list can contain a regular expression. If an item in the list contains a regular expression, prepend an exclamation point (**!**) to the class name.<br>For example:<br>`Ignore_cl = !com.comp.jms.Queue,!com\.comp\..*` |
| **ignore_mtd** (optional) | A specific method to ignore.<br>When the loaded class method matches this hook definition, this method will not be hooked. The method name must be the simple method name followed by the signature (as explained above).<br>To ignore multiple methods, list them in a comma-separated list.<br>To use a regular expression, prepend an exclamation point (**!**) to the method name.<br>For example:<br>`ignore_cl = open, close` |

| | |
|---|---|
| **ignore_tree** (optional) | A specific tree to ignore.<br>When the name of the class matches the ignore tree expression, any class that inherits from it will not be hooked, if it matches this hook definition.<br>To ignore multiple trees, list them in a comma-separated list.<br>To use a regular expression, prepend an exclamation point (**!**) to the class name.<br>This option is relevant only for hooks that are defined as deep. |
| **cb_class** | The callback class that gets the call from the hooked method. It should always be set to **mercury.ProtocolSupport**.<br>The built-in hook definitions may use other classes, such as mercury.jms.JMSSupport. |
| **cb_mtd** (optional) | A method in the callback class that gets the call from the hooked method. If omitted, it uses the default, **general_rec_func**.<br>For cases where you only need to lock the subtree of calls, use **general_func** instead. |
| **general_cb** | The general callback method.<br>This value should always be set to **true**. |
| **deep_mode** | Deep mode refers to classes and interfaces that inherit or implement the class or interface that the hook is listed for. The inherited classes will be hooked according to the type of hook: **Hard**, **Soft**, or **Off**.<br>• **Hard.** Hooks the current class and any class that inherits from it. If regular expressions exist, they are matched against every class that inherits from the class in the hook definition. Interface inheritance is treated the same way as class inheritance.<br>• **Soft.** Hooks the current class and any class that inherits from it, only if the methods are overridden in the inheriting class.<br>If the hook lists an interface, then if a class implements this interface those methods will be hooked. If they exist in classes that directly inherit from that class they will also be hooked.<br>However, if the hook lists an interface and a class implements a second interface that inherits from this interface, the class will not be hooked.<br>**Note:** Regular expressions are not inherited but converted to actual methods.<br>• **Off.** Only the class listed in the hook definition and the direct inheriting class will be hooked. If the hook lists an interface, only classes that directly implement it will be hooked. |

| | |
|---|---|
| **make_ methods_ public** (optional) | Any method that matches the hook definition will be converted to public.<br>This is useful for custom hooks or for locking a sub tree of calls from a non-public method.<br>**Note:** Applies only during record. During replay, the method will use the original access flags. In the case of non-public methods, it will throw `java.lang.VerifyError`. |
| **lock** (optional) | When set to **true** (default), this locks the sub tree and prevents the calling of any method originating from the original method.<br>When set to **false**, this unlocks the sub tree, records any method originating from the current method (if it is hooked), and invokes the callback. |

### See also:

- "Java Record Replay Protocol" on page 567
- "Java Protocols" on page 549

# Java Vuser (Manual) Protocol

To prepare Vuser scripts using Java code, use the **Java** type Vusers. This Vuser type supports Java on a protocol level. The Vuser script is compiled by a Java compiler and supports all of the standard Java conventions. For example, you can insert a comment by preceding the text with two forward slashes "//".

The first step in creating a Java compatible Vuser script is to create a new Vuser script using the **Java Vuser** protocol. Then program or paste the desired Java code into the script. You can add Java Vuser functions to enhance the script and parameterize the arguments to use different values during iterations. For details, see "Manually Create a Java Script" on the next page.

The Java Vuser script runs as a scalable multi-threaded application. If you include a custom class in your script, make sure that the code is thread-safe. For details, see "Java Protocol Programming Tips" on page 551.

After you prepare a script, run it as a standalone test from VuGen. A Java compiler (javac) checks it for errors and compiles the script.

Then integrate the script into your environment: a LoadRunner scenario, Performance Center load test, or Business Process Monitor configuration.

> **Note:** Java Vuser protocol does not support 64-bit replay when using web packages.

### See also:

- "Manually Create a Java Script" on the next page
- "Java Protocols" on page 549

- "Java Record Replay Protocol" on page 567 for recording and replaying Java scripts
- "Java over HTTP Protocol" on page 586 for recording and replaying Java remote calls over HTTP

## Manually Create a Java Script

This task describes how to manually create and edit a custom Java script using the **Java Vuser** protocol.

> **Note:** To work with Java scripts, make sure that your Java environment is set up. For details, see "Set Up the Java Environment" on page 550.

1. Create a new script:
   a. Open VuGen.
   b. Select **File > New** or click the **New** button. The New Virtual User dialog box opens.
   c. From the Select Vuser type list, select **Custom > Java Vuser** and click **OK**. VuGen displays a blank Java Vuser script.
   d. Click the **Actions** section in the left frame to display the **Actions** class.
2. Insert your code into the script

   After generating an empty template, you can insert the desired Java code. When working with this type of Vuser script, you place all your code in the **Actions** class. To view the Actions class, click **Actions** in the left pane. VuGen displays its contents in the right pane.

```
import lrapi.*;
public class Actions
{
    public int init() {
        return 0;
    }
    public int action() {
        return 0;
    }
    public int end() {
        return 0;
    }
}
```

   The Actions class contains three methods: init, action, and end. The following table shows what to include in each method and when each method is executed.

| Script method | Used to emulate... | Is executed when... |
|---|---|---|
| init | a login to a server | the Vuser is initialized (loaded) |
| action | client activity | the Vuser is in "Running" status |
| end | a log off procedure | the Vuser finishes or is stopped |

**Init Method**

Place all the login procedures and one-time configuration settings in the init method. The init method is only executed once—when the Vuser begins running the script. The following sample init method initializes an applet. Make sure to import the **org.omg.CORBA.ORB** function into this section, so that it will not be repeated for each iteration.

```
import org.omg.CORBA.*;
import org.omg.CORBA.ORB.*;
import lrapi.lr;
// Public function: init
    public int init() throws Throwable {
    // Initialize Orb instance...
    MApplet mapplet = new MApplet("http://chaos/classes/", null);
    orb = org.omg.CORBA.ORB.init(mapplet, null);
...
```

**Action Method**

Place all Vuser actions in the action method. The action method is executed according to the number of iterations you set in the runtime settings. For more information on the iteration settings, see "Runtime Settings" on page 301. The following sample action method retrieves and prints the Vuser ID.

```
public int action() {
   lr.message("vuser: " + lr.get_vuser_id() + " xxx");
        return 0;
  }
```

**End Method**

In the **end** method, place the code you want the Vuser to execute at the end of the script, such as logging off from a server, cleaning up the environment, and so forth.

The end method is only executed once—when the Vuser finishes running the script. In the following example, the end method closes and prints the end message to the execution log.

```
public int end() {
       lr.message("End");
       return 0;
   }
```

3. Insert additional API functions

   VuGen provides Java-specific functions for Java Vuser scripts. These functions are all static methods of the lrapi.lr class.

   The Java API functions are classified into several categories: Transaction, Command Line Parsing, Informational, String, Message, and Runtimefunctions.

For more information about each of these functions, see the Function Reference (**Help > Function Reference**). Note that when you create a new Java Vuser script, the import lrapi.* is already inserted into the script.

4. Insert additional Java functions

   To use additional Java classes, import them at the beginning of the script as shown below.

   Make sure that the additional classes are thread-safe and scalable.

```
import java.io.*;
import lrapi.*;
public class Actions
{
...
}
```

5. Set up the Java classes

   All extra files such as jars, classes, and so on, used by the Java Vuser must be accessible by the script by doing one of the following:

   - Add the files to the script's **Extra Files** folder: In the Solution Explorer pane, right click the **Extra Files** node and select **Add Files to Script**. The files are copied to the script's folder.
   - In the script's runtime settings, in **Java Environment > Classpath**, set the classpath to the Java classes. These files are then used as a reference, but are not contained in the script's folder.

6. Add script enhancements

   Enhance your script with rendezvous points, transactions, output messages, and so on. For more information, see "Enhance a Java Script" on page 555.

### See also:

- "Java Protocols" on page 549

# Java over HTTP Protocol

The Java over HTTP protocol is designed to record Java-based applications and applets. It produces a Java language script using web functions. This protocol is distinguished from other Java protocols in that it can record and replay Java remote calls over HTTP.

## Enable .jars for Recording with Java over HTTP

To record with Java over HTTP, you must specify which .jar files to use in order to deserialize the recorded data.

This topic describes how to locate the relevant .jar files and add them to the classpath.

## Java applets

If your application uses Java Applets, you need to find the relevant .jar files and enable them in the **Recording Options > Java Environment Settings > Classpath** node:

1. Clear the JAR cache by selecting **Control Panel > Java > General Tab > Temporary Internet Files > Settings > Delete Files**.
2. Open your application and perform a few business processes to repopulate the JAR cache with .jar files from your application. When you are finished, close your application.
3. Select **Control Panel > Java > General Tab > Temporary Internet Files > View**. This lists the JAR cache and should contain only the .jar files used by your application.
4. Download the files. Try the options below in the order in which they appear. When you succeed, proceed to the next step to add the .jar files to the classpath.
   a. Option 1: For each .jar file, go to the listed URL and download the file. If you cannot download one or more of the .jar files, continue with the next option.
   b. Option 2: Clear the cache again by selecting **Control Panel > Java > General Tab > Temporary Internet Files > Settings > Delete Files**. Open your application again and perform a few business processes. Do not close your application. Open the Java Console. There should be a message for each .jar file telling you the location it is stored in a temporary file on your computer. The files are usually hashed and don't have .jar extensions. Change the name (including changing each extension to .jar) and copy the file to a known location.
   c. Option 3: If the files don't show up in the Java console, locate the temporary folder as listed in **Control Panel > Java > General Tab > Temporary Internet Files > Settings > Location**. Open the specified location and rename all the files in the sub-folders to .jar. Do not rename all the files in the main folder.
5. Add the .jar files to the **Recording Options > Java Environment Settings > Classpath** node. For more information, see "Java > Classpath Recording Options" on page 201.

## Local Java applications

When the Java application (not applet) is installed on your local computer, all of the .jar files exist on your computer.

1. Locate the .jar files:
   - Look in the batch file that launched the application. All of the .jar files that are referenced must be added to the classpath.
   - If you cannot locate the batch file, add all of the .jar files from the application folder and sub-folders to the classpath.
2. Add the .jar files to the classpath in the **Recording Options > Java Environment Settings > Classpath** node.

### Handle custom serialization and encryption of serialized objects

If the application being recorded uses an application-specific serialization, VuGen is not able to handle serialized objects. This can happen, for example, if custom encryption is applied to serialized objects.

To enable recording of such applications:

1. Implement Java interface **SerializationProviderInterface**.
2. Add the .jar with the implementationin the **Recording Options > Java Environment Settings > Classpath** node.
3. Specify the implementation class name in **Recording Options -> Java VM -> Custom SerializationProviderInterface implementation**.

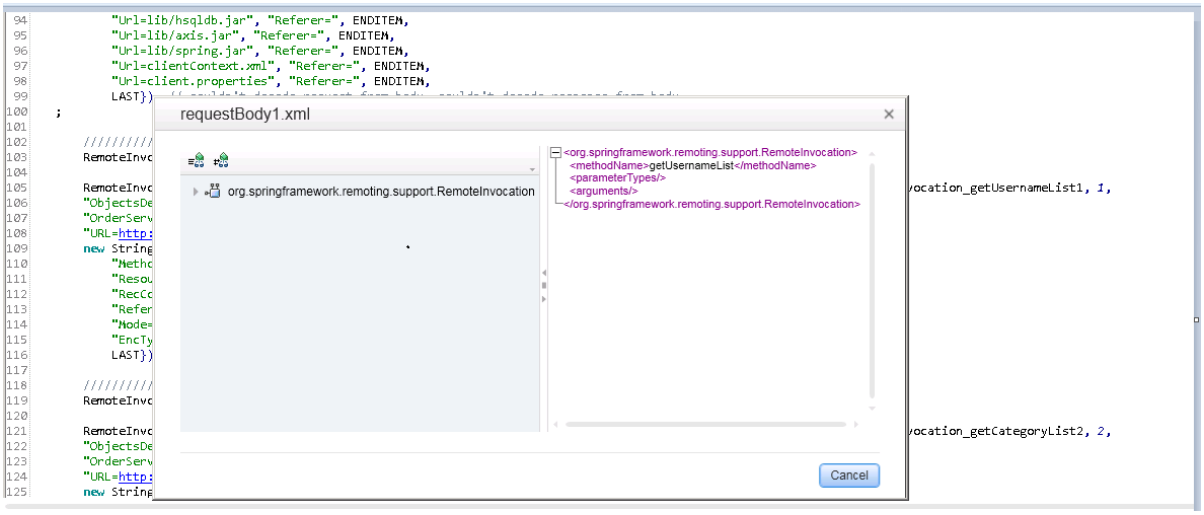
## Viewing Requests and Responses in XML Format

> **Note:** This topic applies to the Java over HTTP protocol only.

For each request and response, you can view the corresponding XML that represents the binary java object during the recording phase.

**To view requests and responses in XML format:**

1. Locate the target request or response section in the code. Right-click the commented **RequestBodyX.xml** or **ResponseBodyX.xml**.
2. Select **View XML**. The XML is displayed in a separate window.




## Notes, Limitations, and Troubleshooting - Java over HTTP

This section describes notes, limitations, and troubleshooting for the Java over HTTP protocol.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

## Notes

- JDK 1.5 or higher is required.
- Although .NET-based and Java protocols support creating threads, we recommend that you do not use background threads in real load testing scenarios because:
  - Threads can degrade tests scalability.
  - Threads can affect performance measurements.
  - The utility functions' behavior is undetermined if called from any thread except the Vuser main thread which runs the vuser_init, Action, and vuser_end actions. This applies to all functions named lr*.
- Java over HTTP supports asymmetric Java object traffic. This means that object serialization traffic is recognized even when it is on only one side of the communication. This occurs when the request is serialization and the response is plain HTTP, or vice versa.

## Limitations

- 64-bit replay is not supported for Java over HTTP.
- Snapshots are not supported for Java over HTTP.
- Due to a restriction in JVM architecture, a method cannot exceed 64 KB.
- You cannot use the GWT DFE extension or other Java-compatible DFE extensions. This is a result of a limitation of VuGen's mdrv process, which only allows you to instantiate a single JVM.
- Lazy evaluating objects are not supported, for example hibernate in lazy mode.
- If there are stateful serialization mechanisms on the application server, this can interfere with VuGen's deserialization and result in unserialized data and unexpected errors.
- The **Remote Application via LoadRunner Proxy > Display recording toolbar on client machine** option, is not supported for the Java over HTTP protocol. For details, see "Start Recording Dialog Box" on page 240.
- When you run a Java script, the replay status may be "Script Not Run" and some errors may appear in the mdrv.log file. However, due to Java internal architecture, these errors may not be included in the VuGen Output and Errors panes. This occurs when VuGen fails to initialize a Java Vuser, and JVM then terminates the replay process.

  **Workaround:** Look for errors directly in the mdrv.log file. If the entry in the log is due to a memory-related issue, try using different memory options for Java in the runtime settings.

## Troubleshooting

### Disable Exception Error Checking

If you are receiving exception errors and you are sure that the error is irrelevant, VuGen allows you to disable all such error messages. To do this, select **Replay > Runtime Settings > Java VM** node. In the **Additional VM Parameters** field, and append the following string to the end of the current entry:

```
-DvalidateServerResponse=false
```

Additionally, you can change the error checking behavior of a specific step by adding a closing argument to the **sendSerialized** function in script view. For more information, see the Function Reference.

### Cannot Correlate Private Object Members

When you need to correlate or parameterize data that is a private member of an object, you can use the **lrapi.lr2.fieldSetter** and **lrapi.lr2.fieldGetter** functions.

```
 RemoteInvocation RemoteInvocation2 = (RemoteInvocation) JavaHTTP.readObject
(RemoteInvocationBA0);
            RemoteInvocation.methodName="applyToSchool";
            Student student=RemoteInvocation.arguments[0];

            Map grades=lr2.fieldGetter(student,"grades");//grades is a private member
of Student
            grades.put("Math","95");
            lr2.fieldSetter(student,"super.name","Tom");
             //Student class inherits the name field from Person. name field is a
string
            lr2.fieldSetter(student,"super.ID","98764321");
             //Student class inherits the ID field from Person. ID field is an int
            RemoteInvocationResult RemoteInvocationResult_ArrayList2 =
(RemoteInvocationResult) JavaHTTP.sendSerialized(RemoteInvocation2, 2,
             "ObjectsDeserializerDefaultImpl",....
```

### See also:

- "Notes and Limitations - Java Vuser protocols" on page 566

# LDAP Protocol

## LDAP Protocol Overview

LDAP, the Lightweight Directory Access Protocol, is a protocol used to access a folder listing. The LDAP folder is composed of many LDAP entries. Each LDAP entry is a collection of attributes with a

name, called a distinguished name (DN). For more information about DN, see "Defining Distinguished Name Entries" on the next page.

LDAP folder entries are arranged in a hierarchical structure that reflects political, geographic, and/or organizational boundaries. Entries representing countries appear at the top of the tree. Below them are entries representing states or national organizations. Below them might be entries representing people, organizational units, printers, documents, or just about anything else.

VuGen records communication over LDAP servers. It creates a Vuser script, with functions that emulate your actions. This includes logging in and out of the server, adding and deleting entries, and querying an entry.

## LDAP Protocol Example Script

All LDAP functions come in pairs—one for global sessions and one where you can indicate a specific session. To apply the action to all sessions, use the version without the **ex** suffix. To apply the action to a specific session, use the version with the session identifier with the **ex** suffix. For example, **mldap_logon** logs on to the LDAP server globally, while **mldap_logon_ex** logs on to the LDAP server for a specific session.

In the following example, the user logs on to an LDAP server, ldap1. It adds an entry and then renames the OU attribute from Sales to Marketing.

```
Action()
{
// Logon to the LDAP server
mldap_logon("Login",
                "URL=ldap://johnsmith:tiger@ldap1:80",
                LAST);
// Add an entry for Sally R. Jones
mldap_add("LDAP Add",
          "DN=cn=Sally R. Jones,OU=Sales, DC=com",
          "Name=givenName", "Value=Sally", ENDITEM,
          "Name=initials", "Value=R", ENDITEM,
          "Name=sn", "Value=Jones", ENDITEM,
          "Name=objectClass", "Value=contact", ENDITEM,
          LAST);
// Rename Sally's OU to Marketing
mldap_rename("LDAP Rename",
             "DN=CN=Sally R. Jones,OU=Sales,DC=com",
             "NewDN=OU=Marketing",
             LAST);
// Logout from the LDAP server
```

```
mldap_logoff();
    return 0;
}
```

## Defining Distinguished Name Entries

The LDAP API references objects by its **distinguished name** (DN). A DN is a sequence of **relative distinguished names** (RDN) separated by commas.

An RDN is an attribute with an associated value in the form attribute=value. The attribute names are not case-sensitive. The following table lists the most common RDN attribute types.

| String | Attribute Type |
|---|---|
| **DC** | domainComponent |
| **CN** | commonName |
| **OU** | organizationalUnitName |
| **O** | organizationName |
| **STREET** | streetAddress |
| **L** | localityName |
| **ST** | stateOrProvinceName |
| **C** | countryName |
| **UID** | userid |

The following are examples of distinguished names:

```
DN=CN=John Smith,OU=Accounting,DC=Fabrikam,DC=COM
DN=CN=Tracy White,CN=admin,DC=corp,DC=Fabrikam,DC=COM
```

The following table lists reserved characters that cannot be used in an attribute value.

| Character | Description |
|---|---|
| | space or # character at the beginning of a string |
| | space character at the end of a string |
| , | comma |
| + | plus sign |

| | |
|---|---|
| " | double quote |
| \ | backslash |
| < | left angle bracket |
| > | right angle bracket |
| ; | semicolon |

To use a reserved character as part of an attribute value, you must precede it with an escape character, a backslash (\). If an attribute value contains other reserved characters, such as the equal sign (=) or non-UTF-8 characters, you must encode it in hexadecimal format—a backslash followed by two hex digits.

The following are examples of DNs that include escaped characters. The first example is an organizational unit name with an embedded comma; the second example is a value containing a carriage return.

```
DN=CN=Bitwise,OU=Docs\, Support,DC=Fabrikam,DC=COM
DN=CN=Before\0DAfter,OU=Test,DC=North America,DC=Fabrikam,DC=COM
```

## LDAP Connection Options

Using the **mldap_logon[_ex]** function, you control the way you login to the LDAP server.

When specifying the URL of the LDAP server, you specify how to connect and with what credentials.

When specifying the server's URL, use the following format:

```
ldap[s][username:[password]@][server[:port]]
```

The following table shows several examples of connections to LDAP servers.

| Syntax | Description |
|---|---|
| ldap://a:b@server.com:389 | Connects to the server (to 389 port) and then binds with username "a" , password "b" |
| ldap://:@server.com | Connects to server (to default unsecured port 389) then binds anonymously with a NULL username and password |
| ldaps://a:@server.com | Connects to server (to default secured port 636) and then binds with username "a", password "" |
| ldap://@server.com, ldap://server.com | Connects to server without binding |
| ldap://a:b@ | Binds with username "a", password "b, executing a bind on the existing session without reconnecting |

| Syntax | Description |
|---|---|
| ldap://:@ | Binds anonymously with a NULL username and password (executes bind on existing session without reconnecting) |

You can also specify LDAP modes or SSL certificates using the following optional arguments:

- **Mode.** The LDAP call mode: *Sync* or *Async*
- **Timeout.** The maximum time in seconds to search for the LDAP server
- **Version.** The version of the LDAP protocol version 1,2, or 3
- **SSLCertDir.** The path to the SSL certificates database file (cert8.db)
- **SSLKeysDir.** The path to the SSL keys database file (key3.db)
- **SSLKeyNickname.** The SSL key nickname in the keys database file
- **SSLKeyCertNickname.** The SSL key's certificate nickname in the certificates database file
- **SSLSecModule.** The path to the SSL security module file (secmod.db)
- **StartTLS.** Requires that the StartTLS extension's specific command must be issued in order to switch the connection to TLS (SSL) mode

For detailed information about these arguments, see the Function Reference (**Help > Function Reference**).

## Troubleshooting and Limitations - LDAP

This section describes troubleshooting and limitations for the LDAP protocol.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

- If an LDAP version 3 script fails during replay, modify the **mldpa_logon_ex** statement to specify the version number by adding "Version=3" after "URL=.."
- Address lists are only partially supported—the script only uses the first address in the list.
- When recording LDAP scripts, the binary parameter values for certain LDAP functions (such as **mldap_add** or **mldap_modify**) are not recorded. Recording of binary parameters is part of the protocol's extended functionality and is not supported by VuGen.
- LDAP Protocol can be recorded in parallel with the Windows Socket protocol in multi-protocol recording mode. This allows you to record additional network activity generated by the LDAP application. The multi-protocol recording may result in duplicate calls to the LDAP application. **Workaround:** Edit the script and manually remove the duplicate calls. Leave the Windows Socket calls that emulate the additional networking I/O.

# Mailing Service Protocols

## Mailing Service Protocols Overview

The Mailing Service protocols emulate users working with email clients, viewing and sending emails. The following mailing services are supported:

- Internet Messaging (IMAP). For details, see "IMAP Protocol Overview" below.
- MS Exchange (MAPI). For details, see "MAPI Protocol Overview" below.
- Post Office Protocol (POP3). For details, see "POP3 Protocol Overview" on the next page.
- Simple Mail Transfer Protocol (SMTP). For details, see "SMTP Protocol Overview" on page 597.

The mail protocols support both record and replay, with the exception of MAPI that supports only replay.

## IMAP Protocol Overview

IMAP Vuser script functions record the Internet Mail Application Protocol.

Each IMAP function begins with an **imap** prefix. For detailed syntax information on these functions, see the Function Reference (**Help > Function Reference**).

In the following example, the **imap_create** function creates several new mailboxes: Products, Solutions, and FAQs.

```
Actions()
{
    imap_logon("ImapLogon",
            "URL=imap://johnd:letmein@exchange.mycompany.com",
            LAST);
    imap_create("CreateMailboxes",
            "Mailbox=Products",
            "Mailbox=Solutions",
            "Mailbox=FAQs",
            LAST);
    imap_logout();
    return 1;
}
```

When recording a login step in which an IP address was specified, the script saves the IP address instead of the host name.

> **Note:** VuGen currently only supports the IMAP LOGIN authentication method, but not the AUTHENTICATE method.

## MAPI Protocol Overview

MAPI Vuser script functions generate activity to and from an MS Exchange server. Each MAPI function begins with a **mapi** prefix. You cannot record Vuser scripts using the MAPI protocol. For detailed syntax

information on these functions, see the Function Reference (**Help > Function Reference**).

> **Note:** To run MAPI scripts, you must define a mail profile on the machine running the script. For example, install Outlook Express or Microsoft Outlook, set it as the default mail client, and create a mail account.

In the following example, the **mapi_send_mail** function sends a sticky note through an MS Exchange server.

```
Actions()
{
    mapi_logon("Logon",
            "ProfileName=John Smith",
            "ProfilePass=Tiger",
            LAST);
    //Send a Sticky Note message
    mapi_send_mail("SendMail",
            "To=user1@techno.merc-int.com",
            "Cc=user0002t@techno.merc-int.com",
            "Subject=<GROUP>:<VUID> @ <DATE>",
            "Type=Ipm.StickyNote",
            "Body=Please update your profile today.",
            LAST);
    mapi_logout();
    return 1;
}
```

## POP3 Protocol Overview

POP3 Vuser script functions emulate actions using the Post Office Protocol, POP3. Each function begins with a **pop3** prefix. For detailed syntax information on these functions, see the Function Reference (**Help > Function Reference**).

In the following example, the **pop3_retrieve** function retrieves five messages from the POP3 server.

```
Actions()
{
pop3_logon("Login", "
        URL=pop3://user0004t:my_pwd@techno.merc-int.com",
        LAST);
// List all messages on the server and receive that value
totalMessages = pop3_list("POP3", LAST);
// Display the received value (It is also displayed by the pop3_list function)
lr_log_message("There are %d messages.\r\n\r\n", totalMessages);
// Retrieve 5 messages on the server without deleting them
pop3_retrieve("POP3", "RetrieveList=1:5", "DeleteMail=false", LAST);
```

```
pop3_logoff();
    return 1;
}
```

**Note:** When recording a login step in which an IP address was specified, the script saves the IP address instead of the host name.

## SMTP Protocol Overview

SMTP Vuser script functions emulate the Single Mail Transfer Protocol traffic. Each SMTP function begins with an **smtp** prefix. For detailed syntax information on these functions, see the Function Reference (**Help > Function Reference**).

In the following example, the **smtp_send_mail** function sends a mail message, through the SMTP mail server, techno.

```
Actions()
{
    smtp_logon("Logon",
               "URL=smtp://user0001t@techno.merc-int.com",
               "CommonName=Smtp Test User 0001",
               NULL);
    smtp_send_mail(    "SendMail",
                    "To=user0002t@merc-int.com",
                    "Subject=MIC Smtp: Sample Test",
                    "MAILOPTIONS",
                    "X-Priority: 3",
                    "X-MSMail-Priority: Medium",
                    "X-Mailer: Microsoft Outlook Express 5.50.400\r\n",
                    "X-MimeOLE: By Microsoft MimeOLE V5.50.00\r\n",
                    "MAILDATA",
                    "MessageText="
                        "Content-Type: text/plain;\r\n"
                        "\tcharset=\"iso-8859-1\"\r\n"
                        "Test,\r\n"
                        "MessageBlob=16384",
                NULL);
    smtp_logout();
    return 1;
}
```

**Note:** When recording a login step in which an IP address was specified, the script saves the IP address instead of the host name.

# Message Protocols

## MMS (Multimedia Messaging Service) Protocol Overview

**Caution:**

- From LoadRunner 12.53, only replay is supported for this protocol. Support for this protocol will be discontinued in future versions.
- To use MMS protocol, you must first manually register the following dlls, using the **regsvr32** command:
  - **regsvr32 MMSEngine.dll**
  - **regsvr32 MM1Client.dll**

The MMS protocol is useful for sending MMS messages between mobile devices.

MMS (Multimedia Messaging Service) is an extension of the SMS protocol. An MMS message typically includes a collection of attachments. MMS usually requires a third generation (3G) network.

To receive an MMS message, a mobile phone receives an MMS notification over SMS. The SMS message can be received over various SMS protocols such as SMPP, UCP, and CIMD2. The SMS message contains a unique path to the MMS message stored in the MMSC server's database and the mobile phone uses this path to download the message from the SMSC. The current version of VuGen supports the receiving of MMS notifications over the SMPP interface.

Multimedia Messaging Service Vuser scripts support the 1.0 and 1.1 versions of the MMS protocol, as defined by OMA (Open Mobile Alliance organization). Using MMS Vusers, you can send MMS messages to the MMSC server directly over the HTTP protocol.

## Run an MMS Scenario in the Controller

**Caution:**

- From LoadRunner 12.53, only replay is supported for this protocol. Support for this protocol will be discontinued in future versions.
- To use MMS protocol, you must first manually register the following dlls, using the **regsvr32** command:
  - **regsvr32 MMSEngine.dll**
  - **regsvr32 MM1Client.dll**

An MMS (Multimedia Messaging Service) scenario requires a command line setting.

To set the MMS command line setting:

1. From the Scenario Schedule screen, click **Details**. The Group Information dialog is displayed.
2. If the Command line box is not visible, click the **More** button.
3. Add the following to the end of the Command line text: `-usingwininet yes`
4. Click **OK** to accept the Command line switch.

# Mobile Protocols

## Select a Recording Method for Mobile Applications

The VuGen mobile protocols expand VuGen's capability to record user activity on mobile applications, both native and browser-based. With these protocols you can:

- Simulate users working on mobile devices
- Create scripts based on the recordings of mobile devices or emulators

### Web HTTP/HTML

A protocol enabling you to develop scripts using mobile devices or mobile device emulators communicating with servers over HTTP. You can record network traffic into a capture file (PCAP file) and then use the PCAP file to create a Vuser script. Additionally, you can use a mobile emulator on your VuGen machine to develop your scripts. For more information, see "Web - HTTP/HTML - Recording Methods for Mobile Applications" on page 601.

| If your client application... | and... | You can use... |
|---|---|---|
| • Communicates with the HTML/HTTP protocol.<br>• Is either a browser-based or native application. | • Your device is in the same network as the VuGen machine.<br>• Your device allows proxy configuration. | "Record a Script via a Proxy" on page 235 |
| | • You have an existing capture file. | "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723 |
| | • You do not want (or cannot) record from the actual device.<br>• Your mobile OS is Android.<br>• You have a device emulator. | "Using Emulation to Record Mobile Applications" on page 716 |

### TruClient - Native Mobile

A protocol enabling you to record user activity in native, browser-based, or hybrid mobile applications using TruClient technology. The replay is performed on real devices, allowing you to obtain client-side

measurements and test end-to-end performance. This method requires the installation of Mobile Center. For details, select the relevant version in the Mobile Center Help and see the **Performance Testing** section.

Use if your client application is either:

- Native, browser-based
- Hybrid

### TruClient - Mobile Web

A protocol enabling you to record user activity in browser-based mobile applications using TruClient technology. The TruClient browser is modified to emulate the display of your mobile browser. For details, see"TruClient - Mobile Web Protocol" on page 602 and "How to Record a Script with TruClient - Mobile Web" on page 602.

Use if your client application is a browser-based mobile version of a Web site and supports Firefox.

### SMP (SAP Mobile Platform)

A protocol enabling you to create .NET based scripts using files generated by SMP. For details, see "SMP (SAP Mobile Platform) Protocol" on page 604.

Use if your client application is built on SMP (SAP Mobile Platform).

## Speed Simulation for Mobile Vuser Scripts

Speed Simulation models the behavior of a mobile network. This enables you to test applications, while taking into consideration end-to-end response time from device to server.

Three configuration options are available, allowing you to maximize the accuracy of your simulation:

- Maximum Bandwidth
- Standard Bandwidth
- Custom Bandwidth

You set the bandwidth using the Speed Simulation Runtime settings as described below.

| **To access** | **VuGen > Runtime settings > Network > Speed Simulation** |
|---|---|

User interface elements are described below:

| **UI Element** | **Description** |
|---|---|
| **Use maximum bandwidth** | Vusers run at the maximum bandwidth that is available over the network. This option is provided in cases where you do not wish to emulate a specific network. This is the default setting. |

| UI Element | Description |
|---|---|
| **Use standard bandwidth** | • General Packet Radio Service (GPRS)<br>• Enhanced Data rates for GSM Evolution (EDGE)<br>• Universal Mobile Telecommunications System (UMTS)<br>• High-Speed Downlink Packet Access (HSDPA)<br>• High-Speed Downlink Packet Access Phase 2 (HSDPA phase 2)<br>• High-Speed Uplink Packet Access (HSUPA)<br>Each network type has both a maximum and expected rate. The maximum rate represents the technology's best case performance rate while the expected rate more accurately reflects real time performance. |
| **Use custom bandwidth** | Sets a custom download and upload speed, defined in bits. You can set a single value or range for either the upload or download speed. A case where this option would be useful, is when you know the expected network speed from your cellular provider for a specific area.<br>**Download speed:**The download speed in bits, defined as a range or single value.<br>**Upload speed:**The upload speed in bits, defined as a range or single value.<br>**Note:**<br>• If you select this option and either the upload speed or download speed fields are left blank, the empty value is automatically set to the value of the non-empty field.<br>• If you select this option and leave the upload speed and download speed fields blank, the default setting of maximum bandwidth will be used. |

## Web - HTTP/HTML - Recording Methods for Mobile Applications

**Note:** This section describes the use of the Web - HTTP/HTML protocol for recording user activity on mobile applications. For comprehensive details about the Web - HTTP/HTML protocol, see "Web - HTTP/HTML Protocol" on page 701.

The Web HTTP/HTML protocol provides the following methods for generating Vuser scripts for mobile applications:

| You can use: | To do this: |
|---|---|
| **A Capture File to Generate a Vuser Script** | You can use VuGen to analyze a capture file that was created with an external capture file utility, such as Wireshark, and then generate a Vuser script.<br>For details see "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723. |
| **An Emulator to Create a Vuser Script** | For many mobile devices, there are third party emulators that you can install on your computer. Once installed, you can use the emulator to record and generate a Vuser script for mobile applications.<br>For details see "Using Emulation to Record Mobile Applications" on page 716. |
| **The LoadRunner Proxy to Create a Vuser Script** | The VuGen machine acts as a proxy server capturing all the traffic from the mobile device to the target server. After the business process has been recorded VuGen creates a script.<br>For details, see:<br>• "Recording via a Proxy - Overview" on page 233<br>• "Record a Script via a Proxy" on page 235 |

# TruClient - Mobile Web Protocol

## TruClient - Mobile Web Protocol Overview

Based on the innovative TruClient technology, TruClient - Mobile Web enables you to test web applications designed for mobile devices.

With this protocol you can:

- Simulate various mobile browsers.
- Develop scripts that are recorded on the user level making them clear and easily maintained.

The following illustrates the workflow for using the TruClient - Mobile Web protocol:

## How to Record a Script with TruClient - Mobile Web

1. Create a new script of the type **TruClient - Mobile Web**.
2. Add or import a mobile device.

   Select **Tools > Mobile TruClient Device Manager > Add Mobile Device** or **Import Mobile Device**.

For details, see "Mobile Device Dialog Box" below.

3. Start developing the script.

   Click the

button. In the Mobile Settings dialog box, select a device. If you need to modify a setting for this recording, do so now.

   For details, see "Mobile Device Dialog Box" below.

4. Record a business process.

   For details on using TruClient's functionality, see the TruClient Help Center (select the relevant version).

## Add, Remove, and Import Mobile Device Settings for TruClient - Mobile Web

### Create a Custom Device Using the Mobile Device Manager

The TruClient - Mobile Web device manager is delivered with the settings for many popular mobile devices, however, you can easily add a custom device.

1. Select **Tools > Mobile TruClient Device Manger > Add Mobile Device**. This opens the Add Mobile Device dialog box.
2. Enter the name of the device you would like to add or select an existing device from the drop-down list to customize.
3. Specify the User Agent.
4. Specify the Display. For details, see "Mobile Device Dialog Box" below.
5. Click **Add**.

### Remove a Mobile Device

1. Select **Tools > Mobile TruClient Device Manger > Remove Mobile Device**. This opens the Remove Mobile Device dialog box.
2. Select the device from the **Mobile Device** dropdown list and click **Remove**.

### How to Import a Mobile Device Settings to Your Script

The import feature can be used to import mobile devices created in other users' scripts.

1. Open a script that contains custom device settings.
2. Select **Tools > Mobile TruClient Device Manger > Import Mobile Device**. This opens the Import Mobile Device dialog box.
3. Select the **Import Device Settings**. This opens the **Import Device Settings** dialog box.
4. Highlight the custom device and click **Import**.

## Mobile Device Dialog Box

The Mobile Device dialog boxes (Mobile Settings, Add, Remove, and Import) enable you to select, add, remove, and import mobile devices for use with the TruClient - Mobile Web Protocol.

| | |
|---|---|
| **To access** | Use one of the following:<br>• **Develop Script** button (opens the Mobile Settings dialog box)<br> <br>• **Tools > Mobile TruClient Device Manager > ...**<br>  • **Add Mobile Device**<br>  • **Remove Mobile Device**<br>  • **Import Mobile Device** |
| **Relevant tasks** | • "Add, Remove, and Import Mobile Device Settings for TruClient - Mobile Web" on the previous page<br>• "How to Record a Script with TruClient - Mobile Web" on page 602 |

| UI Element | Description |
|---|---|
| **Mobile Device** | The mobile device under test.<br>If you added a new device, it will appear at the end of the dropdown list. |
| **User Agent** | The header string that is sent to server to identify your mobile device. Once you have a selected a device, the default header value will appear. However, this header string can be modified. |
| **Display** | Specify the width and height of your mobile device screen. TruClient - Mobile Web will open a browser window according to the display settings. |

## SMP (SAP Mobile Platform) Protocol

The SMP (SAP Mobile Platform) protocol enables you to create and replay .NET based scripts using files that have been generated by SMP, formerly known as SUP. This task describes the steps to create an SMP script.

Although .NET-based and Java protocols support creating threads, we recommend that you do not use background threads in real load testing scenarios because:

- Threads can degrade tests scalability.
- Threads can affect performance measurements.
- The utility functions' behavior is undetermined if called from any thread except the Vuser main thread which runs the vuser_init, Action, and vuser_end actions. This applies to all functions named lr*.

**Prerequisites**

To create an SMP script you will need to record a business process with SMP, a platform that is provided by Sybase, an SAP company. The recording generates the following files:

- Action.cs
- Vuser_init.cs
- Vuser_end.cs

**Create an SMP (SAP mobile Platform) script**

1. Select **New Script and Solution > SMP (SAP Mobile Platform)**.
2. Copy the generated .cs files into the script folder.

   > **Note:** The recording mechanism for SMP scripts is disabled.

3. Save, close and reopen the script.
4. Add the location of the **SAP.Mobile.LoadRunner.dll** file using the **Replay > Runtime Settings > .NET> Shared DLLs** view.

   > **Note:** The SAP.Mobile.LoadRunner.dll has been developed and is maintained by Sybase, an SAP company.

5. Generated .cs files can include objects from external .dlls. To successfully replay the script, include a reference to these .dlls in **Runtime Settings > .NET > Shared DLLs**.
6. Replay the script. For details, see "Debugging .NET Vuser Scripts" on page 618.

# MQTT Protocol

MQTT is a simple, lightweight, publish/subscribe messaging protocol designed for constrained devices and low-bandwidth, high-latency or unreliable networks. MQTT is ideal for machine-to-machine (M2M) and Internet of Things (IoT) communications, as well as for mobile applications where bandwidth and battery power are at a premium. MQTT focuses on minimizing network bandwidth and device resource requirements, while attempting to ensure reliability and some degree of assurance of delivery.

## Create an MQTT script

1. Create a new script by selecting **New Script and Solution > MQTT protocol**.

   By default, a ready-to-use script opens using all of the available MQTT functions, as defined in the Function Reference.

   For general details on creating scripts, see "Creating or Opening Vuser Scripts" on page 136 and "Vuser Script Sections" on page 155.

2. Modify the ready-to-use script template as needed. For a complete description of each of the function calls, see the Function Reference.

If you are creating a multiple-protocol script, you can integrate MQTT steps into a recorded Web - HTTP/HTML script. For details, see "Working with multi-protocol scripts" below.

3. Configure the runtime settings. These affect the way that a Vuser script runs.
   a. In the Solution Explorer, double-click **Runtime Settings.**
   b. In the <test name>:Runtime Settings view, configure the runtime settings as needed. To view descriptions of the individual runtime settings, hold your cursor over a runtime setting field name.

   For general details, see "Runtime Settings" on page 301 and "Configuring Runtime Settings" on page 301.

## Working with multi-protocol scripts

You can test scenarios in which your script uses both of the following:

- MQTT to communicate between the client and the broker
- Web - HTTP/HTML to communicate between the broker and the web application

You prepare your script by recording the steps on your web application. Then you manually add MQTT steps as needed. The pre-populated MQTT script is not included when working with a multiple protocol script, but you can copy/paste from the ready-to-use MQTT script template.

**To enable support:**

1. Create a multiple-protocol script that includes MQTT and Web - HTTP/HTML. For details, see "Create a New Script Dialog Box" on page 150.
2. Record a web script.

   > **Note:** Recording a web script overwrites **globals.h**, removing the default support for MQTT steps.

3. Restore support for MQTT steps:
   a. Open **globals.h** (located in the Solution Explorer under Extra Files).
   b. Add **#include "MqttApi.h"** to the Include Files section.
4. Add MQTT steps to the script.

**Sample multi-protocol script**

```
Action()
{
  MQTT client;

  client = mqtt_create();

  mqtt_set_client_id(client, "myclientid");
```

```
  lr_start_transaction("connect_MQTT");

  mqtt_connect(client, "tcp://mymachine:1883");

  lr_end_transaction("connect_MQTT", LR_AUTO);

  lr_start_transaction("navigate_Web");

  // your recorded web code here

  lr_end_transaction("navigate_Web", LR_AUTO);

  mqtt_publish(client, "topic/subtopic", "payload", MQTT_AUTO, MQTT_DEFAULT, MQTT_
NORETAIN);

  mqtt_disconnect(client);


  return 0;
}
```

## Ready-to-use MQTT script template

The default, single-protocol MQTT script includes actions that are pre-populated with commented out steps (shown below).

When working with multiple-protocol scripts, you first record steps on your web application. Then you add MQTT steps as needed. You can copy/paste the relevant steps from the template below.

For details on any of these functions, see the Function Reference.

### vuser_init

```
vuser_init()
{
  /* Declared in "globals.h" */
  client = mqtt_create();

  /* Optional connection settings */
  // mqtt_set_client_id(client, "<insert Client ID>");
  // mqtt_set_credentials(client, "<username placeholder>", "<password
placeholder>");
  // mqtt_set_lwt(client, "<topic>", "<sample lwt payload>", MQTT_AUTO, MQTT_DEFAULT,
MQTT_RETAIN);


  /* Optional SSL/TLS settings, applicable for SSL/TLS connections only */
```

```
  // mqtt_set_tls_certificate(client, "<path to a certificate file, e.g. cert.pem>",
"<path to a private key file, e.g. key.pem>", "<password for the private key>");
  // mqtt_set_tls_parameters(client, MQTT_TLS_DEFAULT, MQTT_DEFAULT_CIPHERS);

  /* Connect to an MQTT broker */
  mqtt_connect(client, "tcp://<enter an MQTT broker host name>");

  /* Uncomment the following if implementing the "subscriber" logic */
  // mqtt_subscribe(client, "<topic>");
  return 0;
}
```

## Action

```
Action()
{
  /* Uncomment the following if implementing the "subscriber" logic. Make sure to
also uncomment subscribe/unsubscribe in vuser_init/end.c */
  //
  /* Option 1: Wait for messages, then handle their contents */
  // size_t messageCount = mqtt_await_messages(client, MQTT_DEFAULT);
  // size_t i = 0;
  // for ( ; i < messageCount; i++)

  //
{
  // MQTT_MESSAGE m = mqtt_read_inbox(client);
  // const char* p = mqtt_get_payload(m);
  // const char* t = mqtt_get_topic(m);
  // size_t l = mqtt_get_length(m);
  // ... do something meaningful with the message
  // ... for instance, print received message information (assuming its payload is
not binary data)
  // lr_message("received message with size %d from %s", l, t);
  // lr_message("payload %.*s", l, p);
  //
  // mqtt_free_message(m);
  // }

  /* Option 2: Wait for messages, then clear Inbox (if their handling is not needed)
*/
  // mqtt_await_messages(client, MQTT_DEFAULT);
  // mqtt_clear_inbox(client);

  /* Uncomment the following if implementing the "publisher" logic */
  // mqtt_publish(client, "<topic>", "<payload>", MQTT_AUTO, MQTT_DEFAULT, MQTT_
RETAIN);
```

```
  return 0;
}
```

### vuser_end

```
vuser_end()
{
  /* Uncomment the following if implementing the "subscriber" logic */
  // mqtt_unsubscribe(client, "<topic>");

  /* Disconnect from the MQTT broker */
  mqtt_disconnect(client);

  return 0;
}
```

# .NET Protocol

## .NET Protocol Overview

Microsoft .NET Framework provides a foundation for developers to build various types of applications such as ASP.NET, Windows Forms, Web Services, distributed applications, and applications that combine several of these models.

VuGen supports .NET as an application level protocol. VuGen allows you to create Vuser scripts that emulate users of Microsoft .NET client applications created in its .NET Framework. VuGen records all of the client actions through methods and classes, and creates Vuser scripts in C Sharp or VB .NET.

By default, the VuGen environment is configured for .NET Remoting, ADO.NET, Enterprise Services, and WCF (Windows Communication Foundation) applications. Contact Customer Support for information on how to configure VuGen to record applications created with other client-server activity.

Although .NET-based and Java protocols support creating threads, we recommend that you do not use background threads in real load testing scenarios because:

- Threads can degrade tests scalability.
- Threads can affect performance measurements.
- The utility functions' behavior is undetermined if called from any thread except the Vuser main thread which runs the vuser_init, Action, and vuser_end actions. This applies to all functions named lr*.

## Considerations for Working with the .NET Protocol

The .NET protocol enables you to load test by replaying the application's method calls.

You can write a load test script manually, or you can generate a load test script by recording a business process.

Unlike other transport based protocols, the .NET protocol records the application method calls that are specified in the filter. Method calls that are not defined in the filter are not included in the generated script during the recording of the application.

Typically, a user is able to generate a script that can be used for load test using the default environment filter. However, for certain complex applications it may be difficult to generate a working script because the wrong method has been specified in the filter. The most difficult task of creating a load test script with the .NET protocol is resolving recording or code generation errors.

The following requirements will facilitate your ability to define the correct filter for the recording process and generate a working load test script:

- You should be familiar with .NET framework.
- You should be able to code using C# or VB.NET.
- You should be familiar with XML.
- A supported version of Visual Studio must be installed.

  For details, on supported versions, see the System Requirements.
- You should have an understanding of the architecture and communication techniques of the application so as to determine what functions or classes are relevant for the load test script.

The following can streamline the process of creating the correct filter:

You should have access to the application code or have some .NET reflector tools to enable you to view the decompiled code.

You should have access to the developers of the application who can help you identify the methods that are required for the load test.

For more information about .NET and the above environments, see the MSDN Web site.

# Viewing Data Sets and Grids

When you record a method returning a dataset, data table, or data reader action, VuGen generates a grid for displaying the data.

When working with a data reader, VuGen collects the data retrieved from each **Read** operation and converts it to the replay helper function, **DoDataRead**.

For example, after recording the following application code,

```
SqlDataReader reader = command.ExecuteReader();
```

```
    while( reader.Read() )
      {
          // read the values, e.g., get the string located in column 1
          string str = reader.GetString(1)
      }
```

VuGen generates the following lines in the script:

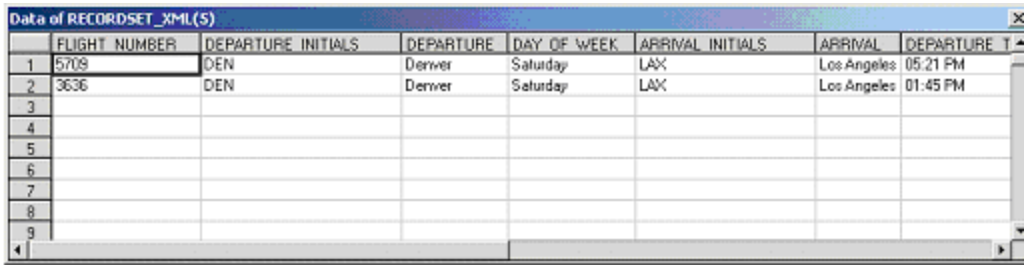
```
SqlDataReader_1 = SqlCommand_1.ExecuteReader();
LrReplayUtils.DoDataRead(SqlDataReader_1, out valueTable_1, true, 27);
```

where the two parameters indicate that during recording, the Application read all 27 available records. Therefore, during replay the script will read all available records.

In addition, VuGen generates a data grid containing all the information retrieved by the **Read** operations.

During replay you can use the output data table, containing the actual retrieved values, for correlation and verification. For more information regarding the **DoDataRead** function, see the Function Reference (**Help > Function Reference**).

When applicable, VuGen displays grid steps in the Step Navigator, and displays the associated grids in the Snapshot pane.




The dataset is stored in an XML file. You can view this XML file in the script's data/datasets folder. The data files are represented by an <*index_name*>.xml file, such as 20.xml. Since one file may contain several data tables, see the file **datasets.grd**, which maps the script index to the file index to determine which XML contains the data.

## Recording WCF Duplex Communication

WCF (Windows Communication Foundation) is a programming model that unifies Web Services, .NET Remoting, Distributed Transactions, and Message Queues into a single Service-oriented programming model for distributed computing.

WCF creates a proxy object to provide data for the service. It also marshals the data returned by the service into the form expected by the caller.

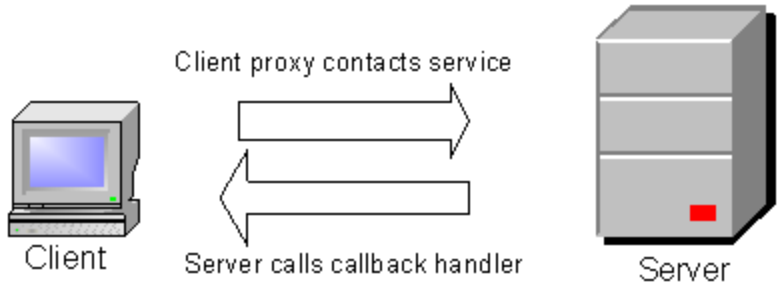
In addition to general support for the WCF environment, VuGen provides specialized support for applications that use WCF's duplex communication. In duplex communication, the client proxy contacts

the service, and the service invokes the callback handler on the client machine. The callback handler implements a callback interface defined by the server. The server does not have to respond in a synchronous manner—it independently determines when to respond and invoke the callback handler.

### Communication Between Client and Server

The communication between the client and server is as follows:

- The server defines the service contract and an interface for the callback.
- The client implements the callback interface defined by the server.
- The server calls the callback handler in the client whenever needed.




When trying to record and replay duplex communication, you may encounter problems when the script calls the original callback methods. By default, the callback handlers are not included in the filter. You could customize the filter to include those callback handlers. However, the standard playback would be ineffective for a load test, since many of the callbacks are local operations such as GUI updates. For an effective load test you cannot replay the original callback method invoked by the server.

VuGen's solution is based on replacing the original callback handler with a dummy implementation. This implementation performs a typical set of actions that you can customize further for your application.

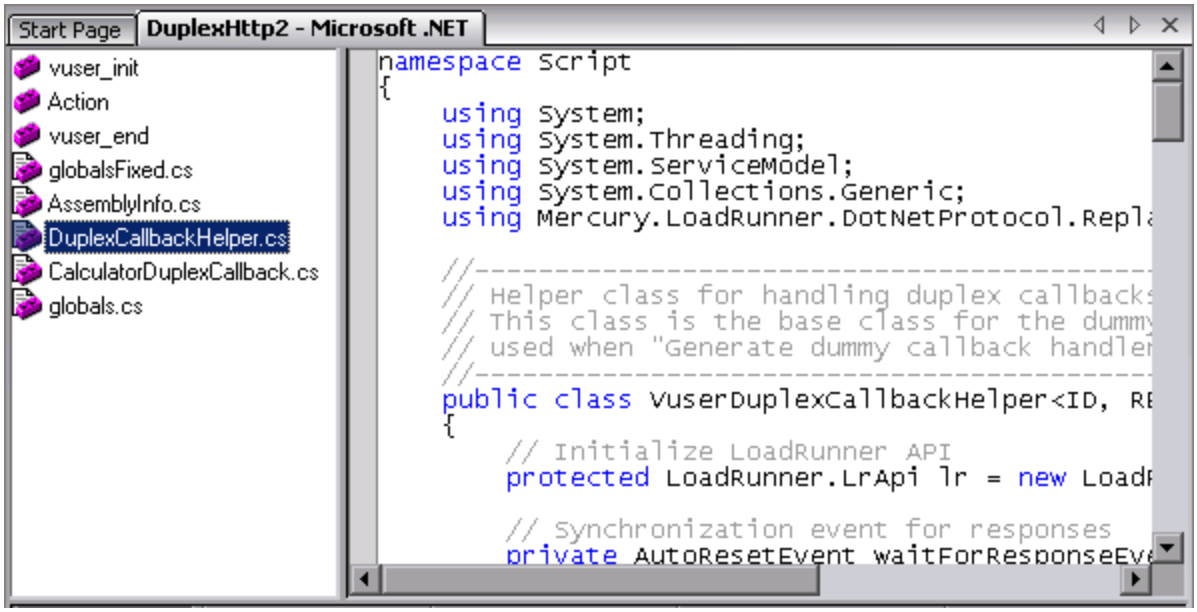
You instruct VuGen to replace the original callbacks by activating the **Generate Dummy Callback Handler** recording option. For more information, see "Microsoft .NET > Recording - Recording Options" on page 201.

### VuGen Implementation of a Duplex Callback

As part of the duplex communication solution, VuGen generates two support files:

- `DuplexCallbackHelper.<language>`
- *Callback_Name*`.<language>`

The following example shows the generated files for a Calculator application using duplex communication:




The Helper file serves as a general template for working with duplex callback handlers. It serves as a base class for the implementation of the callback.

The second file, **Callback_Name**, contains the implementation of the callback. The name of the callback implementation class is **Vuser**<*xxxx*> where *xxxx* is the name of the callback interface and it inherits from the **VuserDuplexCallbackHelper** class defined in the Helper file. VuGen creates separate implementation files for each interface.

This file performs two primary tasks:

- **Set Response.** It stores the data that came from the server in a map. It stores them with sequential IDs facilitating their retrieval. This method is called from the implementation of the callback interface. The following sample code demonstrates the dummy implementation of a callback method named **Result**. The method's arguments are stored in the map as an object array.

```
public virtual void Result(string operation, double result) {
     // Add here your own callback implementation and set the response data
        SetResponse(responseIndex++, new object[] {
                     operation,
                     result});
}
```

- **Get Response.** Waits for the next response to arrive. The implementation of GetNextResponse retrieves the next response stored in the map using a sequential indexer, or waits until the next response arrives.

The script calls GetNextResponse at the point that the original callback handler was called during recording. At that point, the script prints a warning to wait for here for the next response, and indicating the original callback.

## Replacement of the Callback in the Script

When you enable the **Dummy Callback** option (enabled by default), VuGen replaces the original duplex callback handlers with dummy implementations. The dummy implementation is called Vuser <*Callback Name*>. At the point of the original callback handler, the script prints a warning indicating that it was replaced.

### Customizing the Dummy Implementation

You can modify the implementation file to reflect your environment. This section contains several suggested customizations.

#### Timeouts

The default timeout for which the callback waits for the next response is 60000 msec, or one minute. To use a specific timeout, replace the call to **GetNextResponse** with the overloading method which gets the timeout as an argument as shown below. This method is implemented in the callback implementation file <*Callback_Name*> listed in the left pane after the **DuplexCallbackHelper** file.

**Example:**
```
// Get the next response.
 // This method waits until receiving the response from the server
 // or when the specified timeout is exceeded.

public virtual object GetNextResponse(int millisecondsTimeout) {
return base.GetResponse(requestIndex++, millisecondsTimeout);
        }
```

To change the default threshold for all callbacks, modify the **DuplexCallbackHelper** file.

**Example:**
```
// Default timeout threshold while waiting for response
     protected int millisecondsTimeoutThreshold = 60000;
```

#### Key Identifier

Many applications assign key identifiers to the data, which connects the request and response to one another. This allows you to retrieve the data from the map using its ID instead of the built-in incremental index. To use a key identifier instead of the index, modify the file <*Callback_Name*> replacing the first base template parameter, **named ID**, with the type of your key identifier. For example, if your key identifier is a string you may change the first template argument from **int** to **string**:

```
public class VuserXXX : VuserDuplexCallbackHelper<string, object>
```

In addition, you may remove the implementation of GetNextResponse() and replace it with calls to GetResponse(ID) defined in the base class.

#### Return Values

By default, since VuGen supports *OneWay* communication, the implementation callback does not return any value or update an output parameter when it is invoked.

```
public virtual void Result(string operation, double result) {
   // Add your own callback implementation and set the response data
```

If your application requires that the callback return a value, insert your implementation at that point.

#### Get Response Order

In VuGen's implementation, a blocking method waits for each response. This reflects the order of events as they occurred during recording—the server responded with data. You can modify this behavior to execute without waiting for a response or to implement the blocking only after the completion of the business process.

#### Find Port

The **FindPort** method in the Helper file is a useful utility that can be used in a variety of implementations. The Helper class uses this method to find unique ports for running multiple instances of the script. You can utilize this utility method for other custom implementations.

#### Recording Server Hosted By Client Applications

If the communication in your system is a server hosted by a client, VuGen's default solution for duplex communication will not be effective. In server hosted by client environments, it is not true duplex communication since the client opens the service and does not communicate through the Framework. For example, in queuing, the client sends a message to the service and opens a response queue to gather the responses.

To emulate a server hosted by a client, use the pattern depicted in the above solution—replace the original response queues with dummy callbacks and perform synchronization as required. For more information, contact Software Support.

## Asynchronous Calls

When VuGen records asynchronous calls on remote objects, you can specify how the calls are handled in the "Microsoft .NET > Recording - Recording Options" on page 201. These options are particularly relevant for .NET Remoting and WCF environments.

You can configure VuGen to one of the following options:

- **Call original callbacks by default.** Uses the recorded application's original callback when generating and replaying the script. If the callback method is explicitly excluded by a filter, the callback will be excluded even if you enable this option. If your callbacks perform actions that are not directly related to the business process, such as updating the GUI, disable this option.
- **Generate asynchronous callbacks.** This option defines how VuGen will handle callbacks when the original callbacks are not recorded. This is relevant when the above option, **Call original callbacks** is

disabled or when the callbacks are explicitly excluded.

When you enable this option, it creates a dummy method which will be called during replay instead of the original callback. This dummy callback will be generated in the **callbacks.cs** section of the script.

When you disable this option, VuGen inserts a NULL value for the callback and records the events as they occur.

> **Note:** VuGen supports the *Async* and *Await* modifiers, so if your AUT (application under test) uses these modifiers, they will be included in your script.

To display the callback method, OnComplete1, you click on the **callback.cs** file in the left pane.

> **Note:** If you recorded a script with specific recording options, and you want to modify them, you do not need to re-record the script. Instead, use the **Record > Regenerate Script** option to recreate the script with the new settings.

## Recording Dual HTTP Bindings

If your application employs dual HTTP Binding, since HTTP is inherently not a duplex protocol, the framework uses a standard port to receive response data being passed to the callback. When you attempt to run multiple instances of your application, you may be unable to do so using the same port number. VuGen provides you with an option of replacing the original client base address's port number with a unique port.

When you enable the **Generate Unique Client Base Address** recording option, VuGen checks the type of communication used by the application. If it detects dual HTTP communication, **WSDualHttpBinding**, it runs the **FindPort** utility (provided in LrReplayUtils) in the Helper file and finds unique ports for each instance of the callback.

This option is enabled by default. It is only relevant when you enable the above option, **Generate dummy callback handler**.

When you enable this option, VuGen generates the following code in the script:

```
#warning:  Code Generation Warning
    // Override the original client base address with a unique port number
    DualProxyHelper.SetUniqueClientBaseAddress<XXXX>(YYYYY);
```

For more information, see "Microsoft .NET > Recording - Recording Options" on page 201.

## Connection Pooling

ADO.NET providers deploy a feature called **connection pooling** which can significantly influence load test accuracy. Whenever only one app domain is used for all Vusers, connection pooling is turned on—.NET Framework keeps the database connections open and tries to reuse them when a new

connection is requested. Since many Vusers are executed in the context of a single application domain, they may interfere with one another. Their behavior will not be linear and that may decrease their accuracy.

In the .NET runtime settings, the AppDomain Per Vuser property enables execution of each Vuser in a separate app domain (true by default). This means that there is connection pooling in the scope of each Vuser, but the Vusers will not interfere with one another. This setting provides more accuracy, but lower scalability.

If you disable this option, you need to manually disable connection pooling for the database.

The following table describes how to manually disable connection pooling:

| Provider | Option |
|---|---|
| **.NET Framework Data Provider for SQL Server** | "Pooling=false" or "Pooling=no" |
| **.NET Framework Data Provider for Oracle** | "Pooling=false" or "Pooling=no" |
| **.NET Framework Data Provider for ODBC** | Connection pooling is managed by an ODBC Driver Manager. To enable or disable connection pooling, use the ODBC Data Source Administrator (found in Control Panel or the Administrative Tools folder). The Connection Pooling tab allows you to specify connection pooling parameters for each of the installed ODBC drivers. |
| **.NET Framework Data Provider for OLE DB** | "OLE DB Services=-2" |
| **Oracle Data Provider for .NET** | "Pooling=false" |
| **Adaptive Server Enterprise ADO.NET Data Provider** | "Pooling=False" |

# Debugging .NET Vuser Scripts

You can compile a .NET Vuser script to check its syntax, without running the script. To compile the script directly from VuGen, press Shift+F5 or select **Vuser > Compile**. If VuGen detects a compilation error, it displays the error in the Output window. Double-click on the error to go to the problematic line in the script.

To run the script directly from VuGen, press F5 or select **Replay > Run**. Breakpoints and step-by-step replay are not supported in VuGen's editor window for .NET Vusers. To debug a script and run it with breakpoints or step-by-step, run it from within Visual Studio .NET as described below.

## Viewing Scripts in Visual Studio

Visual Studio provides you with additional tools to view, edit, and debug your Vuser scripts. You can add breakpoints, view variable values, add assembly references, and edit the script using Visual Studio's IntelliSense. You can also run the script in a step-by-step mode for debugging.

When you save your script, VuGen creates a Visual Studio solution file, **Script.sln**, in your script's folder. You can open the solution file in Visual Studio .NET and view all of its components in the Solution Explorer.

To open the Vuser script in Visual Studio, select **Design > Open Script in Visual Studio** or click the **Visual Studio** button on the VuGen toolbar.

> **Note:** By default, the Vuser script opens in the latest version of Visual Studio that is installed on your computer. For example, if you have both Visual Studio 2013 and 2015 installed, the script will be converted and opened in Visual Studio 2015.

Double-click the appropriate section in the Solution Explorer, such as **vuser_init.cs**, to view the contents of the script.

> **Note:** VuGen automatically loads all of the necessary references that were required during recording. You can add additional references for use during compilation and replay through the Solution Explorer. Select the **Reference** node and select **Add Reference** from the right-click menu.

Click on **globals.cs** or **globals.vb** in the Solution Explorer to view a list of the variables defined and used by your script.

# .NET Filters Overview

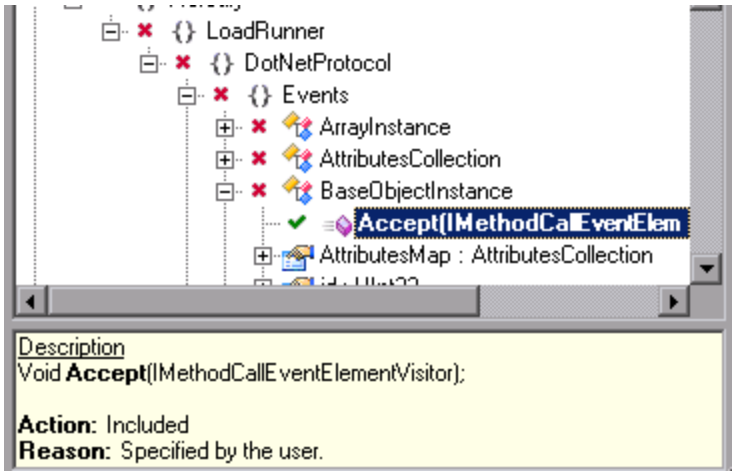
Recording filters indicate which assemblies, interfaces, namespaces, classes, or methods to include or exclude during the recording and script generation.

By default, VuGen provides built-in system filters for .NET Remoting, ADO.NET, Enterprise Services, and WCF (Windows Communication Foundation). These filters were designed to include the relevant interfaces for standard ADO.NET, Remoting, Enterprise Services, and WCF. VuGen also allows you to design custom filters.

Custom filters provide several benefits:

- **Remoting.** When working with .NET Remoting, it is important to include certain classes that allow you to record the arguments passed to the remote method.
- **Missing Objects.** If your recorded script did not record a specific object within your application, you can use a filter to include the missing interface, class or method.
- **Debugging.** If you receive an error, but you are unsure of it's origin, you can use filters to exclude methods, classes, or interfaces in order to pin-point the problematic operation.
- **Maintainability.** You can record script in higher level, make script more easy to maintain and to correlate.

The **.Net Recording Filter pane** lets you manipulate existing custom filters. It displays the assemblies, namespaces, classes, methods, and properties in a color-coded tree hierarchy.




The bottom pane provides a description of the assembly, namespace, class, method, property, or event. It also indicates whether or not it is included or excluded and a reason for the inclusion or exclusion.

### See also:

- ".NET Recording Filter Pane" on page 106

## Guidelines for Setting .NET Filters

When testing your .NET application, your goal is to determine how the server reacts to requests from the client. When load testing, you want to see how the server responds to a load of multiple users.

When recording a .NET application, your script may include calls to methods that do not affect the server, such as calls to a local utility or the GUI interface. These calls are usually not relevant to your testing goals, and it would be correct to filter them out.

The built-in filters, .NET Remoting, ADO.NET, Enterprise Services, and WCF, were designed to record only the server related traffic relevant to your testing goals. In some instances, however, you may need to customize filters to capture your .NET application's calls or exclude unnecessary calls. Using the ".NET Recording Filter Pane" on page 106, you can design custom filters to exclude the irrelevant calls and capture the server related calls.

Before creating a test, we recommend that you become familiar with your application and determine its primary classes and methods, so that you will know which ones to include in your recording.

If you are not familiar with your application's classes, you can use **Visual Studio** or a **Stack Trace** to help you determine which methods are being called by your application in order to include them in the filter. VuGen allows you to record with a stack trace that logs all of the methods that were called by your application.

Once you determine the required methods and classes, you include them using the .NET Recording Filter pane. When preparing a script, you may need to customize the filter several times in order to achieve the optimal filter. An optimal filter records the relevant methods without introducing a large number of irrelevant calls to the script.

> **Tip:** Strive to modify the filter so that your script will compile (Shift+F5) inside VuGen. Then customize the filter further to create a functional script that runs inside VuGen.

If you plan to add manual code to your script such as control flow or message statements, make sure to do so after you have a functional script that runs inside VuGen. The reason for this is that if you re-record a script or regenerate the script, you will lose all of the manual changes.

### Determining which Elements to Include or Exclude

When designing a custom filter, we recommend that you start by choosing the appropriate built-in filter as a base filter. You can then customize the filter using one of the following approaches:

- **Top Down Approach.** An approach in which you include the relevant namespace and exclude specific classes that are not part of the client-server activity. This is recommended if you are familiar with your application and you can identify a well-defined assembly which implements all client-server activity without involving any GUI elements, such as MyDataAccessLayer.dll.
- **Bottom up Approach.** An approach in which you use the default filter and refine it by adding individual methods or classes. Use this approach if you cannot identify a well-defined layer or if you are not familiar with your application. Do not add all AUT assemblies and then try to remove extra component one by one.

The following guidelines indicate when to include or exclude elements:

- If, as a result of your including a class, your script has many unrelated method calls, try modifying the filter to exclude the irrelevant methods.
- If you identify a non-client/server call in your script, exclude its method in the filter.
- During recording, VuGen may detect an unknown input argument, for example, an argument whose construction it had never encountered before. If this argument supports serialization, VuGen

**serializes** it by saving it to a file in a special format. During replay, VuGen reconstructs the argument by **deserializing** it.

- VuGen serializes objects passed as arguments that were not included by the filter. We recommend that you include this object in the filter in order to track its construction and activity instead of using it in its serialized form. You can identify serialized objects in the script by searching for calls to the **LrReplayUtils.GetSerializedObject** method or, in WCF environments, **LrReplayUtils.GetSerializedDataContract**. VuGen stores serialized objects in the script's **\data\SerializedObjects** folder as XML files with indexes: **Serialization_1.xml**, **Serialization_2.xml** and so forth.
- When no rules are specified for a method, it is excluded by default. However, when the remoting environment is enabled, all remote calls are included by default, even if they are not explicitly included. To change the default behavior, you can add a custom rule to exclude specific calls which are targeted to the remote server.
- Arguments passed in remoting calls whose types are not included by the filter, are handled by the serialization mechanism. To prevent the arguments from being serialized, you can explicitly include such types in order to record the construction and the activity of the arguments.
- Exclude all activity which involves GUI elements.
- Add assemblies for utilities that may be required for the script to be compiled.

For information on how to include and exclude elements, see ".NET Recording Filter Pane" on page 106.

> **Tip:** You can include or exclude a method directly from the VuGen editor. Select the method and choose **Open current method in Filter Pad** from the right-click menu. Once the method is selected in the filter tree, choose **Include** or **Exclude** from the right-click menu.

### Defining an Effective Filter

When preparing a script, you may need to customize the filter several times in order to achieve the optimal filter. An optimal filter records the relevant methods without introducing a large number of irrelevant calls to the script.

## To define an effective filter:

1. Create a new filter based on one of the built-in filters. If you know that the AUT (Application Under Test) does not use ADO.NET, Remoting, WCF, or Enterprise Services, clear that option since unnecessary filters may slow down the recording.
2. Set the **Stack Trace** option to true for both recording and code generation. Open the Recording Options ( ctrl+f7) and select the **Recording** node. Enable **Debug Options: Stack Trace** and **Code Generation: Show Stack Trace**.
3. Record your application. Click **Start Record** ( ctrl + r) to begin and Stop ( ctrl + f5) to end.
4. View the script's steps. If you can determine the business logic from the steps and apply correlation, you may not need to create custom filters. If, however, the script is very long or hard to maintain or correlate, you should customize the script's filter.

5. Try to identify the high-level method in the call that captures or wraps one or more client server calls. You can do this by opening the AUT source files (if they are available) in Visual Studio or by viewing a Stack Trace of the script.
6. Set the filter to include the relevant methods—you may need to add their assembly beforehand. For tips about including and excluding elements in the filter, see ".NET Filters Overview" on page 618.
7. Record the application again. You should always re-record the application after modifying the filter.
8. Repeat steps 4 through 7 until you get a simple script which can be maintained and correlated.
9. After creating an optimal script, turn off the **Stack Trace** options and regenerate the script. Open the Recording Options ( ctrl+f7) and select the **Recording** node. Disable **Debug Options: Stack Trace** and **Code Generation: Show Stack Trace**. This will improve the performance of subsequent recordings.
10. Correlate the script. For your test to run properly, you may need to insert a correlation to capture a value and use it at a later point in the script. For more information, see "Microsoft .NET Correlation" on page 281.

## .NET Filters - Advanced

The filter tree hierarchy only displays public classes and methods. It does not show non-public classes or delegates.

You can add classes or methods that are not public by manually entering them in the filter's definition file.

The filter definition files, **<filter_name>.xml** reside in the dat\DotnetFilters folder of your installation. The available Action properties for each element are: **Include**, **Exclude**, or **Totally Exclude**. For more information, see ".NET Recording Filter Pane" on page 106.

By default, when you exclude a **class**, the filter mechanism applies **Exclude**, excluding the class, but including activity generated by the excluded class. When you exclude a **method**, however, it applies **Totally Exclude**, excluding all referenced methods.

For example, suppose Function A calls function B. If Function A is **Excluded**, then when the service calls Function A, the script will include a call to Function B. However, if function A is **Totally Excluded**, the script will not include a call to Function B. Function B would only be recorded if called directly—not through Function A.

VuGen saves a backup copy of the filter as it was configured during the recording, **RecordingFilterFile.xml**, in the script's **data** folder. This is useful if you made changes to the filter since your last recording and you need to reconstruct the environment.

## Reference List Dialog Box [.NET Protocol]

This dialog box allows you to manage the list of all of the referenced DLLs for your script.

| | |
|---|---|
| **To access** | Click within the VuGen editor and select **References** from the right-click menu. |

User interface elements are described below:

| Column | Description |
|---|---|
| **Active** | Entries in the list are global for all .NET scripts. This check box allows you to indicate whether this file should be active for the current script. |

| | |
|---|---|
| **Shared** | Indicates whether or not the DLL is shared. |
| **Name** | The name of the DLL, usually containing its details, such as version number and token information (read-only).<br><br>**Note:** If you add a reference to a DLL for a .NET filter, VuGen adds it to the script's reference list only if the script accesses the DLL's methods during recording. |
| **Copy Local** | Indicates whether or not to copy the DLL locally. Clear this check box if the DLL loads the library by itself, or if it is available system wide, such as a DLL registered in GAC. |
| **Hint Path** | The HintPath of the project file, if specified (read-only). |
| **Is Specific Version** | Indicates whether or not the DLL is for a specific version. |

**Note:** If you make any changes, such as selecting **Copy Local**, removing a row, or disabling **Active** for a specific reference, they will not be effective until you regenerate the script (**Record > Regenerate Script)**.

### See also:

- .NET Recording Filter Pane

## Add Reference Dialog Box [.NET Protocol]

This dialog box enables you to add references to your .NET script or to the .NET filters.

| | |
|---|---|
| **To access** | To add a reference to your script:<br>1. In a .NET script editor, select **References** from the right-click menu.<br>2. In the References List window, click **Add Reference**.<br>To add a reference to a .NET filter:<br>1. Select **View > .NET Recording Filter** or click .<br>2. Select the **Custom filters** option in the left pane.<br>3. Click **New** to create a new filter. Select an option and click **OK**.<br>4. In the **.NET Recording Filter** pane, select a filter element.<br>5. Click **Add Reference**. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Component List>** | A list of .NET Framework components or assemblies in the Public Assemblies folder.<br>• To add one of the listed items, select it and click **Select**. You can select multiple components using Ctrl-click. The bottom pane shows the selected references.<br>• To add an assembly that is not in the list, click **Browse** and locate the reference on your file system or network. |
| **<Selected Component List>** | The list of selected components. The Type column indicates **.NET** for a component from the Public Assemblies folder and **File** for a component that was added by selecting **Browse**.<br>• To clear an item from the list, select it in the bottom pane and click **Remove**. |

### See also:

- .NET Recording Filter Pane

## Create a New Filter Dialog Box [.NET Protocol]

This dialog box enables you to create a new filter for .NET Vuser scripts.

| | |
|---|---|
| **To access** | Perform the following:<br>1. Select **View > .Net Recording Filter** or click the toolbar button.<br>2. Select the **Custom filter** option in the left pane.<br>3. Click **New**. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Start with an empty filter** | Create a new filter that is not based on a pre-existing filter. |
| **Based on an environment filter** | Create a new filter based on an environment filter. Use the check boxes next to the environment filters to indicate which environment filters to base the filter on. |
| **Based on a custom filter** | Create a new filter based on a custom filter. Use the drop-down menu to select the custom filter. |

### See also:

- .NET Recording Filter Pane

## Configure Application Security and Permissions

A Security Exception that occurs while recording an application is usually due to a lack of permissions—the recording machine does not have sufficient permissions to record the application. This is common where your application is not local, but on the Intranet or network.

To solve this problem, you need to allow the recording machine to access the application and the script with Full Trust.

One solution is to copy the application and save your script locally, since by default, users have Full Trust permissions to all local applications and folders.

An additional solution is to create new code groups that gives Full Trust to each application folder, and the script folder.

#### Grant Full Trust permissions to a Specific Folder (Visual Studio NOT installed)

1. From the command prompt, run the caspol.exe application.
2. Set the desired permission.

#### Grant Full Trust Permissions to a Specific Folder (Visual Studio installed)

1. Open the .NET Configuration settings. Select **Start > All Programs > Administrative Tools > Microsoft .NET Framework 2.0 Configuration**. The .NET Configuration window opens.
2. Expand the **Runtime Security Policy** node to show the Code Groups of the machine.
3. Select the **All_Code** node.
4. Select **Action > New**. The Create New Code Group dialog box opens.
5. Enter a name for a new Code Group for your application or script. Click **Next**.
6. Select the **URL** condition type. In the URL box, specify the full path of the application or script in the format `file://...` and click **Next**.
7. Select the **FullTrust** permission set. Click **Next**.
8. Click **Finish** in the Completing the Wizard dialog box. The configuration tool adds your Code Group to the list of existing groups.
9. Repeat the above procedure for all .NET applications that you plan to record.
10. Repeat the above procedure for the Vuser script folder.

> **Note:** Make sure that the script folder has **FullTrust** permissions on all Load Generator machines that are participating in the test (LoadRunner only).

## Troubleshooting and Limitations - .NET

This section describes troubleshooting and limitations for .NET Vuser scripts.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

## Replay Limitations

- Exceptions in .NET scripts generated by vts functions, may cause the replay to abort, even when the **Continue on error** Runtime setting is enabled.
  **Workaround:** Use `try` and `catch` statements for error handling of the vts functions.
- .NET scripts that contain UI objects can behave unexpectedly during replay in VuGen, the Controller, or on a load generator machine.

## Recording Limitations

The following limitations apply to the VuGen recording of a Microsoft .NET application:

- .NET Vuser scripts support only single-protocol recording in VuGen.
- Direct access to public fields is not supported—the AUT must access fields through methods or properties.
- VuGen does not record static fields in the applications—it only records methods within classes.
- Multi-threaded support is dependent on the client application. If the recorded application supports multi-threading, then the Vuser script will also support multi-threading.
- In certain cases, you may be unable to run multiple iterations without modifying the script. Objects that are already initialized from a previous iteration, cannot be reinitialized. Therefore, to run multiple iterations, make sure to close all of the open connections or remoting channels at the end of each iteration.
- Recording is not supported for Enterprise Services communication based on MSMQ and Enterprise Services hosted in IIS.
- The .NET version should be the same on the record and replay machines.
- VuGen partially supports the recording of WCF services hosted by the client application.
- Recording is not supported for Remoting calls using a custom proxy.
- Recording is not supported for **ExtendedProperties** property of ADO.NET objects, when using the default ADO.NET filter.
- Applications created with .NET Framework 1.1 which are not compatible with Framework 2.0, cannot be recorded. To check if your Framework 1.1 application is compatible, add the following XML tags to your application's .config file:

```
<configuration>
    <startup>
        <supportedRuntime version="v2.0.50727"/>
    </startup>
</configuration>
```

Invoke the application (without VuGen) and test its functionality. If the application works properly, VuGen can record it. Remove the above tags before recording the AUT with VuGen. For more information regarding this solution, see the MSDN Knowledge Base.

- Applications that use the .NET Remoting Framework and are executed in CLR 2 (.NET frameworks 2/3/3.5), might crash during recording. During a crash you will receive a message containing the strings Version=4.x.x.x, and "is not registered for activation".
  **Potential Workaround:** In the Microsoft .NET: Recording user interface under Support for previous.NET version, select **Emulate previous .NET versions in transport level**, and then record again.
- When the application under test retrieves a server-activated object by calling new RemoteObject(), VuGen generates a RemotingServices.Connect function.
- Applications using multiple processes or multiple application domains are only partially supported.
- Shared DLLs are configured through the runtime settings and using the filters mechanism.
- When McAfee antivirus is active, it may issue the following message when recording a .NET script: "The solution has been changed externally".
  **Workaround:** Add the **VuGen.exe** process to the Low-Risk processes in the McAfee antivirus **On-Access Scan Properties**.
- VuGen does not record a method that gets a private class as an argument. For example:

```
public class A
{
     private class P
     {

          …
     }
     private void MethodUsingPrivateClass(P p)
     {
          …
     }
}
```

- References added to a .NET script during recording, by means of the .NET Recording Filter, are not removed from the script by removing them in the References List dialog box.

### Script Editing Limitations

Disable FIPS if you want to open or debug a .NET Vuser script in Visual Studio 2015.

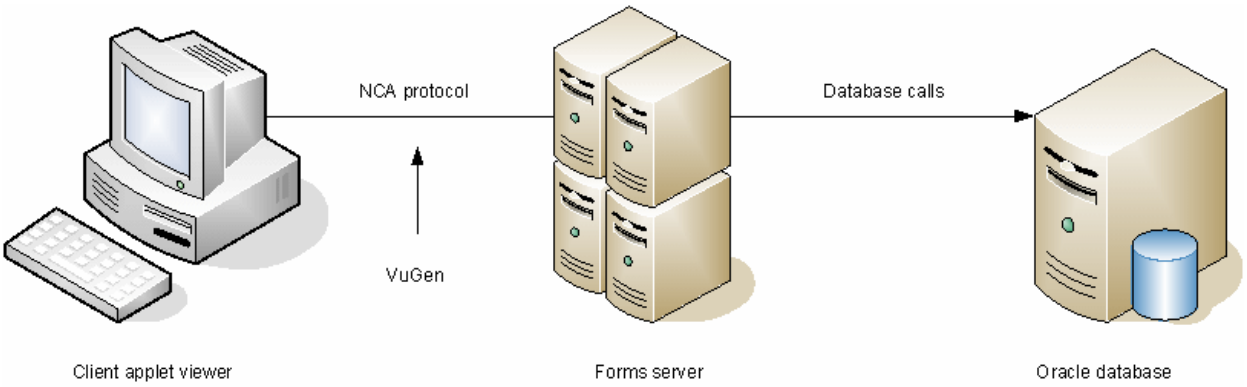
# Oracle NCA Protocol

## Oracle NCA Protocol Overview

Oracle NCA is a protocol that handles communication with the Oracle Forms server. Using your browser, you launch the database client, an applet viewer. You perform actions on the NCA database through its

applet viewer. This eliminates the need for client software and allows you to perform database actions from all platforms that support the applet viewer.

The NCA environment is a three-tier environment. The user first sends an http call from his browser to a Web server. This call accesses the startup HTML page which invokes the Oracle Applications applet. The client (applet viewer) communicates through the proprietary NCA protocol with the application server (Oracle Forms server) which then submits information to the database server.




VuGen records and replays the NCA communication between the client and the Forms server (application server).

The Oracle NCA protocol is commonly used as a multi-protocol in combination with Web - HTTP/HTML. This is the recommended way to record with Oracle NCA. If you are using Oracle NCA as a single protocol, web events are recorded but steps are not generated (or replayed) by default.

If you initially created a single protocol script for Oracle NCA, and at a later stage you require the Web functions for testing, you can regenerate your script in VuGen to add the Web functions, without having to re-record the session. You indicate this from the Protocols node in the Recording Options.

## Oracle NCA Protocol Example Scripts

In the following example, the user selected an item from a list (**nca_list_activate_item**), pressed a button (**nca_button_press**), retrieved a list value (**nca_lov_retrieve_items**), and performed a click in an edit field (**nca_edit_click**). The logical names of the objects are the parameters of these functions.

```
nca_lov_select_item("Responsibilities","General Ledger, Vision Operations");
nca_list_activate_item("FNDSCSGN.NAVIGATOR.LIST.0","+ Journals");
nca_list_activate_item("FNDSCSGN.NAVIGATOR.LIST.0"," Enter");
nca_button_press("GLXJEENT.TOOLBAR.LIST.0");
nca_lov_find_value("Batches","");
nca_lov_retrieve_items("Batches",1,9);
nca_lov_select_item("Batches","AR 1020 Receivables 2537: A 1020");
nca_edit_click("GLXJEENT.FOLDER_QF.BATCH_NAME.0");
```

In certain tests, such as those performed on Oracle Configurator applications, information returned by one function is required throughout the session. VuGen automatically saves the dynamic information to a parameter by inserting a **web_reg_save_param** function into the script.

In the following example, the connection information is saved to a parameter called NCAJServSessionID. The right boundary is **\r**. The actual right boundary may differ between systems.

```
web_reg_save_param ("NCAJServSessionId", "LB=\r\n\r\n", "RB=\r",
       LAST);
web_url("f60servlet",
         "URL=http://usscifforms05.sfb.na/servlet/f60servlet\?config=mult",
         LAST);
```

> **Note:** If **web_reg_save_param** parameters are generated automatically, we recommend to leave as they are. If you need to modify them, rather manually add a new **web_reg_save_param** function or add a new correlation rule.

## Oracle NCA Record and Replay Tips

When recording an Oracle NCA Vuser script, follow these guidelines:

- **Recommended:** Install Jinitiator before recording a script.
- Close all browsers before you begin recording.
- Record the login procedure in the **vuser_init** section. Record a typical business process in the Actions section. When you run the script, you can then specify multiple iterations for a specific business process. For more information, see "Solution Explorer Pane" on page 69.
- VuGen supports the recording of Oracle Forms applications using the Forms Listener Servlet in multi-protocol mode. The application server uses the **Forms Listener Servlet** to create a runtime process for each client. The runtime process, **Forms Server Runtime**, maintains a persistent connection with the client and sends information to and from the server.
- Use the **Oracle NCA > Code Generation** recording options to identify Oracle NCA Java objects that behave in an asynchronous way, such as **webutil** objects, and add them to an array in the header file.

  A **nca_java_set_area_class** step that calls the array is inserted into the script just before the **nca_connect_server** step. This replaces the need to insert a **nca_java_action** step for each of these object throughout the script, thereby allowing for a cleaner script.

  For more details, see "Oracle NCA > Code Generation Recording Options" on page 209
- To support Forms 4.5 in replay, modify the **mdrv_oracle_nca.dat** file in the **dat > mdrv** folder to match the following example:

```
[Oracle_NCA]
ExtPriorityType=protocol
WINNT_EXT_LIBS=ncarp110.dll
```

```
WIN95_EXT_LIBS=ncarp110.dll
LINUX_EXT_LIBS=liboranca.so
SOLARIS_EXT_LIBS=liboranca.so
HPUX_EXT_LIBS=liboranca.sl
AIX_EXT_LIBS=liboranca.so
LibCfgFunc=oracle_gui_configure
UtilityExt=lrun_api
```

To restore Forms support for versions later than 4.5, restore the original values.

## Pragma Mode

The client side of the Oracle NCA Vuser can be configured to send an additional header to the server named **Pragma**.

The header is a counter that behaves as follows:

The initial message of the NCA handshake has a value of 1.

The messages that follow the handshake are counted, beginning with 3. The counter is incremented by 1 for each message sent by the client.

If the message received from the server is of type `plain\text` and the body of the message begins with `ifError:#/#00`, the client sends a 0 byte message to the server and the Pragma value changes its sign to a minus. This sign changes back after the client succeeds in receiving the information from the server.

Recording of the Pragma header is only supported in the multi-protocol mode (Oracle NCA and Web). You can identify the Pragma mode within the script's default.cfg file. When operating in Pragma mode, the UseServletMode is set to 2.

```
[HttpConnectMode]
UseHttpConnectMode=1
RelativeURL=<NCAJServSessionId>
UseServletMode=2
```

For information on the Pragma related settings, see the **Oracle NCA > Client Emulation** view in the runtime settings.

To identify the Pragma mode, you can perform a WinSock level recording and check the buffer contents. In the first example, the buffer contains the Pragma values as a counter:

```
send buf108
    "POST /ss2servlet/oracle.forms.servlet.ListenerServlet?JServSessionIdss2ser"
    "vlet=gk5q79uqy1 HTTP/1.1\r\n"
  "Pragma: 1\r\n"
    ...
send buf110
    "POST /ss2servlet/oracle.forms.servlet.ListenerServlet?JServSessionIdss2ser"
```

```
    "vlet=gk5q79uqy1 HTTP/1.1\r\n"
  "Pragma: 3\r\n"
    ...
```

In the following example, the buffer contains the Pragma values as an error indicator:

```
recv buf129 281
    "HTTP/1.1 200 OK\r\n"
    "Date: Tue, 21 May 2002 00:03:48 GMT\r\n"
    "Server: Oracle HTTP Server Powered by Apache/1.3.19 (Unix) mod_fastcgi/2.2"
    ".10 mod_perl/1.25 mod_oprocmgr/1.0\r\n"
    "Content-Length: 13\r\n"
    "Content-Type: text/plain\r\n"
    "\r\n"
  "ifError:8/100"
send buf130
    "POST /ss2servlet/oracle.forms.servlet.ListenerServlet?JServSessionIdss2ser"
    "vlet=gk5q79uqy1 HTTP/1.1\r\n"
  "Pragma: -12\r\n"
    ...
```

## Enable the Recording of Objects by Name

When recording an Oracle NCA script, you must record the session using object names instead of the standard object ID. If the script is recorded using the object ID, replay may fail because the ID is generated dynamically by the server and may differ between iterations. You can verify that your script is being recorded with object names by examining the **nca_connect_server** function.

```
nca_connect_server("199.35.107.119","9002"/*version=11i*/,"module=/d1/oracle
        /visappl/fnd/11.5.0/forms/US/FNDSCSGN userid=APPLSYSPUB/PUB@VIS
        fndnam=apps record=names ");
```

If the **record=names** argument does not appear in the **nca_connect_server** function, you may be recording object IDs. You can instruct VuGen to record object names by modifying one of the following:

### Startup HTML file

If you have access to the startup HTML file, instruct VuGen to record object names instead of its object ID by setting the **record=names** flag in the startup file— that is, the file loaded when you start the Oracle NCA application:

1. In the startup file that is called when the applet viewer begins, locate the following line:

   ```
   <PARAM name="serverArgs ... fndnam=APPS">
   ```

2. Add the Oracle key, **record=names**:

```
<PARAM name="serverArgs ... fndnam=APPS record=names">
```

### Forms configuration file

If the application has a startup HTML file that references a Forms Web CGI configuration file, **formsweb.cfg** (a common reference), you may encounter problems if you add **record=names** to the Startup file. In this case, modify the configuration file. To enable the recording of object names using the configuration file:

1. Open the Forms Web CGI configuration file.
2. Define a new parameter in this file (see sample Web CGI configuration file below for this change).

```
serverApp=forecast
serverPort=9001
serverHost=easgdev1.dats.ml.com
connectMode=socket
archive=f60web.jar
archive_ie=f60all.cab
xrecord=names
```

3. Open the startup HTML file and locate:

```
PARAM NAME="serverArgs"
```

4. Add the variable name as an argument to the ServerArgs parameter, for example, **record=%xrecord%**:

```
<PARAM NAME="serverArgs" VALUE="module=%form% userid=%userid% %otherParams%
record=%xrecord%">
```

5. Alternatively, edit the **basejini.htm** file in the Oracle Forms installation folder. This file is the default HTML file for running a form on the web using JInitiator-style tags to include the Forms applet. In the basejinin.htm file add the following line to the parameter definitions:

```
<PARAM NAME="recordFileName" VALUE="%recordFileName%">
```

In the <EMBED> tag, add the following line:

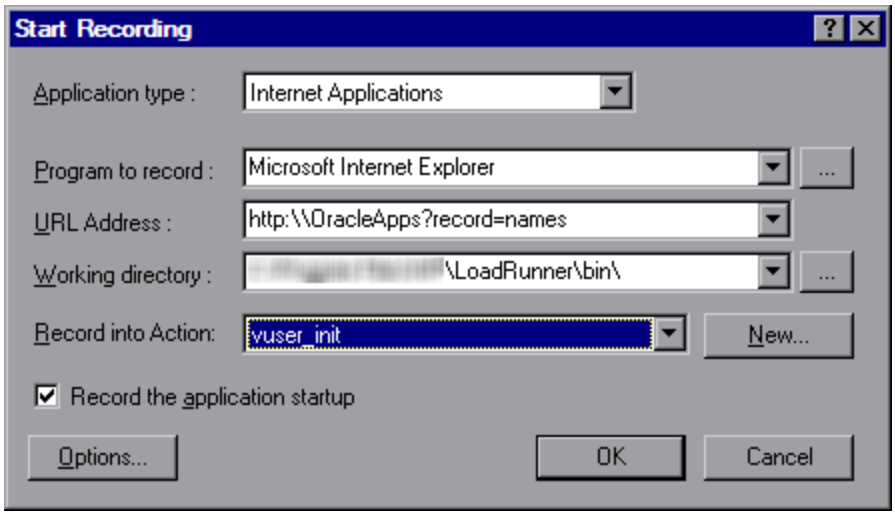
```
serverApp="%serverApp%"
logo="%logo%"
imageBase="%imageBase%"
formsMessageListener="%formsMessageListener%"
recordFileName="%recordFileName%"
```

The drawback in editing this file instead of the servlet configuration file **formsweb.cfg**, is that this file is replaced when you reinstall Oracle Forms. To avoid this, you can create a copy of the **basejini.htm** file and store it at another location. In the servlet configuration file, edit the **baseHTMLJinitiator** parameter to point to the new file.

### URL to record

If you do not have access to the startup HTML file, you can still have Oracle NCA record object names instead of its object ID by modifying the URL to record. The following solution only works if the startup HTML file does not reference another file while loading.

For this solution, you add "**?record=names**" after the URL in the Start Recording dialog box, after the URL name to record.




Alternatively, if you are using an application developed on Oracle Forms 10G or later, add **&otherparams=record=names&** to the URL.

## Launch Oracle Applications via the Personal Home Page

When launching Oracle Forms applications (versions 6i and higher) by logging in through the **Personal Home Page,** you must set several system profile options at the user level. It is desirable to pass such variables at the user level, and not at the site level, where it will affect all users. The following steps describe how to configure the "ICX: Forms Launcher" profile.

1. Sign on to the application and select the "System Administrator" responsibility.
2. Select **Profile/System** from the Navigator menu.
3. Within the **Find System Profile Values** form:
   a. Select the **Display > Site** option
   b. **Users** = <your user logon> for example, operations, mfg, and so on)
   c. **Enter Profile** =%ICX%Launch%

d. Click **Find**.

4. Update the User value to the **ICX:Forms Launcher** profile:
   - If no parameter has been passed to the URL, append the following string to the end of the URL of the user value: **?**`play==;record=names`
   - If a parameter has been passed to the URL, append the following string to the end of the URL of the user value: **=;**`play==;record=names`
5. Save the transaction.
6. Log out of the Oracle Forms session.
7. Log out of the Personal Home Page session.
8. Sign on again via the **Personal Home Page** using your username.

If you were unable to update the ICX: Forms Launcher profile option at the user level, open the **Application Developer** responsibility and select the **Updatable** option for the ICX_FORMS_LAUNCHER profile.

The first parameter passed to the URL must begin with a question mark (?). You pass all subsequent parameters with an ampersand (=;). In most cases, the URL already contains parameters, which you can identify by searching for a question mark.

## Oracle NCA - Troubleshooting and Limitations

This section describes troubleshooting and limitations for Oracle NCA and Oracle-Web protocol scripts.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

### General Limitations

- The **Remote Application via LoadRunner Proxy > Display recording toolbar on client machine** option, is not supported for the Oracle NCA or Oracle-Web protocols. For details, see "Start Recording Dialog Box" on page 240.
- Recording by proxy (local or remote) works only for servers which use HTTP/S communication. It does not work for servers that use only Socket communication.

### Testing Secure Oracle NCA Applications

- In the **Mapping and Filtering** node of the Recording Options dialog box, delete any existing Port mapping entries for port 443 and create a new Port mapping entry for the Oracle server name:

  **Service ID:** HTTP
  **Target Server:** Oracle Forms Server IP address or long host name
  **Target Port:** 443
  **Connection Type:** SSL
  **SSL Version:** Active SSL version. If in doubt, select SSL 2/3.

  Note that the **Service ID** is **HTTP** and not **NCA**.

For more information, see "Network > Mapping and Filtering Recording Options" on page 204.

- If you encounter problems when replaying an NCA HTTPS script during the **nca_connect_server** command, insert the following function at the beginning of the script.

```
web_set_sockets_option("SSL_VERSION","3");
```

## IE crashes when recording Oracle NCA or Oracle - Web scripts

This can occur due to an incompatible dll file.

Replace the incompatible dll

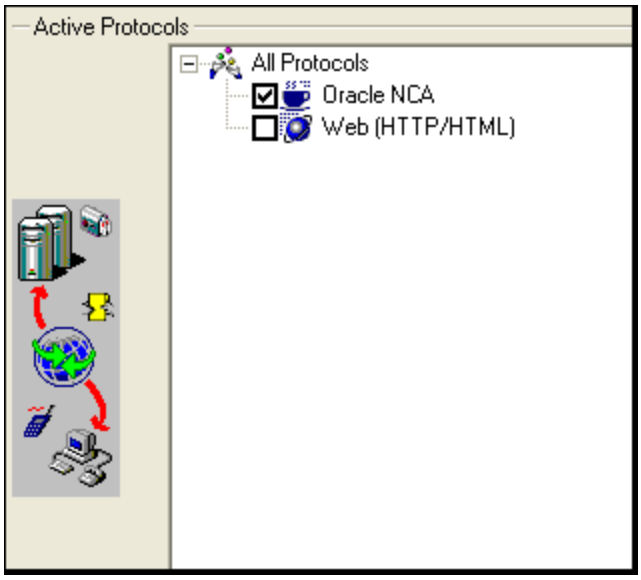
1. Open the **C:\program files\oracle\JInitiator_1.3.1.18\bin\hotspot** folder.
2. Back up the **jvm.dll** file.
3. Delete the **jvm.dll** file and replace it with a different version of the file.

## Testing Servlets and other Oracle NCA Applications

Certain NCA sessions use servlets:

- the Forms Listener servlet
- applications or modules that use both NCA and HTTP communications, such as the Oracle Configurator
- the initializing of the NCA application (downloading the applet, jar, and gif files)

When recording servlets, you must record both Oracle NCA and Web functions. You can do this by using the Oracle Apps IIi protocol or creating an Oracle NCA multi-protocol script. Alternatively, if you created a single protocol script for Oracle NCA, open the **General > Protocols** node in the Recording Options, and enable the Web protocol. Then you can begin recording.



If you are unsure whether your application uses servlets, you can check the **default.cfg** file in the script folder after recording a script. Locate the entry "**UseServletMode="**

If the value is 1 or 2, then servlets are being used and you must enable HTTP recording in addition to Oracle NCA.

If you already recorded a script, you can regenerate the code automatically to include the Web functions without having to re-record. Select **Record > Regenerate Script**, and select the Web protocol in the Protocols section.

## Determining the Recording Mode

When recording Oracle NCA scripts: VuGen automatically determines the correct connection mode: HTTP or Socket mode. Generally, you are not required to modify any of the recording settings as VuGen auto-detects the system configuration (unless you are working with Forms Server 4.5). In systems where the standard port mapping are reserved by other applications, you may need to modify the Port Mapping settings, depending on the recording mode.

You can determine the recording mode in one of the following ways:

- When using the NCA application, open the Java Console.

```
proxyHost=null
proxyPort=0
connectMode=HTTP
Forms Applet version is: 60812
```

  The **connectMode** entry indicates **HTTP**, **HTTPS**, or **socket**.

- After recording an NCA session, open the **default.cfg** file in the Vuser folder and check the value of the **UseHttpConnectMode** entry.

```
[HttpConnectMode]
UseHttpConnectMode= 2
// 0 = socket 1 = http 2 = https
```

When defining a new port mapping in the Server Entry dialog box, use a **Service ID** of HTTP for HTTP or HTTPS modes. For Socket mode, use a **Service ID** of NCA.

For more information about Port Mapping settings, see "Network > Mapping and Filtering Recording Options" on page 204.

## Recording Trace Information for Oracle DB

To debug your script, you can use the Oracle DB breakdown graphs. To gather data for this graph, you turn on the trace mechanism before running the script.

To manually turn on the tracing mechanism, use the **nca_set_custom_dbtrace** function. For more information, see the Function Reference (**Help > Function Reference**).

# RDP Protocol

## RDP Protocol - Overview

The Microsoft RDP (Remote Desktop Protocol) enables one computer [the client] to connect to another computer [the server] over a network connection. For example, you can use RDP to connect to a central, powerful server for working on specific business applications or graphic terminals. This provides you with the same look and feel as if you are working on a standalone PC. The client computer employs RDP client software for this purpose, while the other computer must run RDP server software. The client software is referred to as **Remote Desktop Connection**. The server software is referred to as **Remote Desktop Services**.

For details on the versions of RDP that are supported by VuGen, see the System Requirements.

> **Note:** RDP versions 5.1 and later have an **Experience** tab that allows you to set various options. This tab is not supported by VuGen recording. All options are set to the ON position.

## RDP Recording Tips

> **Note:** This topic applies to RDP Vuser scripts only.

When recording an RDP Vuser script, follow these guidelines in order to create an effective script.

### Single vs. Multi-Protocol Scripts

When creating a new script, you may create a single protocol or multi-protocol script. For example, to record both RDP traffic and Web responses, create a multi-protocol script for RDP and Web to enable the recording of both protocols.

### Record into Appropriate Sections

Record the connection process into the **vuser_init** section, and the closing process into the **vuser_end** section. This will prevent you from performing iterations on the connecting and disconnecting processes. For more information about recording into sections, see "Vuser Script Sections" on page 155.

### FIPS

Beginning with LoadRunner version 12.50, recording is supported for RDP scripts when FIPS enforcement is enabled on the machine.

### Run a Clean Session

When recording a session, make sure to perform the complete business process, starting with the connection and ending with the cleanup. End your session at a point from where you could start the

entire process from the beginning. Do not leave any client or application windows open.

You should also configure your terminal server to end disconnected sessions. Select **Administrative Tools > Terminal Services Configuration > Connection Properties > Sessions > Override User Settings** and set the server to end disconnected sessions.

## Explicit Mouse Clicks

When opening expanded menu options, click explicitly on each option—do not depend on the expanding menu. For example, when choosing **Start > All Programs > Microsoft Word**, be sure to click on the word **Programs**.

## Using Windows Logo key combinations

> **Note:** This tip applies only to Windows 8 installations on remote computers.

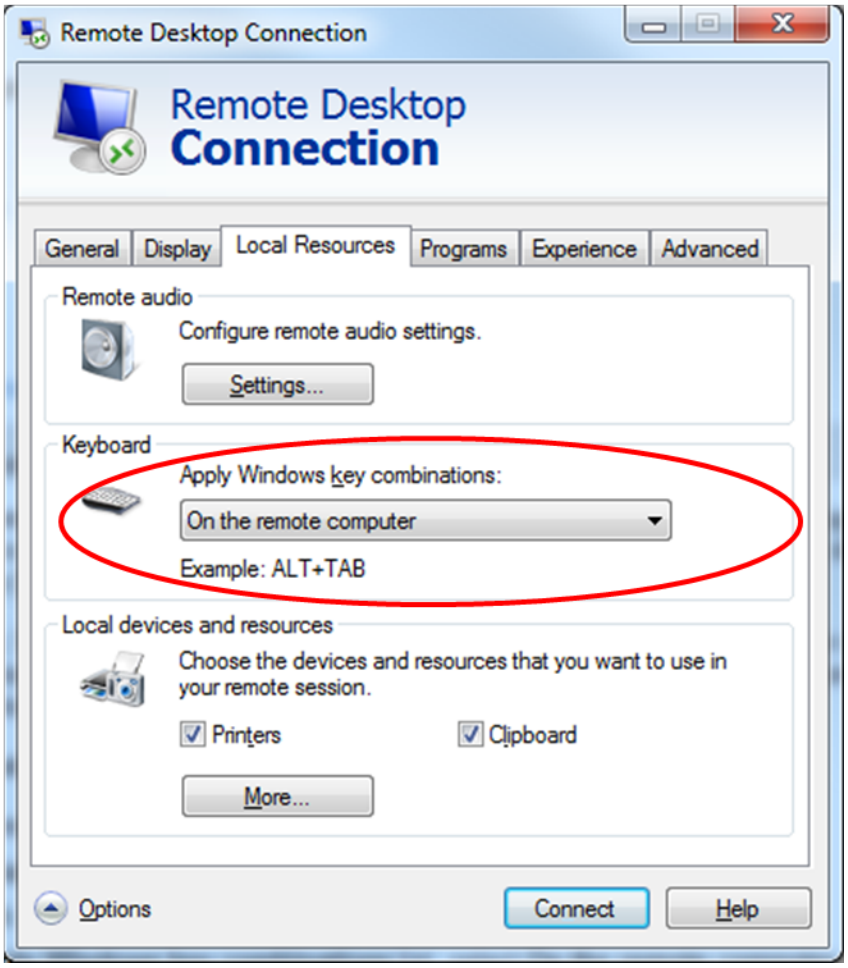
Because support for mouse movement in RDP Vusers can cause performance issues, by default, mouse movement support is disabled. Therefore, when you record an RDP Vuser script, it is recommended that you use the **Windows Logo** key combinations to display the Start screen [Windows Logo key], to show the Desktop [Windows logo key + D], and to open the charms bar [Windows Logo key + C].

When you run an RDP Vuser script, Windows key combinations can be applied on either the host computer or on the remote computer. To ensure that Windows key combinations are applied on the remote computer, when you record the connection to the remote computer, you must specify that Windows key combinations are applied on the remote computer.

How to apply Windows key combinations on the remote computer:

1. Open the Remote Desktop Connection dialog box.
2. Click **Options** to expand the dialog box.
3. Click the **Local Resources** tab.

4. Under **Keyboard**, from the **Apply Windows key combinations** list, select **On the remote computer**.




## Synchronizing using Windows 8 apps

> **Note:** This tip applies only to Windows 8 installations on remote computers.

Because many Windows 8 apps have dynamic user interfaces, avoid using these apps for image-based synchronization.

# RDP Replay Tips

Before replaying an RDP Vuser script, consider these guidelines:

> **Note:** This topic applies to RDP Vuser scripts only.

### Modify the RDP remote desktop resolution size for replay

Before replaying your script, you can modify the remote desktop resolution size using the **RDP > Configuration > Remote desktop resolution** runtime setting.

However, if your script contains synchronization calls, changing the resolution may introduce synchronization errors during replay, and your script may fail to run.

**Recommended:** Keep the default **As recorded** value.

## Working with Clipboard Data (RDP Protocol)

> **Note:** This topic applies to RDP Vuser scripts only.

VuGen allows you to copy and paste the text of a clipboard during an RDP session. You can copy the contents locally and paste them remotely, or vice versa—copy the contents from the remote machine and paste them locally. The copying of text is supported in TEXT, LOCALE, and UNICODE formats.

VuGen generates separate functions when copying or saving the clipboard data.

#### Code sample #1

The following example illustrates a copy operation on a local machine and a paste operation on a remote machine:

```
//Notifies the Remote Desktop that new data is available in the Local machine's
//clipboard.The data can be provided in three formats: TEXT, UNICODE and LOCALE
rdp_notify_new_clipboard_data(
"StepDescription=Send local clipboard formats 1",
"Snapshot=snapshot1.inf",
"FormatsList=RDP_CF_TEXT|RDP_CF_UNICODE|RDP_CF_LOCALE",
RDP_LAST );
rdp_key(
"StepDescription=Key Press 2",
"Snapshot=snapshot_9.inf",
"KeyValue=V",
"KeyModifier=CONTROL_KEY",
RDP_LAST );
//Provides clipboard data to the Remote Desktop when it requests the data.
rdp_send_clipboard_data(
"StepDescription=Set Remote Desktop clipboard 1",
"Snapshot=snapshot1.inf",
"Timeout=20",
REQUEST_RESPONSE, "Format=RDP_CF_UNICODE", "Text=text for clipboard",
RDP_LAST);
```

### Code sample #2

This example illustrates a copy operation on a remote machine and a paste operation on a local machine:

```
rdp_key(
"StepDescription=Key Press 2",
"Snapshot=snapshot_9.inf",
"KeyValue=C",
"KeyModifier=CONTROL_KEY",
RDP_LAST);
// The function requests the Remote Desktop UNICODE text and saves it to a
//parameter
rdp_receive_clipboard_data(
"StepDescription=Get Remote Desktop clipboard 1",
"Snapshot=snapshot1.inf",
"ClipboardDataFormat=RDP_CF_UNICODE",
"ParamToSaveData=MyParam",
RDP_LAST);
```

Normally, the Remote Desktop clipboard data is saved in UNICODE format. If the Remote Desktop requests data in the TEXT or LOCALE formats, the **rdp_send_clipboard_data** function automatically converts the content of MyParam from UNICODE into the requested format and sends it to the Remote Desktop. The Replay log indicates this conversion with an informational message. If the conversion is not possible, the step fails.

For more information about the rdp functions, see the Function Reference (**Help > Function Reference**).

## Correlating Clipboard Parameters

During a recording session, if the client sends the server the same data that it received, VuGen replaces the sent data with a parameter during code generation. VuGen performs this correlation only when the received and sent data formats are the same.

### Code sample #3

The following example shows how the same parameter, **MyParam**, is used for both receiving and sending the data.

```
// Receive the data from the server
rdp_receive_clipboard_data("StepDescription=Get Remote Desktop clipboard 1",
"Snapshot=snapshot_9.inf",
"Timeout=0",
"ClipboardDataFormat=RDP_CF_UNICODETEXT",
"ParamToSaveData=MyParam",
RDP_LAST);
...
```

```
// Send the data to the server
rdp_send_clipboard_data("StepDescription=Get Remote Desktop clipboard 1",
"Snapshot=snapshot_9.inf",
"Timeout=10",
REQUEST_RESPONSE, "Format=RDP_CF_UNICODETEXT", "Text={MyParam}",
RDP_LAST);
```

## RDP Snapshots - Overview

> **Note:** This topic applies to RDP Vuser scripts only.

Vuser scripts based on the RDP protocol utilize VuGen's Snapshot pane.

- For details on how to work with the Snapshot pane, see "Work with Snapshots" on page 297.
- For details on the Snapshot pane UI, see "Snapshot Pane" on page 85.

When you open an RDP Vuser script, VuGen's standard Snapshot pane functionality is available. The Snapshot pane displays snapshots of the remote display, saved during recording and playback of the Vuser script. Typically, these snapshots are used to synchronize playback of the Vuser script.

In addition to the basic Snapshot pane functionality, the Snapshot pane for RDP Vuser scripts lets you display snapshots in one of the following views:

- **Image.** Displays only the image of the snapshot and is ideal for visually comparing two images. This view displays the snapshot faster and requires less memory than the Full view. You can synchronize two snapshots in the Snapshot pane if both snapshots are displayed in the Image view. The Image view does not automatically scroll to show the area of a snapshot that is used for synchronization.
- **Full.** Scrolls to display the area that is used for synchronization. This view displays the snapshot slower and requires more memory than the Image view. You cannot synchronize two snapshots displayed in the Snapshot pane if either of the snapshots is displayed in the Full view. By default, snapshots are displayed in the Full view.

To display a specific synchronization snapshot in the Snapshot pane, do one of the following:

- In the Editor, select the step that contains a reference to the snapshot.
- In the Step Navigator, double-click the step that contains a reference to the snapshot.

When working with RDP Vuser scripts, the Snapshot pane lets you copy a snapshot to the clipboard, and display a snapshot of the most recent replay error. For more information on how to use the Snapshot pane, see "Work with Snapshots" on page 297.

## Image Synchronization Overview (RDP)

An RDP session executes remotely on a computer that is referred to as the server. All keyboard and mouse operations are done on the server, and it is the server that reacts to input from the keyboard and mouse. For example, when you double-click an application icon on the desktop, it is the server that realizes that a double-click took place, and that the application must be loaded.

When an RDP client connects to a server, the client does two things:

- It sends the server coordinates of actions. For example, 'clicked the left mouse button at coordinates (100, 100) on the screen'.
- It receives images from the server showing the current status of the screen after the action took place.

The RDP client (and therefore, the Vuser) does not know that the remote screen contains windows, buttons, icons, and other objects. It knows only that the screen contains an image and at what coordinates the user performed an action. To allow the server to correctly interpret an action, you set a synchronization point within the script. When you replay the script, the synchronization point instructs the Vuser to wait until the image on the server screen matches the corresponding image stored as part of the synchronization point.

### See also:

- "Image Synchronization Tips (RDP Protocol)" below

## Image Synchronization Tips (RDP Protocol)

> **Note:** This topic applies to RDP Vuser scripts only.

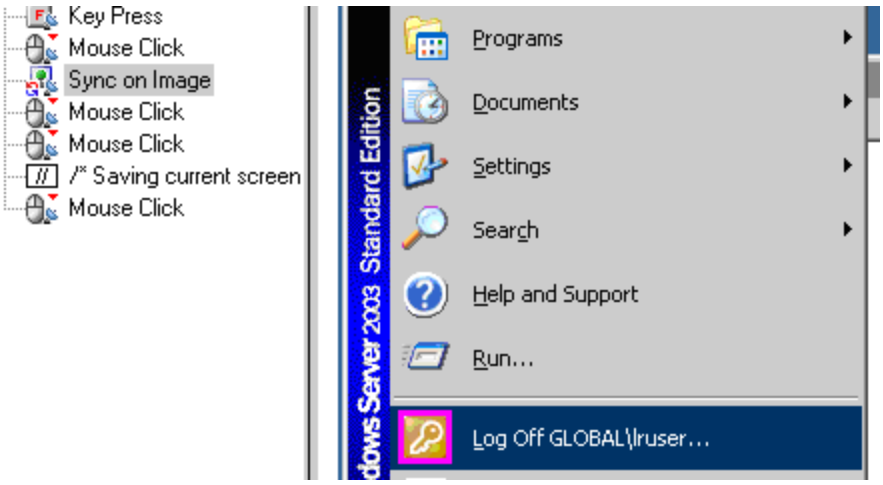
Use the following guidelines for effective image synchronization:

#### Synchronize on the Smallest Significant Area

When synchronizing on an image, try to synchronize only the part of the image that is necessary. Additional details within the image may not be reproduced during replay and could result in a synchronization failure.

For example, when synchronizing on an image of a button, select only the text itself and not the dotted lines around the text as they may not appear during replay.

When synchronizing a highlighted area, try to capture only the part of the image that is not effected by the highlighting. In the following example, perform a synchronization on the Log Off icon, but not the entire button, since the highlighting may not appear during replay, and the color could vary with different color schemes.




### Synchronize Before Every User Action

It is recommended that you synchronize before every mouse operation. You should also synchronize before the first **rdp_key** or **rdp_type** operation that follows a mouse operation.

## Image Synchronization - Shifted Coordinates (RDP Protocol)

> **Note:** This topic applies to RDP Vuser scripts only.

When replaying a script, a recorded object may appear at different coordinates on the screen. The object is the same, but its placement has been shifted. For example, during recording a window opened at coordinates (100, 100), but during replay at (200, 250).

In this case, the synchronization point will automatically find the new coordinates without any intervention on your part. It will automatically note the difference of 100 pixels in the horizontal axis and 150 pixels in the vertical axis.

All subsequent mouse operations that are coordinate dependent will use the modified coordinates, so that a mouse click recorded at (130, 130) will be replayed to (230, 280) = (130 + 100, 130 + 150).

You control the shifting of the coordinates through the **AddOffsetToInput** parameter in the **rdp_sync_on_image** step. You can override this parameter to either add or not add the differences in location during replay to the recorded coordinates for any further operations. If you do not override this parameter, VuGen takes its value from the default setting in the runtime settings.

The corresponding parameter in the operations (for example **rdp_mouse_click** or **rdp_mouse_drag**) is **Origin**. This parameter decides whether the operation should take its coordinates only from the 'clean' values that were recorded, or whether it should take into account the differences that were added by the last synchronization point. If not explicitly specified, VuGen takes the value for this parameter from the runtime settings.

# Setting Security Levels in RDP Vuser Scripts

> **Note:** This topic applies to RDP Vuser scripts only.

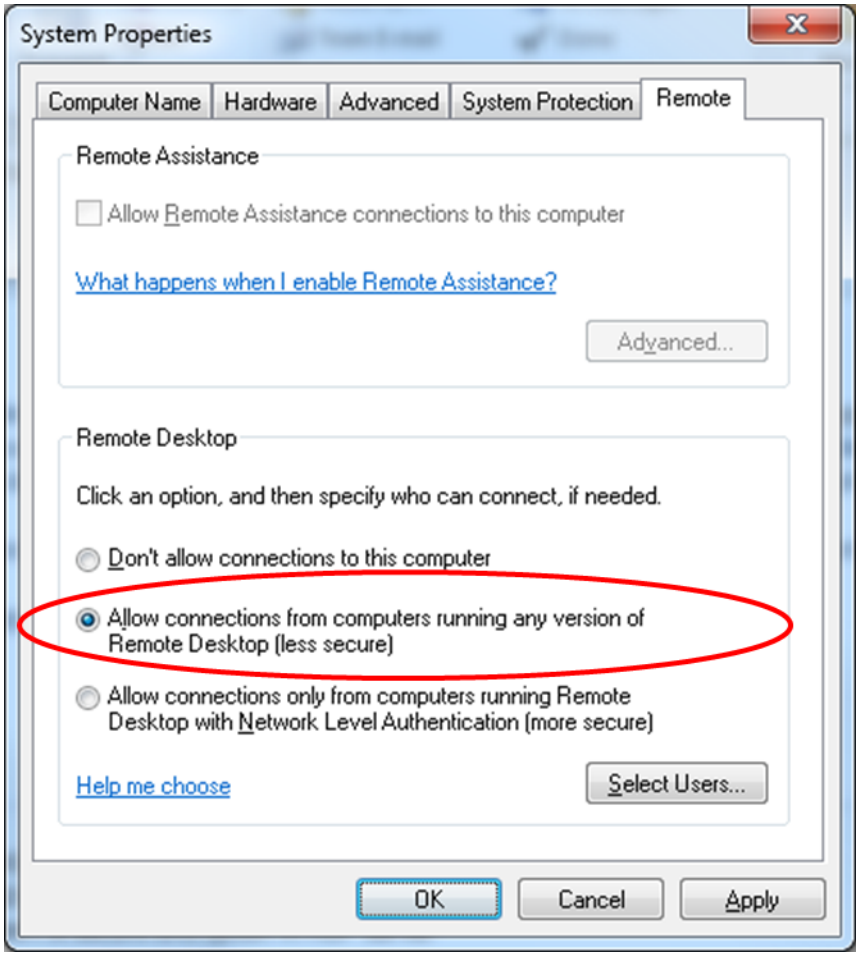
Remote Desktop Protocol (RDP) enables a client computer to connect to a server. Various security options are available for the connection, depending on the particular Windows operating systems that are installed on the client and server computers. The security options define security-related issues, such as the authentication and encryption, that are used for the connection.

The list of security options that are available for a Vuser script is different when you record a Vuser script and when you replay the script.

**Security levels when recording an RDP Vuser Script**

Standard RDP security is the only form of security that you can use when you record an RDP Vuser script. Before you record an RDP Vuser script, make sure that the server is configured to allow connections from computers that are running any version of Remote Desktop, and not only from computers that are running Remote Desktop with Network Level Authentication. You use the **Remote** tab in the System Properties dialog box on the server to set the security level that is required to establish the connection.

> **Note:** If your RDP session is launched through an RDP configuration file, you must disable *credssp authentication* in the configuration file, using the following string:
>
> `enablecredsspsupport:i:0`




**Security levels when replaying an RDP Vuser Script**

You can use the Vuser script's runtime settings to specify the security that is used for the connection when the Vuser runs. The available security levels are:

- **RDP:** Connects using standard RDP security. RDP provides the least secure connection.
- **SSL:** Connects using SSL as an external security protocol to enhance the standard RDP security. SSL provides a moderate level of security.
- **CredSSP:** Connects using the Credential Security Support Provider (CredSSP) protocol. CredSSP provides the most secure connection.

> **Note:** If you specify CredSSP authentication, you must make certain changes to the Vuser script each time the script is regenerated. For details, see Modifying a script to support CredSSP authentication below.

The security level that you specify in the runtime settings is an indication to the server of the maximum level of security that is supported by the client. However, the security that is actually used for the connection is defined by the server settings. For example, if you specify CredSSP as the encryption level in the runtime settings, when you run the Vuser, the Vuser will inform the server that the Vuser

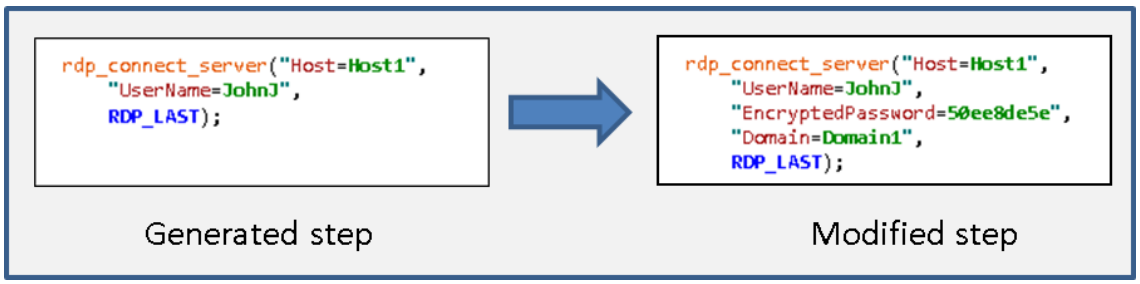
supports CredSSP, SSL, and RDP security. If the server supports only RDP security (for example,its operating system is Windows 2003), then the connection will be made using RDP.

To set the RDP security level for the Vuser script, click **Replay > Runtime Settings > RDP > Configuration** and then select the required level from the **Supported Encryption Level** list.

**Modifying a script to support CredSSP authentication**

If you specify CredSSP authentication in the Vuser script's runtime settings, you must perform the following tasks each time the script is regenerated:

1. In the **rdp_connect_server** step in the script, modify the step to provide the user name, password, and domain that are required to access the server. For details on the **rdp_connect_server** step, see the Function Reference (**Help > Function Reference**).



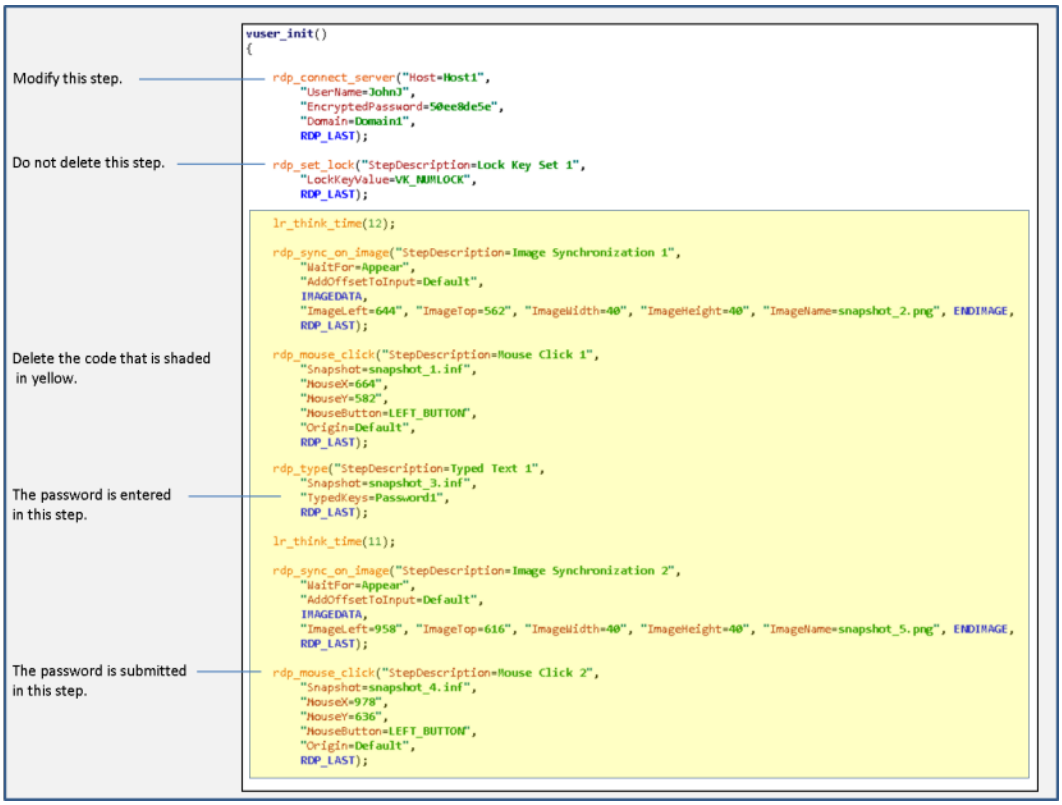

2. Remove the block of code that contains the login-related mouse, keyboard, and image synchronization steps from the generated script, as described below.
   a. Locate the **rdp_connect_server** step in the Vuser script.

      The step after the **rdp_connect_server** step is the first step in the block of code to delete.
   b. Locate the **rdp mouse_click** step or the **rdp_key step** that submits the password to the server.

      This is the last step in the block of code to delete.

> **Note:** If an **rdp_set_lock** step exists immediately after the **rdp_connect_server** step, do not delete the **rdp_set_lock** step.

c. Delete all the steps in the block of code that is defined above.




## RDP Agent (for Microsoft Terminal Server) Overview

The Agent for Microsoft Terminal Server is an optional utility that you can install on the RDP server. It provides enhancements to the normal RDP functionality. It is provided in the LoadRunner installation package and you can install it on any RDP server. The agent provides you with more intuitive and readable scripts, built-in synchronization, and detailed information about relevant objects. Note that when you run RDP Vusers with the agent installed, each Vuser runs its own process of lrrdpagent.exe. This results in a slight reduction in the number of Vusers that can run on the server machine.

### Tips for Using the Agent for Microsoft Terminal Server

- When opening application menus (e.g. File, Edit...) with the mouse, sync steps will sometimes fail. To avoid this issue, use the keyboard to select menu items when recording.
- When you add a **rdp_sync_object_mouse_click** step manually, the coordinates given are absolute coordinates (relating to the entire screen). To create the synchronization point, you need to calculate the offset in the window (relative coordinates) of the desired click location and modify the absolute coordinates accordingly for the synchronization to successfully replay.
- If a synchronization object exists at the correct location and time during replay, but is covered by another window (such as a pop-up), then the synchronization step will pass and a click will be executed on the window which is covering the synchronization point and therefore harm the script flow.

- During recording, if you want to return the application window to the foreground, either click on the title bar, or use the keyboard (ALT+TAB). Note that if you click inside the application window to return it to the foreground, the RDP session may terminate unexpectedly.

## Enhancements to RDP functionality

The Agent for Microsoft Terminal Server provides the following enhancements to the normal RDP functionality:

### Resumed Sessions

Beginning with LoadRunner version 12.50, the RDP Agent process is always active on the server machine. This allows you to record a resumed RDP session on a machine with the RDP agent—you do not need to start a new one.

### Object Detail Recording

When the Agent for Microsoft Terminal Server is installed, VuGen can record specific information about the object that is being used instead of general information about the action. For example, VuGen generates **rdp_sync_object_mouse_click** and **rdp_sync_object_mouse_double_click** steps instead of **rdp_mouse_click** and **rdp_mouse_double_click** that it generates without the agent.

The following example shows a double-mouse-click action recorded with and without the agent installation. Note that with the agent, VuGen generates sync_object functions for all of the mouse actions.

```
rdp_sync_object_mouse_double_click("StepDescription=Mouse Double Click on
Synchronized Object 1",
        "Snapshot=snapshot_12.inf",
        "WindowTitle=RDP2",
        "Attribute=TEXT",
        "Value=button1",
        "MouseX=100",
        "MouseY=71",
        "MouseButton=LEFT_BUTTON",
        RDP_LAST);
rdp_mouse_double_click("StepDescription=Mouse Double Click 1",
        "Snapshot=snapshot_2.inf",
        "MouseX=268",
        "MouseY=592",
        "MouseButton=LEFT_BUTTON",
        "Origin=Default",
        RDP_LAST);
```

### Expanded Right-Click Menu

When you click within a snapshot, you can insert several functions into the script using the right-click menu. When the agent is not active, you are limited to inserting only **rdp_mouse_click**, **rdp_mouse_**

**double_click**, and **rdp_sync_on_image** steps. When the agent is installed, you are able to insert all possible steps that involve the RDP agent:

- **rdp_get_object_info** and **rdp_sync_on_object_info**. Provide information about the state of the object, and synchronize on a specific object property such as: ENABLED, FOCUSED, CONTROL_ID, ITEM_TEXT, TEXT, CHECKED, and LINES.
- **rdp_sync_on_text** and **rdp_get_text**. For details, see the Function Reference (**Help > Function Reference**).

In the following example, the **rdp_sync_on_object_info** function provides synchronization by waiting for the Internet Options dialog box to come into focus.

```
rdp_sync_on_object_info("StepDescription=Sync on Object Info 0",
        "Snapshot=snapshot_30.inf",
        "WindowTitle=Internet Options",
        "ObjectX=172",
        "ObjectY=155",
        "Attribute=FOCUSED",
        "Value={valueParam}",
        "Timeout=10",
        "FailStepIfNotFound=No",
        RDP_LAST);
```

# Install/Uninstall the RDP Agent

> **Note:** This topic applies to RDP Vuser scripts only.

If you are upgrading the agent, make sure to uninstall the previous version before installing the new one (see uninstallation instructions below).

## Install the RDP Agent (Agent for Microsoft Terminal Server)

1. If your server requires administrator permissions to install software, log in as an administrator to the server.
2. Locate the installation file, **Setup.exe**, in your LoadRunner installation package's **Additional Components\ Agent for Microsoft Terminal Server** folder.
3. Install the RDP agent on your RDP server machine (not on Load Generator machines), following the installation wizard to completion.

## Use the RDP Agent (Agent for Microsoft Terminal Server)

Set the recording options before recording a Vuser script.

1. In the Start Recording dialog box, click **Options**.
2. In the Advanced Code Generation node, select the **Use RDP Agent** check box.

### Uninstall the RDP Agent (Agent for Microsoft Terminal Server)

1. If your server requires administrator privileges to remove software, log in to the server as an administrator.
2. From **Add or remove a program**, remove **Software Agent for Microsoft Terminal Server**.

## Add Image Synchronization Points to a Script

> **Note:** This topic applies to RDP Vuser scripts only.

1. Select an operation to which you would like to add a synchronization point in your script.
2. Right-click on the image snapshot and select Insert Synch On Image from the menu. The cursor will change to a cross-hair.
3. Mark the area on the screen that you would like to synchronize upon by clicking on the left button and dragging the box to enclose the area. When you release the mouse button, the Sync on Image dialog box opens.
4. Click **OK**. VuGen adds a new Sync on Image step before the selected step. When you select this step, VuGen displays a snapshot that contains a pink box around the area you selected for synchronization.

The next time you replay the script, it will wait until the image returned by the server matches the image you selected.

## Failed Image Synchronization Dialog Box (RDP Protocol)

This dialog box opens when an image synchronization fails during the replay of a script. You can stop the script or continue the replay despite the error.

The content of this dialog box varies depending on the reason for the failed synchronization:

- **Append Snapshot.** The Failed Image Synchronization - Append Snapshot dialog box opens when the replay image is so different from the record image that changing the tolerance level will not help.
- **Raise Tolerance.** The Failed Image Synchronization - Raise Tolerance dialog box opens when the script replay failed to find the exact image requested, but if the tolerance level for performing synchronization on images was relaxed, then it would have succeeded in finding the image.
- **Lower Tolerance.** The Failed Image Synchronization - Lower Tolerance dialog box opens when the script replay fails to meet the NotAppear or Change conditions. VuGen detected an image match where you expected it not to detect one. If the tolerance level was reduced, the recorded and replay images would not match, and the NotAppear or Change conditions would be met resulting in a successful replay.
- **Non Specified.** The Failed Image Synchronization dialog box opens when the script replay fails to meet any of the synchronization conditions such as NotAppear or Change. VuGen did not find another image at the original coordinates that could be appended to the script.

| | |
|---|---|
| **To access** | Opens automatically when an image synchronization fails. |
| **See also** | "Image Synchronization Overview (RDP)" on page 643 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Stop** | Consider the mismatch between the snapshots to be an error. This error will be handled like all other errors and halt the execution of the script. |
| **Continue** | This button performs different actions depending on the type of dialog box:<br>• **Append Snapshot.** Accept the mismatch. VuGen appends the replay snapshot as a "record" snapshot for the step. In future replays of the step, VuGen uses all existing record snapshots and the appended snapshot as the basis for comparison between screens. If the replay returns any of the record snapshots, the Vuser will not fail. You can view the original and appended snapshots for a step by clicking the navigation arrows in the Snapshot pane toolbar.<br>• **Lower Tolerance.** Accept the mismatch and lower the tolerance level so that VuGen permits a smaller mismatch between the record images and those displayed during replay.<br>• **Raise Tolerance.** Accept the mismatch and raise the tolerance level so that VuGen permits a greater mismatch between the record images and those displayed during replay.<br>• **Non-specified.** Accept the mismatch, and do not make any changes in the script. Continue script execution despite the mismatch.<br>**Note:** Raising or lowering the tolerance level from the dialog box changes the level for the current step only. To change the tolerance level for the whole script, change the **Default tolerance for image synchronization** setting in the **Runtime Settings > RDP > Synchronization** view. |

## Troubleshooting and Limitations for RDP

This section describes troubleshooting information for RDP scripts using the Agent for Microsoft Terminal Server.

**Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

- Clipboard sharing supports only short simple textual content.
- When recording with RDP Agent, applications which were developed using CBuilder may not record

properly.

- RDP does not support 32-bit color depth. If recording uses this color depth, VuGen automatically switches to a lower color depth and a "[RDP Analyzer Warning (790: 418)] 32-bit color depth is not supported, switch to lower one". warning log item appears in the Recording Window.
- For Windows 8, we recommend using the Windows key to switch between the Desktop and Startup screen. This reduces the number of generated mouse calls and simplifies debugging.
- When working on a 64-bit Windows 8.1 machine, you must manually install VcDist_x32 for Visual Studio 2012 before installing the RDP agent.
- Replay fails on **rdp_sync_object_mouse_click/double_click** steps.

  **Workaround 1:** Modify RDPAgentCodeGen.cfg file

  The **RDPAgentCodeGen.cfg** file can configure VuGen to automatically create an **rdp_sync_on_image** and **rdp_mouse_click** step the next time the script is generated for each **rdp_sync_object_mouse_click/double_click** steps which occur within a given window. To do this, you specify the name of the window, update a variable which counts the total number of windows for which this process occurs, and regenerate the script.

  a. Open the **RDPAgentCodeGen.cfg file** in the **Script Directory > data** folder.
  b. Open the **Step Navigator** and double-click the step that failed.
  c. Copy the name of the window.
  d. In the **RDPAgentCodeGen.cfg** file, increase the value of **NumberOfTitles** by 1.
  e. Add a line as follows:

     ```
     WindowTitleX=<name of window>
     ```

     where **X** is the new value of **NumberOfTitles**.
  f. Regenerate the script.

  > **Note:** The **RDPAgentCodeGen.cfg file** can be used to automatically produce **rdp_sync_on_image** and **rdp_mouse_click** steps in a similar way for **rdp_sync_object_mouse_click/double_click** steps which are specified in different ways as well. Steps can be targeted based on the class attribute of the control. For more information, contact Software Support.

  **Workaround 2:** Manually Insert a New Step

  Manually insert an **rdp_sync_on_image** and **rdp_mouse_click** step for each step that fails. This method is less desirable, since steps added in this way will be lost if the script is regenerated.

- Connecting to a Windows 10 or Windows Server 2016 server

  When using the RDP protocol to record a connection to a Windows 10 or Windows Server 2016 server, the following error may appear on the RDP client: "An authentication error occurred."

  **Resolution**

  Perform the following procedure on the server:

a. Open the Local Group Policy Editor by typing "gpedit.msc" into either a Run prompt or the Start menu.

b. In the Local Group Policy Editor, select **Local Computer Policy > Computer Configuration > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Security**.

c. Enable "Require use of specific security layer for remote (RDP) connections" and set the security layer to "**SSL (TLS 1.0)**." If you are testing a non-SSL connection, set the security layer to **Negotiate**.

- Failed to connect to RDP server on Windows 10 server

  When using the RDP protocol to record a connection to a Windows 10 server, the following error may appear on the RDP client: "An authentication error occurred."

  **Resolution**

  Perform the following procedure on the server:

  a. Open the Local Group Policy Editor by typing "gpedit.msc" into either a Run prompt or the Start menu.

  b. In the Local Group Policy Editor, select **Local Computer Policy > Computer Configuration > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Security**.

  c. Enable "Require use of specific security layer for remote (RDP) connections" and set the security layer to **RDP**.

  > **Note:** Alternatively, open **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp** in the Registry Editor, and set the Value data for **SecurityLayer** to **1**.

# RTE Protocol

## RTE Protocol Overview

An RTE Vuser types character input into a terminal emulator, submits the data to a server, and then waits for the server to respond. For instance, suppose that you have a server that maintains customer information for a maintenance company. Each time a field service representative makes a repair, he accesses the server database by modem using a terminal emulator. The service representative accesses information about the customer and then records the details of the repair that he performs.

You could use RTE Vusers to emulate this case. An RTE Vuser would:

1. Type **60** at the command line to open an application program.
2. Type **F296**, the field service representative's number.
3. Type **NY270**, the customer number.

4. Wait for the word "Details" to appear on the screen. The appearance of "Details" indicates that all the customer details are displayed on the screen.
5. Type **Changed gasket P249, and performed Major Service** the details of the current repair.
6. Type **Q** to close the application program.

You use VuGen to create RTE Vuser scripts. The script generator records the actions of a human user on a terminal emulator. The script generator records the keyboard input from the terminal window, generates the appropriate statements, and inserts them into the Vuser script. While you record, the script generator automatically inserts synchronization functions into the script. For details, see "RTE Synchronization Overview" on page 662.

The functions developed to emulate a terminal communicating with a server are called TE Vuser functions. Each TE Vuser function has a **TE** prefix. VuGen automatically records most of the TE functions listed in this section during an RTE recording session. You can also manually program any of the functions into your script**.**

For syntax and examples of the TE functions, see the Function Reference (**Help > Function Reference**).

An RTE Vuser emulates the actions of a real user. Human users use terminals or terminal emulators to operate application programs.

In the RTE Vuser environment, a Vuser replaces the human. The Vuser operates PowerTerm, a terminal emulator.

PowerTerm works like a standard terminal emulator, supporting common protocols such as IBM 3270 =; 5250, VT100, VT220, and VT420-7.

## Working with Ericom Terminal Emulation

VuGen supports record and replay with Ericom Terminal Emulators.

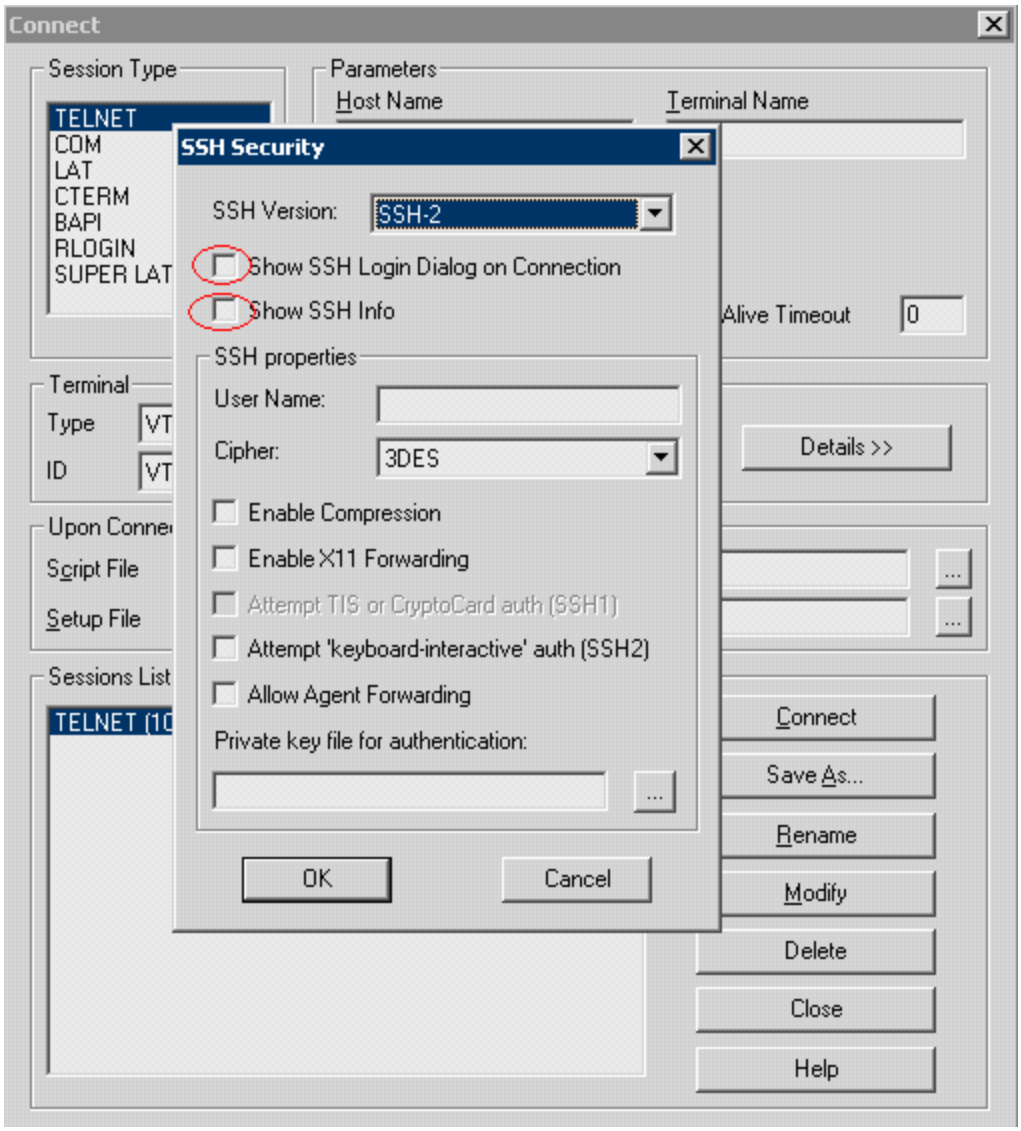
The Ericom support handles escape sequences during record and replay. Ericom's PowerTerm lets you map PC keys to custom escape sequences. For information about mapping, see the PowerTerm help.

When a user presses mapped keys while recording an Ericom VT session, VuGen generates **TE_send_text** functions instead of the standard **TE_type**. This allows the script to handle custom escape sequences in a single step. For more information, see the Function Reference (**Help > Function Reference**) for the **TE_send_text** function.

## SSL and SSH Support for Ericom

VuGen also supports SSL/SSH record and replay for the RTE Ericom library. To work with SSL or SSH, you select the type in the **Security** section of the Connect dialog box.

When working with SSH Security, by default VuGen opens a popup dialog box prompting you for more information. We recommend that you disable the **Show** options to prevent the pop-ups from being issued. If you enable these pop-ups, it may affect the replay. You can access the advanced security options by clicking the **Details** button.




**Note:** Running multiple Vusers with SSH works only when you run each Vuser as a process. By default, a Vuser runs as a thread. To change it to run as a process, go to **Runtime Settings** > **Miscellaneous Multithreading** and enable **Run Vuser as a process**. This is a 3rd party software limitation when SSH is used.

# Typing Input into a Terminal Emulator

Two TE Vuser functions enable Vusers to "type" character input into the PowerTerm terminal emulator:

- **TE_type** sends characters to the terminal emulator. When recording, the VuGen automatically generates **TE_type** functions for keyboard input to the terminal window. For details, see below.
- **TE_typing_style** determines the speed at which the Vuser types. You can manually define the typing style by inserting a **TE_typing_style** function into the Vuser script. Alternatively, you can set the typing style by using the runtime settings. For more information, see "Runtime Settings" on page 301.

> **Note:** While recording an RTE Vuser script, do not use the mouse to relocate the cursor within the terminal emulator window. VuGen does not record these cursor movements.

### Using the TE_type Function

When you record a script, the VuGen records all keyboard input and generates appropriate **TE_type** functions. During execution, **TE_type** functions send formatted strings to the terminal emulator.

Keyboard input is defined as a regular text string (including blank spaces). For example:

```
TE_type ("hello, world");
```

Input key names longer than one character are represented by identifiers beginning with the letter k, and are bracketed within greater-than/less-than signs (< >).

For example, the following function depicts the input of the Return key followed by the Control and y keys:

```
TE_type("<kReturn><kControl-y>");
```

Some other examples include: <kF1>, <kUp>, <kF10>, <kHelp>, <kTab>.

To determine a key name, record an operation on the key, and then check the recorded statement for its name.

> **Note:** When you program a **TE_type** statement (rather than recording it), use the key definitions provided in the Function Reference (**Help > Function Reference**).

## Setting the Timeout Value for TE_type

If a Vuser attempts to submit a **TE_type** statement while the system is in X SYSTEM (or input inhibited) mode, the Vuser will wait until the X SYSTEM mode ends before typing. If the system stays in X SYSTEM mode for more than TE_XSYSTEM_TIMEOUT milliseconds, then the **TE_type** function returns a TE_TIMEOUT error.

You can set the value of TE_XSYSTEM_TIMEOUT by using **TE_setvar**. The default value for TE_XSYSTEM_TIMEOUT is 30 seconds.

## Allowing a Vuser to Type Ahead

Under certain circumstances you may want a Vuser to submit a keystroke even though the system is in X SYSTEM (or input inhibited) mode. For example, you may want the Vuser to press the Break key. You use the TE_ALLOW_TYPEAHEAD variable to enable the Vuser to submit a keystroke even though the system is in X SYSTEM mode.

Set TE_ALLOW_TYPEAHEAD to zero to disable typing ahead, and to any non-zero number to permit typing ahead. You use **TE_setvar** to set the value of TE_ALLOW_TYPEAHEAD. By default, TE_ALLOW_TYPEAHEAD is set to zero, preventing keystrokes from being sent during X SYSTEM mode.

For more information about the **TE_type** function and its conventions, see the Function Reference (**Help > Function Reference**).

### Setting the Typing Style

You can set two typing styles for RTE Vuser: FAST and HUMAN. In the FAST style, the Vuser types input into the terminal emulator as quickly as possible. In the HUMAN style, the Vuser pauses after typing each character. In this way, the Vuser more closely emulates a human user typing at the keyboard.

You set the typing style using the **TE_typing_style** function. The syntax of the **TE_typing_style** function is:

int TE_typing_style (char * style );

where style can be FAST or HUMAN. The default typing style is HUMAN. If you select the HUMAN typing style, the format is:

HUMAN, delay [, first_delay]

The delay indicates the interval (in milliseconds) between keystrokes. The optional parameter first_delay indicates the wait (in milliseconds) before typing the first character in the string. For example,

```
TE_typing_style ("HUMAN, 100, 500");
TE_type ("ABC");
```

means that the Vuser will wait 0.5 seconds before typing the letter A; it will then wait 0.1 seconds before typing "B" and then a further 0.1 seconds before typing "C".

For more information about the **TE_typing_style** function and its conventions, see the Function Reference (**Help > Function Reference**).

In addition to setting the typing style by using the **TE_typing_style** function, you can also use the runtime settings. For details, see "Runtime Settings" on page 301.

## Generating Unique Device Names

Some protocols, such as APPC, require a unique device name for each terminal that logs on to the system. Using the runtime settings, you can specify that the TE_connect function generate a unique 8-character device name for each Vuser, and connect using this name. Although this solves the requirement for uniqueness, some systems have an additional requirement: The device names must conform to a specific format. See "Runtime Settings" on page 301 for more information.

To define the format of the device names that the TE_connect function uses to connect a Vuser to the system, add an **RteGenerateDeviceName** function to the Vuser script. The function has the following prototype:

```
void RteGenerateDeviceName(char buf[32])
```

The device name should be written into **buf**.

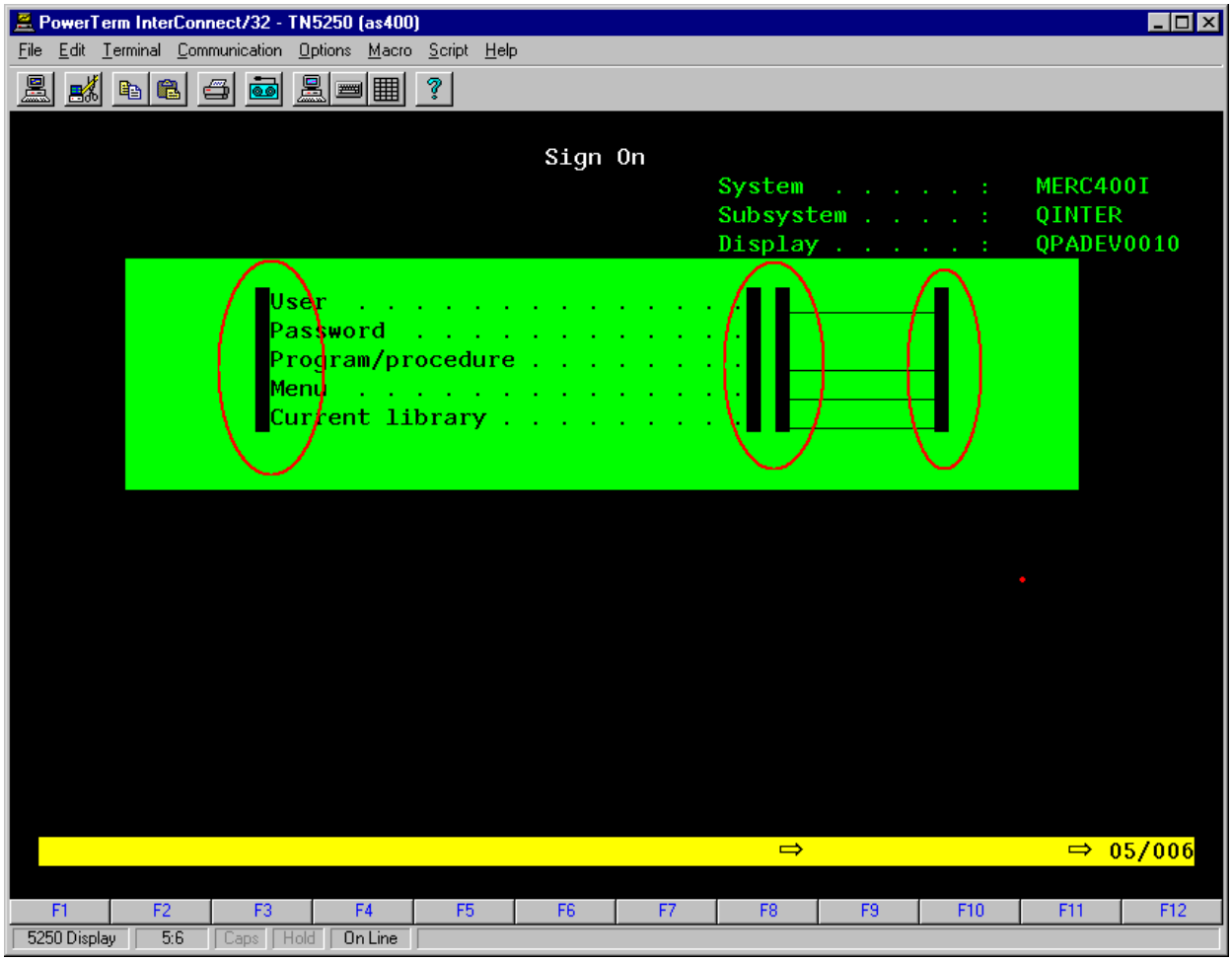
If an **RteGenerateDeviceName** function exists in a Vuser script, the Vuser calls the function each time a new device name is needed. If no **RteGenerateDeviceName** function is defined in the script—and unique device names are required—the **TE_connect** function generates the required names.

In the following example, the RteGenerateDeviceName function generates unique device names with the format "TERMx". The first name is TERM0, followed by TERM1, TERM2, and so forth.

```
RteGenerateDeviceName(char buf[32])
{
    static int n=0;
    sprintf(buf, "TERM%d", n);
    n=n+1;
}
```

## Setting the Field Demarcation Characters

Some terminal emulators use demarcation characters to mark the beginning and the end of each field. These demarcation characters are not visible—appearing on the screen as spaces. In the terminal emulator shown below, the colors in the middle section of the screen have been inverted to display the field demarcation characters. These characters are surrounded by ellipses.




The **TE_wait_text**, **TE_get_text**, and **TE_find_text** functions operate by identifying the characters in a specified portion of the screen. If a field demarcation character is located within the specified section, you can identify the character as a space or an ASCII character. You use the TE_FIELD_CHARS system variable to specify the method of identification. You can set TE_FIELD_CHARS to 0 or 1:

- 0 specifies that the character in the position of the field demarcation characters is returned as a space.
- 1 specifies that the character in the position of the field demarcation characters is returned as an ascii code (ascii 0 or ascii 1).

By default, TE_FIELD_CHARS is set to 0.

You retrieve and set the value of TE_FIELD_CHARS by using the **TE_getvar** and **TE_setvar** functions.

## Reading Text from the Terminal Screen

There are several Vuser functions that RTE Vusers can use to read text from the terminal screen. You can use these functions, **TE_find_text** and **TE_get_text_line**, to check that the terminal emulator is responding correctly, or to enhance the logic in your scripts.

After recording, you can manually insert **TE_find_text** and **TE_get_text_line** statements directly into your RTE Vuser scripts.

### Searching for Text on the Screen

The **TE_find_text** function searches for a line of text on the screen. The syntax of the function is:

```
int TE_find_text (char *pattern, int col1, int row1, int col2, int row2,
        int *retcol, int *retrow, char *match );
```

This function searches for text matching pattern within the rectangle defined by col1, row1, col2, row2. Text matching the pattern is returned to match, and the actual row and column position is returned to retcol and retrow. The search begins in the top-left corner. If more than one string matches pattern, the one closest to the top-left corner is returned.

The **pattern** can include a regular expression. See the Function Reference (**Help > Function Reference**) for details on using regular expressions.

You must manually type **TE_find_text** statements into your Vuser scripts. For details on the syntax of the **TE_find_text** function, see the Function Reference (**Help > Function Reference**).

### Reading Text from the Screen

The **TE_get_text_line** function reads a line of text from the area of the screen that you designate. The syntax of the function is:

```
char *TE_get_text_line (int col, int row, int width, char * text );
```

This function copies a line of text from the terminal screen to a buffer text. The first character in the line is defined by col, row. The column coordinate of the last character in the line is indicated by width. The text from the screen is returned to the buffer text. If the line contains tabs or spaces, the equivalent number of spaces is returned.

In addition, the **TE_get_cursor_position** function can be used to retrieve the current position of the cursor on the terminal screen. The **TE_get_line_attribute** function returns the character formatting (for instance, bold or underline) of a line of text.

You must manually type **TE_get_text_line** statements into your Vuser scripts. For details on the syntax of the **TE_get_text_line** function, see the Function Reference (**Help > Function Reference**).

## RTE Synchronization Overview

Depending on the system you are testing, you may need to synchronize the input that a Vuser sends to a terminal emulator with the subsequent responses from the server. When you synchronize input, you instruct the Vuser to suspend script execution and wait for a cue from the system, before the Vuser performs its next action. For instance, suppose that a human user wants to submit the following sequence of key strokes to a bank application:

1. Type 1 to select "Financial Information" from the menu of a bank application.
2. When the message "What information do you require?" appears, type 3 to select "Dow Jones

Industrial Average" from the menu.

3. When the full report has been written to the screen, type 5 to exit the bank application.

In this example, the input to the bank application is synchronized because at each step, the human user waits for a visual cue before typing.
This cue can be either the appearance of a particular message on the screen, or stability of all the information on the screen.

You can synchronize the input of a Vuser in the same way by using the TE-synchronization functions, **TE_wait_sync**, **TE_wait_text**, **TE_wait_silent**, and **TE_wait_cursor**. These functions effectively emulate a human user who types into a terminal window and then waits for the server to respond, before typing in the next command.

The **TE_wait_sync** function is used to synchronize block-mode (IBM) terminals only. The other TE-synchronization functions are used to synchronize character-mode (VT) terminals.

When you record an RTE Vuser script, VuGen can automatically generate and insert **TE_wait_sync**, **TE_wait_text**, and **TE_wait_cursor** statements into the script. You use VuGen's recording options to specify which synchronization functions VuGen should insert.

> **Note:** Do not include any synchronization statements in the Vuser_end section of a Vuser script. Since a Vuser can be aborted at any time, you cannot predict when the Vuser_end section will be executed.

## Synchronizing Block-Mode (IBM) Terminals

The **TE_wait_sync** function is used for synchronization RTE Vusers operating block-mode (IBM) terminals. Block-mode terminals display the "X SYSTEM" message to indicate that the system is in Input Inhibited mode. When a system is in the Input Inhibited mode no typing can take place because the terminal emulator is waiting for a transfer of data from the server.

When you record a script on a block-mode terminal, by default, VuGen generates and inserts a **TE_wait_sync** function into the script each time the "X SYSTEM" message appears. You use VuGen's recording options to specify whether or not VuGen should automatically insert **TE_wait_sync** functions.

When you run a Vuser script, the **TE_wait_sync** function checks if the system is in the X SYSTEM mode. If the system is in the X SYSTEM mode, the **TE_wait_sync** function suspends script execution. When the "X SYSTEM" message is removed from the screen, script execution continues.

> **Note:** You can use the **TE_wait_sync** function only with IBM block-mode terminals emulators (5250 and 3270).

In general, the **TE_wait_sync** function provides adequate synchronization for all block-mode terminal emulators. However, if the **TE_wait_sync** function is ineffective in a particular situation, you can enhance the synchronization by including a **TE_wait_text** function. For more information on the **TE_wait_text**

function, see "Synchronizing Character-Mode (VT) Terminals" on the next page, and the Function Reference (**Help > Function Reference**).

In the following script segment, the Vuser logs on with the user name "QUSER" and the password "MYLAB". The Vuser then presses `Enter` to submit the login details to the server. The terminal emulator displays the X SYSTEM message while the system waits for the server to respond.

The TE_wait_sync statement causes the Vuser to wait until the server has responded to the login request, that is, for the X SYSTEM message to be removed—before executing the next line of the script.

```
TE_type("QUSER");
lr_think_time(2);
TE_type("<kTab>MYLAB");
lr_think_time(3);
TE_type("<kEnter>");
TE_wait_sync();
....
```

When a **TE_wait_sync** function suspends the execution of a script while an X SYSTEM message is displayed, the Vuser continues to monitor the system—waiting for the X SYSTEM message to disappear. If the X SYSTEM message does not disappear before the synchronization timeout expires, the **TE_wait_sync** function returns an error code. The default timeout is 60 seconds.

### Set the TE_wait_sync synchronization timeout

1. Select **Vuser** > **Runtime Settings**. The runtime settings dialog box appears.
2. Select the **RTE:RTE** node in the Runtimesetting tree.
3. Under **X SYSTEM Synchronization**, enter a value (in seconds) in the **Timeout** box.
4. Click **OK** to close the runtime settings dialog box.

   After a Vuser executes a **TE_wait_sync** function, the Vuser waits until the terminal is no longer in the X S YSTEM mode . When the terminal returns from the X S YSTEM mode, the Vuser continues to monitor the system for a short period to verify that the terminal is fully stable, that is, that the system does not return to the X S YSTEM mode. Only then does the **TE_wait_sync** function terminate and allow the Vuser to continue executing its script. The period that the Vuser continues to monitor the system, after the system has returned from the X SYSTEM mode, is known as the stable time. The default stable time is 1000 milliseconds.

   You may need to increase the stable time if your system exhibits the following behavior:

   When a system returns from the X SYSTEM mode, some systems "flickers" to and from the X SYSTEM for a short period of time until the system stabilizes. If the system remains out of the X SYSTEM mode for more than one second, and then returns to the X SYSTEM mode, the **TE_wait_sync** function will assume that the system is stable. If a Vuser then tries to type information to the system, the system will shift into keyboard-locked mode.

   Alternatively, if your system never flickers when it returns from the X SYSTEM mode, you can reduce the stable time to less than the default value of one second.

### Change the stable time for TE_wait_sync functions

1. Select **Vuser** > **Runtime Settings**. The runtime settings dialog box appears.
2. Select the **RTE:RTE** node.
3. Under **X SYSTEM Synchronization**, enter a value (in milliseconds) in the **Stable time** box.
4. Click **OK** to close the runtime settings dialog box.

   For more information on the **TE_wait_sync** function, see the Function Reference (**Help > Function Reference**).

   You can instruct VuGen to record the time that the system remains in the X SYSTEM mode each time that the X SYSTEM mode is entered. To do so, VuGen inserts a TE_wait_sync_transaction function after each TE_wait_sync function, as shown in the following script segment:

   ```
   TE_wait_sync();
   TE_wait_sync_transaction("syncTrans1");
   ```

   Each **TE_wait_sync_transaction** function creates a transaction with the name "default." This allows you to analyze how long the terminal emulator waits for responses from the server during a scenario run. You use the recording options to specify whether VuGen should generate and insert **TE_wait_sync_transaction** statements.

### Instruct VuGen to insert TE_wait_sync_transaction statements

1. Select **Vuser** > **Recording Options**. The Recording Settings dialog box appears.
2. Select the **Generate Automatic X SYSTEM transactions** option, and then click **OK**.

## Synchronizing Character-Mode (VT) Terminals

You select various types of synchronization for character-mode (VT) terminals according to:

- the design of the application that is running in the terminal emulator
- the specific action to be synchronized

### Wait for the cursor to appear at a specific location

The preferred method of synchronization for VT type terminals is cursor synchronization. Cursor synchronization is particularly useful with full-screen or form-type applications, as opposed to scrolling or TTY-type applications.

Cursor synchronization uses the **TE_wait_cursor** function. When you run an RTE Vuser script, the **TE_wait_cursor** function instructs a Vuser to suspend script execution until the cursor appears at a specified location on the screen. The appearance of the cursor at the specified location means that the application is ready to accept the next input from the terminal emulator.

The syntax of the **TE_wait_cursor** function is:

```
int TE_wait_cursor (int col, int row, int stable, int timeout );
```

During script execution, the **TE_wait_cursor** function waits for the cursor to reach the location specified by col , row.

The **stable** parameter specifies the time (in milliseconds) that the cursor must remain at the specified location. If you record a script using VuGen, **stable** is set to 100 milliseconds by default. If the client application does not become stable in the time specified by the **timeout** parameter, the function returns TIMEOUT. If you record a script using VuGen, **timeout** is set by default to the value of TIMEOUT, which is 90 seconds. You can change the value of both the **stable** and **timeout** parameters by directly editing the recorded script.

The following statement waits for the cursor to remain stable for three seconds. If the cursor doesn't stabilize within 10 seconds, the function returns TIMEOUT.

```
TE_wait_cursor (10, 24, 3000, 10);
```

For more information on the **TE_wait_cursor** function, see the Function Reference (**Help > Function Reference**).

You can instruct VuGen to automatically generate TE_wait_cursor statements, and insert them into a script, while you record the script. The following is an example of a TE_wait_cursor statement that was automatically generated by VuGen:

```
TE_wait_cursor(7, 20, 100, 90);
```

## Instruct VuGen to automatically generate and insert TE_wait_cursor statements while recording

1. Select **Vuser** > **Recording Options**. The Recording Settings dialog box appears.
2. Under **Generate Automatic Synchronization Commands** select the **Cursor** check box, and then click **OK**.

## Wait for text to appear on the screen

You can use text synchronization to synchronize an RTE Vuser running on a VT terminal emulator. Text synchronization uses the **TE_wait_text** function. During script execution, the **TE_wait_text** function suspends script execution and waits for a specific string to appear in the terminal window before continuing with script execution. Text synchronization is useful with those applications in which the cursor does not consistently appear in a predefined area on the screen.

> **Note:** Although text synchronization is designed to be used with character mode (VT) terminals, it can also be used with IBM block-mode terminals. Do not use automatic text synchronization with block-mode terminals.

The syntax of the **TE_wait_text** function is:

```
int TE_wait_text (char * pattern, int timeout, int col1, int row1, int
col2, int row2,
                    int * retcol, int * retrow, char * match );
```

This function waits for text matching pattern to appear within the rectangle defined by col1, row1, col2, row2. Text matching the pattern is returned to **match**, and the actual row and column position is returned to **retcol** and **retrow**. If the **pattern** does not appear before the **timeout** expires, the function returns an error code. The **pattern** can include a regular expression. See the Function Reference for details on using regular expressions. Besides the **pattern** and **timeout** parameters, all the other parameters are optional.

If **pattern** is passed as an empty string, the function will wait for timeout if it finds any text at all within the rectangle. If there is no text, it returns immediately.

If the pattern does appear, then the function waits for the emulator to be stable (finish redrawing, and not display any new characters) for the interval defined by the TE_SILENT_SEC and TE_SILENT_MILLI system variables. This, in effect, allows the terminal to become stable and emulates a human user.

If the terminal does not become stable within the interval defined by TE_SILENT_TIMEOUT, script execution continues. The function returns 0 for success, but sets the TE_errno variable to indicate that the terminal was not silent after the text appeared.

To modify or retrieve the value of any of the TE_SILENT system variables, use the TE_getvar and TE_setvar functions. For more information, see the Function Reference (**Help > Function Reference**).

In the following example, the Vuser types in its name, and then waits for the application to respond.

```
/* Declare variables for TE_wait_text */
int ret_row;
int ret_col;
char ret_text [80];
/* Type in user name. */
TE_type ("John");
/* Wait for teller to respond. */
TE_wait_text ("Enter secret code:", 30, 29, 13, 1, 13, =;ret_col, =;ret_row,
            ret_text);
```

You can instruct VuGen to automatically generate **TE_wait_text** statements, and insert them into a script, while you record the script.

## Instruct VuGen to automatically generate and insert TE_wait_text statements while recording

1. Select **Vuser** > **Recording Options**. The Recording Settings dialog box appears.
2. Under **Generate Automatic Synchronization Commands**, select the **Prompt** check box, and then click **OK**.

The following is an example of a TE_wait_text statement that was automatically generated by VuGen. The function waits up to 20 seconds for the string "keys" to appear anywhere on the screen. Note that VuGen omits all the optional parameters when it generates a TE_wait_text function.

```
TE_wait_text("keys", 20);
```

### Wait for the terminal to be silent

In instances when neither cursor synchronization nor text synchronization are effective, you can use "silent synchronization" to synchronize the script. With "silent synchronization," the Vuser waits for the terminal emulator to be silent for a specified period of time. The emulator is considered to be silent when it does not receive any input from the server for a specified period of time.

> **Note:** Use silent synchronization only when neither cursor synchronization nor text synchronization are effective.

You use the **TE_wait_silent** function to instruct a script to wait for the terminal to be silent. You specify the period for which the terminal must be silent. If the terminal is silent for the specified period, then the **TE_wait_silent** function assumes that the application has stopped printing text to the terminal screen, and that the screen has stabilized.

The syntax of the function is:

```
int  TE_wait_silent (int  sec, int  milli, int  timeout );
```

The **TE_wait_silent** function waits for the terminal emulator to be silent for the time specified by sec (seconds) and milli (milliseconds). The emulator is considered silent when it does not receive any input from the server. If the emulator does not become silent (i.e. stop receiving characters) during the time specified by the time timeout variable, then the function returns an error.

For example, the following statement waits for the screen to be stable for three seconds. If after ten seconds, the screen has not become stable, the function returns an error.

```
TE_wait_silent (3, 0, 10);
```

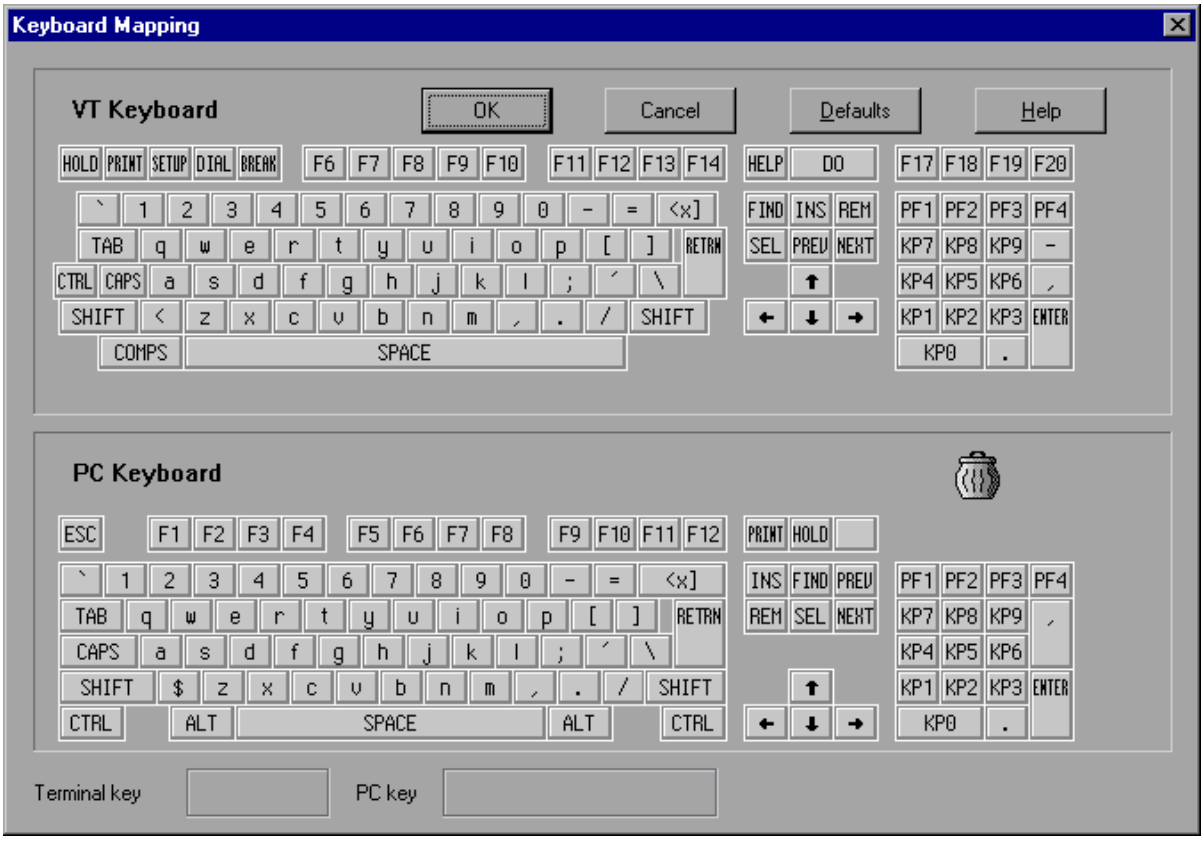
For more information, see the Function Reference (**Help > Function Reference**).

## Map Terminal Keys to PC Keyboard Keys

Because you are using a terminal emulator, you will be using a PC keyboard in place of a terminal keyboard. Many keys that are found on the terminal keyboard are not available on a PC keyboard. Examples of such keys are HELP, AUTHOR, and PUSH, which are found on the IBM 5250 keyboard. To successfully operate the terminal emulator and any associated application programs, you may have to map certain terminal keys to keys on the PC keyboard.

### Map a Terminal Key to a Key on the PC Keyboard

1. In the terminal emulator, select **Options** > **Keyboard Map**, or click the **Keyboard Mapping** button . The Keyboard Mapping dialog box opens.




2. Click the Keyboard **Mapping** button on the toolbar. To map a terminal key to a PC key, drag a key from the upper terminal keyboard to a PC key on the lower keyboard.

   You can click the Shift and/or Control keys on the upper keyboard to display additional key functions that can be viewed only by first selecting either of these keys. You can then drag the required key from the upper terminal keyboard to a key on the lower PC keyboard.

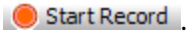
   To cancel a definition, drag the PC key definition to the wastebasket. This restores the default function of the PC key.

   To restore the default mappings, click **Defaults**.

## Record RTE Vuser Scripts

You use VuGen to record RTE Vuser scripts. VuGen uses the PowerTerm terminal emulator to emulate a wide variety of terminal types.

This task describes how to record RTE Vuser scripts. This procedure differs from the general recording procedure in "Recording" on page 154.

1. **Record the terminal setup and connection**
   a. Open an existing RTE Vuser script, or create a new one.
   b. In the **Sections** box, select the **vuser_init** section to insert the recorded statements.
   c. In the Vuser script, place the cursor at the location where you want to begin recording.
   d. Click the **Start Record** button 

. The PowerTerm main window opens.
   e. From the PowerTerm menu bar, select **Terminal** > **Setup** to display the Terminal Setup dialog box.
   f. Select the type of emulation from the VT Terminal and IBM Terminal types, and then click **OK**.

> **Note:** Select an IBM terminal type to connect to an AS/400 machine or an IBM mainframe; select a VT terminal type to connect to a Linux workstation.

   g. Select **Communication** > **Connect** to display the Connect dialog box.
   h. Under **Session Type**, select the type of communication to use.
   i. Under **Parameters**, specify the required options. The available parameters vary depending on the type of session that you select. For details on the parameters, click **Help**.

> **Tip:** Click **Save As** to save the parameter-sets for re-use in the future. The parameter-sets that you save are displayed in the Sessions List box.

   j. Click **Connect**. PowerTerm connects to the specified system, and VuGen inserts a **TE_connect** function into the script, at the insertion point. The **TE_connect** statement has the following form:

```
/* *** The terminal type is VT 100. */
TE_connect(
    "comm-type = telnet;"
    "host-name = alfa;"
    "telnet-port = 992;"
    "terminal-id = ;"
    "set-window-size = true;"
    "security-type = ssl;"
    "ssl-type = tls1;"
    "terminal-type = vt100;"
    "terminal-model = vt100;"
    "login-command-file = ;"
    "terminal-setup-file = ;"
    , 60000);
if (TE_errno != TE_SUCCESS)
    return -1;
```

   The inserted **TE_connect** statement is followed by an if statement that checks whether or not the **TE_connect** function succeeds during replay**.**

> **Note:** Do not record more than one connection to a server **(TE_connect)** in a Vuser script.

2. **Record typical user actions**

   After recording the setup procedure, you perform typical user actions or business processes. You record these processes into the **Actions** section of the Vuser script. Only the **Actions** section of a Vuser script is repeated when you run multiple iterations of the script.

   When recording a session, VuGen records the text strokes and not the text. Therefore, it is not recommended that you copy and paste commands into the PowerTerm window—instead, type them in directly.

   a. Select the **Actions** section in the **Section** box.

   b. Proceed to perform typical user actions in the terminal emulator. VuGen generates the appropriate statements, and inserts them into the Vuser script while you type. If necessary, you can edit the recorded statements while you record the script.

3. **Record the log-off procedure**

   a. Make sure that you have performed and recorded the typical user actions as described in the previous section.

   b. In the VuGen main window, click **vuser_end** in the **Section** box.

   c. Perform the log off procedure. VuGen records the procedure into the **vuser_end** section of the script.

   d. Click **Stop Recording** on the Recording toolbar. The main VuGen window displays all the recorded statements.

   e. Click **Save** to save the recorded session. After recording a script, you can manually edit it in VuGen's main window.

## Implement Continue on Error

To configure the Continue on Error functionality in RTE Scripts:

- To continue running the script on error, insert the following function:

  **TE_setvar(TE_IGNORE_ERRORS, 1)**

- To restore the default behavior of failing the script on error, insert the following function:

  **TE_setvar(TE_IGNORE_ERRORS, 0)**

## Troubleshooting and Limitations - RTE

This section describes troubleshooting and limitations for RTE Vusers.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

### IP Spoofing

IP spoofing is not supported for RTE Vusers.

### Disconnection Failures

When running an RTE script from the Controller, it may hang in the **Gradual Exiting** stage. A possible reason may be that the terminal session tried to disconnect but did not finish disconnecting before a new connect command arrived. As a result, the scenario cannot exit properly.

**Possible workaround:** In scripts created with versions of LoadRunner prior to 12.50, **TE_disconnect** was not automatically recorded. Manually add **TE_disconnect** functions to older RTE scripts.

### Initialization Failures

When running an RTE script from the Controller, it may fail during initialization with an error message like this:

**"Warning: Extension rterun32.dll reports error -1 on call to function ExtPerThreadInitialize. Error: Vuser failed to initialize extension rterun32.dll."**

**Possible solution:** This could be an "Out of Memory" issue. This can be fixed by increasing the Desktop Heap. In this case, change the last value of the **SharedSection** parameter in **[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\SubSystems\Windows]**.

For more details on how to fix "Out of memory" issues, see https://support.microsoft.com/en-us/kb/126962.

# SAP Protocols

## Selecting an SAP Protocol Type

- To test the SAP GUI user operating only on the client, use the SAP GUI Vuser type.
- To test a SAP GUI user that also uses a Web browser, use the SAP-Web protocol.

To record a SAP GUI session that uses browser controls, create a multi-protocol Vuser script with the SAP GUI and SAP-Web protocols. This allows VuGen to record Web-specific functions when encountering the browser controls. This will not work if you attempt to combine SAP GUI and Web protocols.

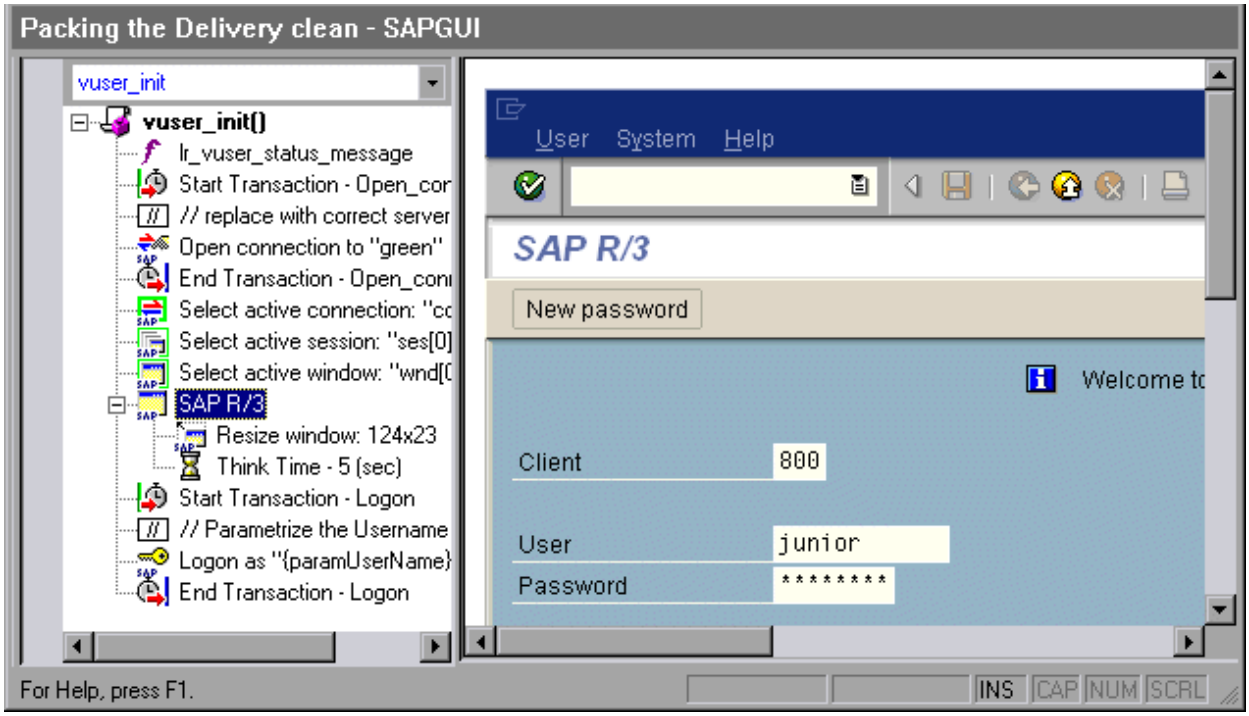
Before recording a session, verify that your modules and client interfaces are supported by VuGen. The following table describes the SAP client modules for SAP Business applications and the relevant tools:

| SAP module | VuGen support |
|---|---|
| **SAP Web Client or mySAP.com** | Use the SAP-Web protocol. |
| **SAP GUI for Windows** | Use the SAP GUI protocol. This also supports APO module recording (requires patch level 24 for APO 3.0 for SAP 6.20). |
| **SAP GUI for Java** | This client is not supported. |

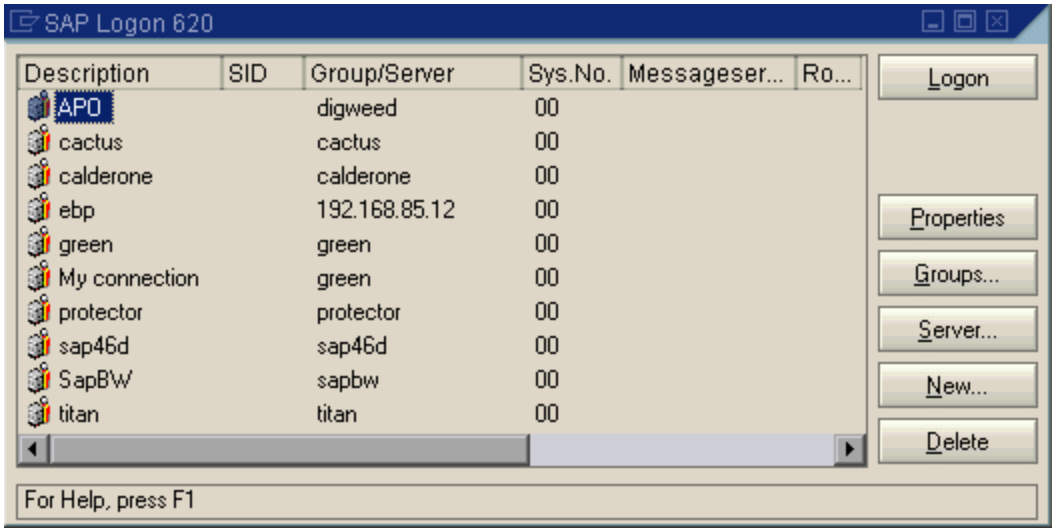
## SAP GUI Protocol

The SAP GUI Vuser script typically contains several SAP transactions which make up a business process. A business process consists of functions that emulate user actions. Open the **Step Navigator** to see each user action as a Vuser script step.

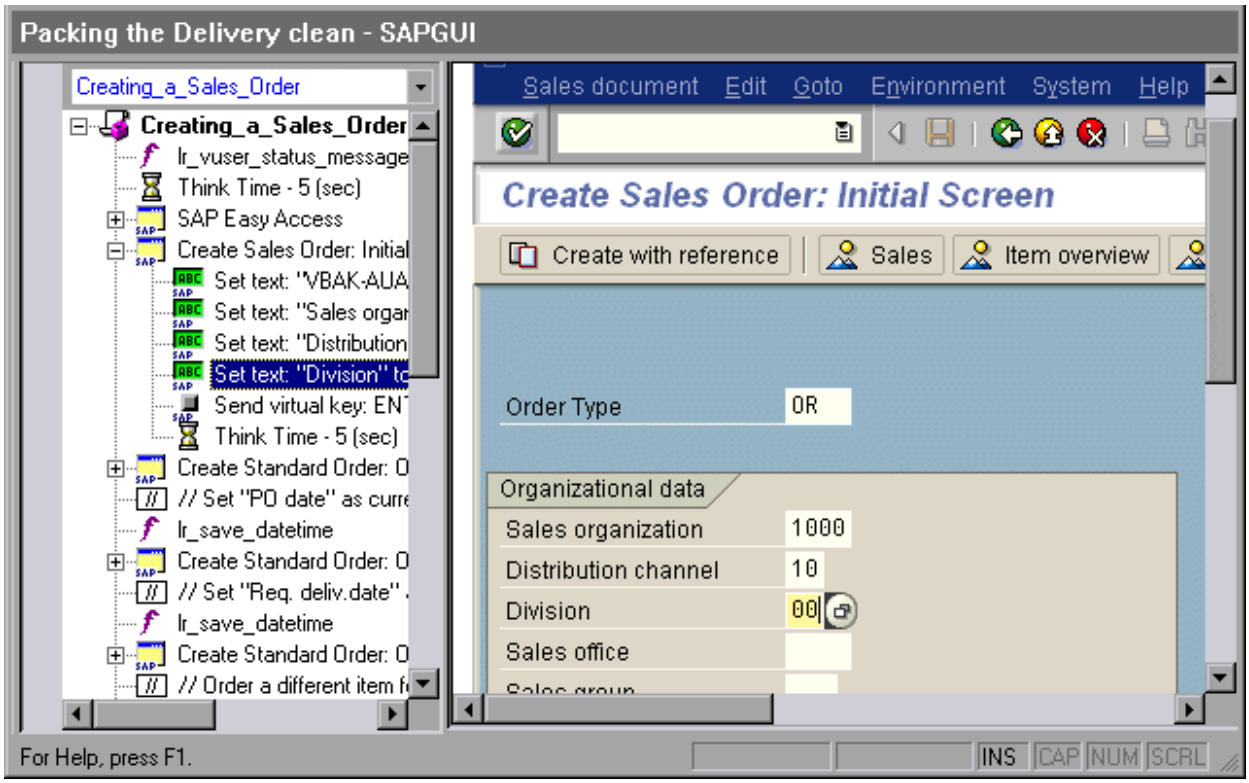
The following example shows a typical recording of a SAP GUI client. The first section, **vuser_init**, contains the opening of a connection and a logon.



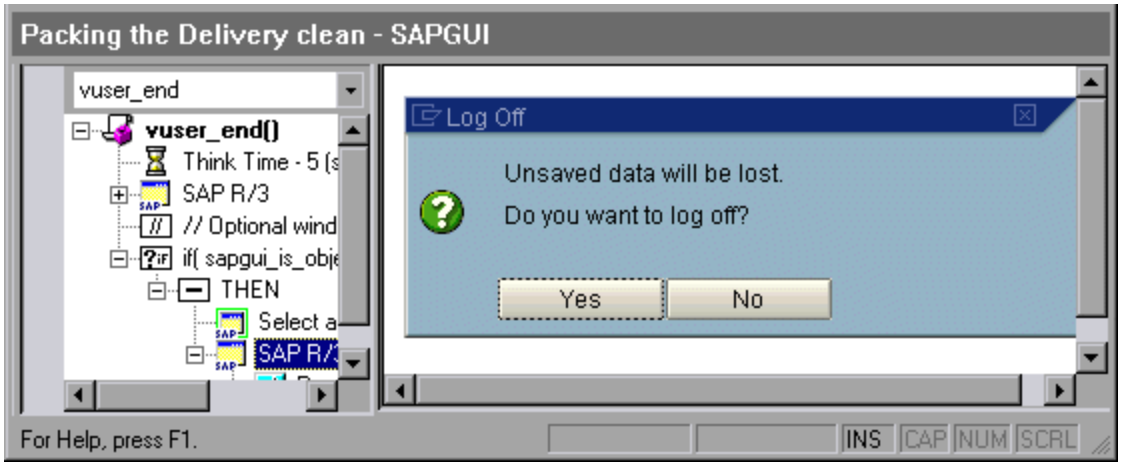

**Note:** The Open Connection step uses one of the connection names in the SAP Logon **Descriptions** list. If the specified connection name is not in the list, the Vuser looks for a server with that name.




In the following section, the functions emulate typical user operations such as menu selection and the setting of a check box.




The final section, **vuser_end**, illustrates the logoff procedure.




When recording a multi- protocol script for both SAP GUI and Web, VuGen generates steps for both protocols. In the Script view, you can view both **sapgui** and **web** functions.

The following example illustrates a multi-protocol recording in which the SAP GUI client opens a Web control. Note the switch from **sapgui** to **web** functions.

**Example:**

```
sapgui_tree_double_click_item("Use as general WWW browser, REPTITLE",
    "shellcont/shell",
    "000732",
    "REPTITLE",
    BEGIN_OPTIONAL,
        "AdditionalInfo=sapgui1020",
    END_OPTIONAL);
...
sapgui_set_text("",
    "http:\\\\yahoo.com",
    "usr/txtEDURL",
    BEGIN_OPTIONAL,
        "AdditionalInfo=sapgui1021",
    END_OPTIONAL);
...
web_add_cookie("B=7pt5cisv1p3m2=;b=2; DOMAIN=www.yahoo.com");
web_url("yahoo.com",
        "URL=http://yahoo.com/",
        "Resource=0",
        "RecContentType=text/html",
        "Referer=",
        "Snapshot=t1.inf",
        "Mode=HTML",
        EXTRARES,

"URL=
http://srd.yahoo.com/hpt1/ni=17/ct=lan/sss=1043752588/t1=1043752575385/d1=1251
          /d2=1312/d3=1642/d4=4757/0.4097009487287739/*1",
```

```
        "Referer=http://www.yahoo.com/", ENDITEM,
    LAST);
```

## SAP Web Protocol

The SAP-Web Vuser script typically contains several SAP transactions which make up a business process. The business process consists of functions that emulate user actions. For information about these functions, see the Web functions in the Function Reference (**Help > Function Reference**).

> **Note:** You can generate a SAP - Web Vuser script by analyzing an existing network traffic file (capture file). This method may be useful for creating Vuser scripts that emulate activity on mobile applications. For details, see "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723.

**Example:**

The following example shows a typical recording for an SAP Portal client:

```
vuser_init()
{
    web_reg_find("Text=SAP Portals Enterprise Portal 5.0",
        LAST);
    web_set_user("junior{UserNumber}",
        lr_unmask("3ed4cfe457afe04e"),
        "sonata.mylab.com:80");
    web_url("sapportal",
        "URL=http://sonata.mylab.com/sapportal",
        "Resource=0",
        "RecContentType=text/html",
        "Snapshot=t1.inf",
        "Mode=HTML",
        EXTRARES,
        "Url=/SAPPortal/IE/Media/sap_mango_polarwind/images/header/branding_
image.jpg",

"Referer=http://sonata.mylab.com/hrnp$30001/sonata.mylab.coml:80/Action/26011
[header]"
           , ENDITEM,
        "Url=/SAPPortal/IE/Media/sap_mango_polarwind/images/header/logo.gif",

"Referer=http://sonata.mylab.com/hrnp$30001/sonata.mylab.com:80/Action/26011
[header]",
             ENDITEM,
...
        LAST);
```

The following section illustrates an SAP Web and SAP GUI multi-protocol recording in which the Portal client opens an SAP control. Note the switch from **web_xxx** to **sapgui_xxx** functions.

```
web_url("dummy",
        "URL=http://sonata.mylab.com:1000/hrnp$30000/sonata.mylab.com:
          1000/Action/dummy?PASS_PARAMS=YES=;dummyComp=dummy=;
Tcode=VA01=;draggable=0=;CompFName=VA01=;Style=sap_mango_polarwind",
        "Resource=0",
        "RecContentType=text/html",
        "Referer=http://sonata.mylab.com/sapportal",
        "Snapshot=t9.inf",
        "Mode=HTML",
        LAST);
    sapgui_open_connection_ex(" /H/Protector/S/3200 /WP",
        "",
        "con[0]");
    sapgui_select_active_connection("con[0]");
    sapgui_select_active_session("ses[0]");
    /*Before running script, enter password in place of asterisks in logon function*/
    sapgui_logon("JUNIOR{UserNumber}",
        "ides",
        "800",
        "EN",
        BEGIN_OPTIONAL,
            "AdditionalInfo=sapgui102",
        END_OPTIONAL);
```

## Replaying SAP GUI Optional Windows

When working with SAP GUI Vuser Scripts, you may encounter optional windows in the SAP GUI client—windows that were present during recording, but do not exist during replay. If you try to replay your recorded script as is, it will fail when it attempts to find the missing windows.

VuGen's optional window mechanism performs the actions on a window only after verifying that it exists. The Vuser checks if the window indicated in the **Select active window** step exists. If the window is found during replay, it performs the actions as they were recorded in the script. If it does not exist, the Vuser ignores all window actions until the next **Select active window** step. Note that only SAP GUI steps (beginning with a **sapgui** prefix) are ignored.

To use this feature, in Tree view select the appropriate Select Active Window step and select **Run steps for window only if it exists** from the right-click menu.

To disable this feature and attempt to run these steps at all times, regardless of whether the Vuser finds the window or not, select **Always run steps for this window** from the right-click menu.

## Configure the SAP Environment

This task describes configure and verify the SAP environment for use with VuGen.

VuGen support for the SAP GUI for Windows client, is based on SAP's Scripting API. This API allows Vusers to interact with the SAP GUI client, receive notifications, and perform operations.

The Scripting API is available only in recent versions of the SAP Kernel. In kernel versions that support scripting, the option is disabled by default. In order to use VuGen, first make sure that the SAP servers support the Scripting API, and enable the API on both the server and clients. For more information and to download patches, see the SAP OSS note #480149.

## Use IPV6 with a SAP installation

Create an `SAP_IPV6_ACTIVE` environment variable and set the value to 1.

## Check that SAPGUI Scripting API is enabled

Run the **VerifyScripting.exe** file from the **Additional Components\SAP_Tools\VerifySAPGUI** folder. For details, see "Additional Components" on page 1566. For more information, see the file **VerifyScripting.htm** provided with this utility.

## Check the Patch Level

1. Open the SAP GUI logon window. Click the top left corner of the SAP Logon dialog box and select **About SAP Logon** from the menu.
2. The SAP version information dialog box opens. Verify that the Patch Level entry is 32 or higher.

## Check the Kernel Patch Level

1. Log in to the SAP system.
2. Select **System > Status**.
3. Click the **Other kernel information** button.
4. In the **Kernel Information** section, check the value of the **Sup. Pkg. lvl.**

   The level must be greater than the level listed in the following chart depending on the SAP version you are using.

| Software Component | SAP Release | Kernel Patch Level |
|---|---|---|
| SAP_APPL | 31I | Kernel 3.1I level 650 |
| SAP_APPL | 40B | Kernel 4.0B level 903 |
| SAP_APPL | 45B | Kernel 4.5B level 753 |
| SAP_BASIS | 46B | Kernel 4.6D level 948 |
| SAP_BASIS | 46C | Kernel 4.6D level 948 |
| SAP_BASIS | 46D | Kernel 4.6D level 948 |

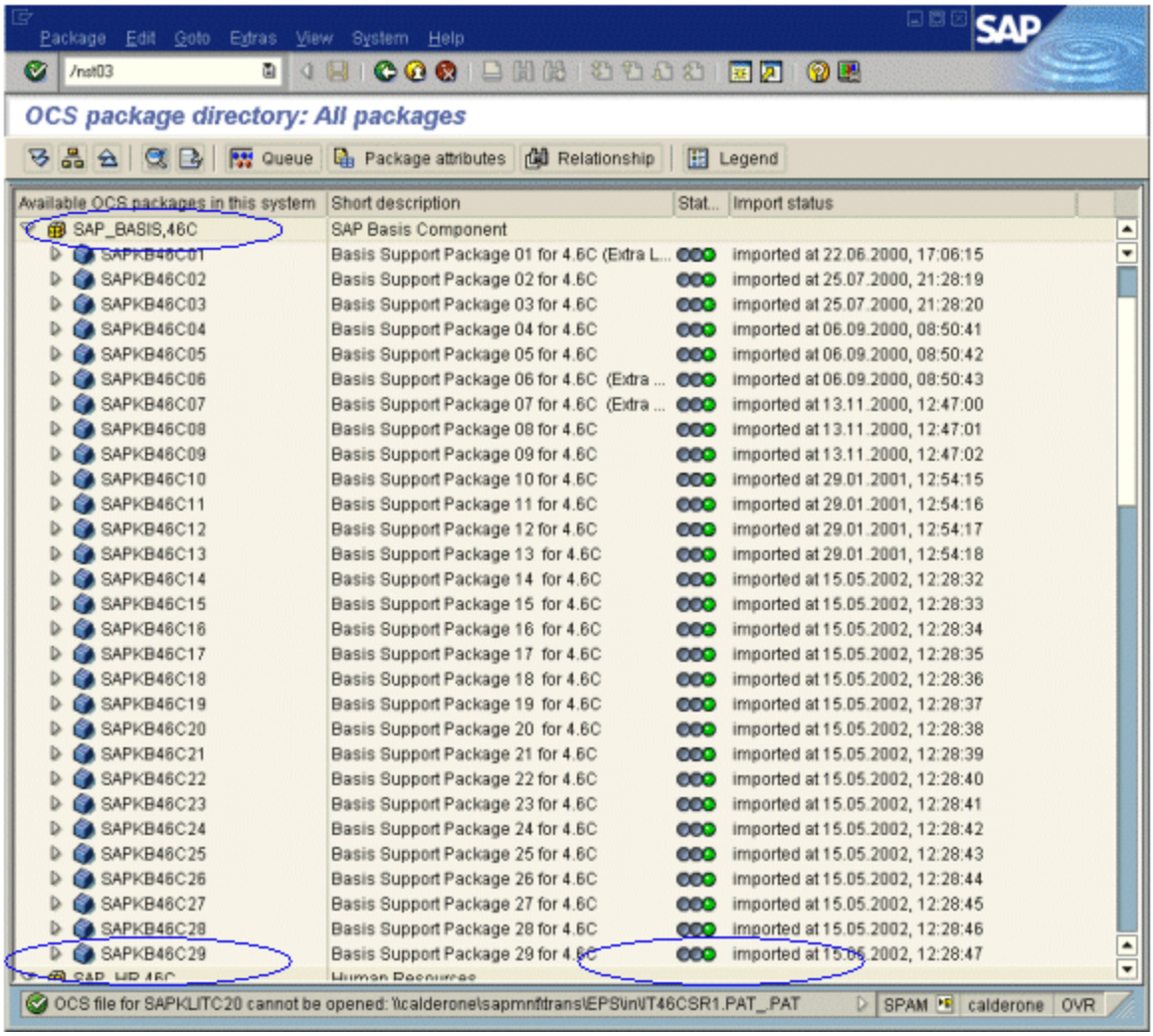
| | | |
|---|---|---|
| SAP_BASIS | 610 | Kernel 6.10 level 360 |

## Check the R/3 Support Packages

1. Log on to the SAP system and run the SPAM transaction.
2. In the **Directory** section, select **All Support Packages**, and click the **Display** button.
3. Verify that the correct package is installed for your version of SAP according to the table below.

| Software Component | Release | Package Name |
|---|---|---|
| SAP_APPL | 31I | SAPKH31I96 |
| SAP_APPL | 40B | SAPKH40B71 |
| SAP_APPL | 45B | SAPKH45B49 |
| SAP_BASIS | 46B | SAPKB46B37 |
| SAP_BASIS | 46C | SAPKB46C29 |
| SAP_BASIS | 46D | SAPKB46D17 |
| SAP_BASIS | 610 | SAPKB61012 |

If the correct version is installed, a green circle appears in the Status column.

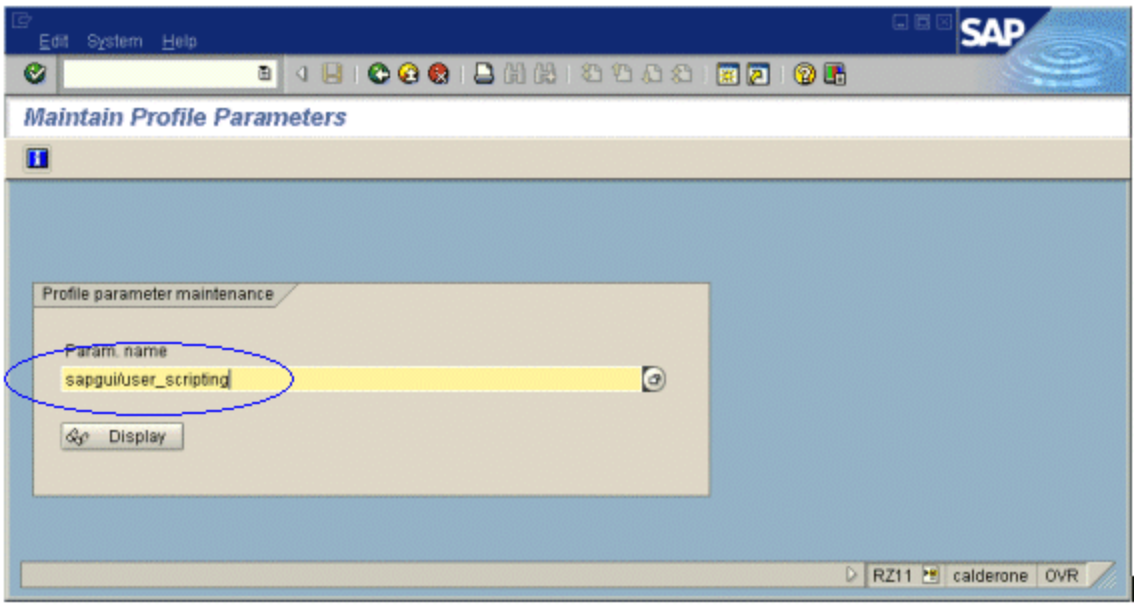



If you do not have the OCS package installed, download it the from the www.sap.com Web site and install it. For more information, see the SAP OSS note #480149.

## Enable scripting on the SAP Application Server

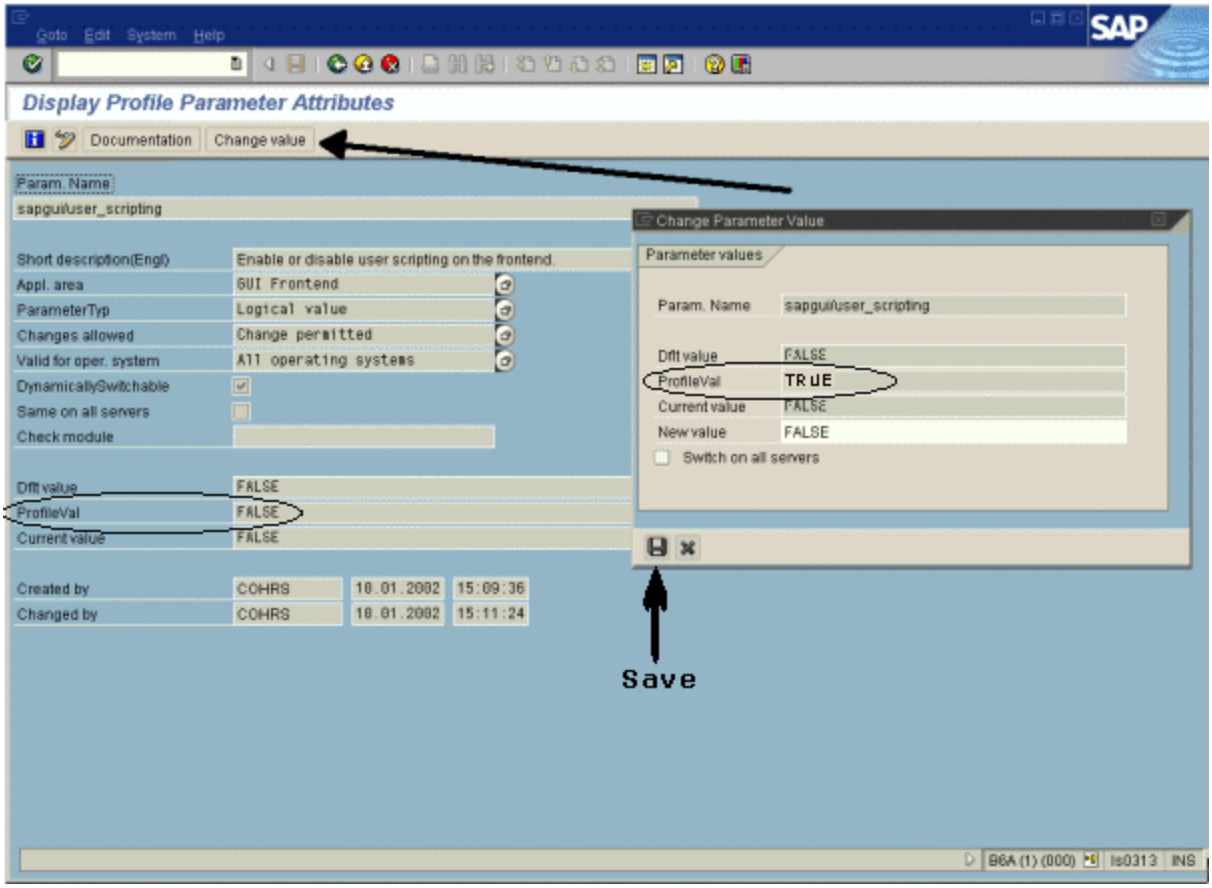
A user with administrative permissions enables scripting by setting the **sapgui/user_scripting** profile parameter to **TRUE** on the application server. To enable scripting for all users, set this parameter on all application servers. To enable scripting for a specific group of users, only set the parameter on application servers with the desired access restrictions. The following steps describe how to change the profile parameter.

1. Open transaction **rz11.** Specify the parameter name **sapgui/user_scripting** and click **Display**. The Display Profile Parameter Attributes window opens.




If **Parameter name is unknown** appears in the status bar, this indicates that you are missing the current Support Package. Import the Support Package that corresponds to the SAP BASIS and kernel versions of the application server, as described in the steps above.

2. If **Profile Val** is FALSE, you need to modify its value. Click the **Change value** button in the toolbar. The Change Parameter Value window opens. Enter TRUE in the **ProfileVal** box and click the **Save** button.




When you save the change, the window closes and **ProfileVal** is set to TRUE.

3. Restart the application server to apply your changes.

   If the updated **ProfileVal** did not change, even after restarting the server, then the kernel of the application server is outdated. Import the required kernel patch, as specified in the steps above.

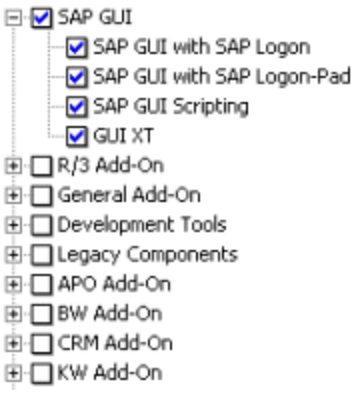
> **Note:** The Profile Value may be dynamically activated in the following kernel versions, using transaction rz11, without having to restart the application server.
>
> | Release | Kernel Version | Patch Level |
> |---|---|---|
> | 4.6B, 4.6C, 4.6D | 4.6D | 972 |
> | 6.10 | 6.10 | 391 |
> | 6.20 | all versions | all levels |

## Enable scripting on SAP GUI client

To allow VuGen to run scripts, you must also enable scripting on the SAP GUI client. You should also configure the client not to display certain messages, such as when a connection is established, or when a

script is attached to the GUI process. The following steps describe how to configure the SAP GUI client to work with .

1. **During installation.** While installing the SAP GUI client, enable the **SAP GUI Scripting** option.




2. **After installation.** Suppress warning messages. Open the Options dialog box in the SAP GUI client. Select the **Scripting** tab and clear the following options:
   - **Notify when a script attaches to a running GUI**
   - **Notify when a script opens a connection**

   You can also prevent these messages from popping up by setting the values for **WarnOnAttach** and **WarnOnConnection** to 0 in the following registry key:

   `HKCU\SOFTWARE\SAP\SAPGUI Front\SAP Frontend Server\Security.`

### Examine the hierarchy of GUI Scripting objects - Optional

The **SAPGUI Spy** utility, is part of the "Additional Components" on page 1566. It examines the hierarchy of GUI Scripting objects, in open windows of **SAPGUI Client for Windows**.

### To install the SAPGUI Spy component:

1. Copy the files **mscomctl.ocx**, **Msflxgrd.ocx** and **msvbvm60.dll** from the **SAP_Tools\SapGuiSpy\System32VBdlls** folder to your Windows **System32** folder.
2. Register the files.
   a. In the Windows **Run** box, enter: `regsvr32 <File name>`
   b. Run the **SapSpy.exe** file from the **SAP_Tools\SapGuiSpy** folder.

## Record SAP GUI Scripts

The following steps describe some prerequisites to recording a SAP GUI script.

#### Configure the application server for scripting

As a security precaution, scripting is disabled by default. In order to record, you need to enable scripting

on the application server. From the RZ11 transaction, set the following profile parameters as follows:

- sapgui/user_scripting TRUE
- sapgui/user_scripting_force_notification FALSE
- sapgui/user_scripting_set_readonly FALSE
- sapgui/user_scripting_disable_recording FALSE

### Close SAPLogon application when recording with Multi

When recording a multi-protocol script in which the SAP GUI client contains Web controls, close the SAPLogon application before recording.

### Use Modal dialog boxes for F1 Help

Instruct the SAP GUI client to open the F1 help in a modal dialog box as follows:

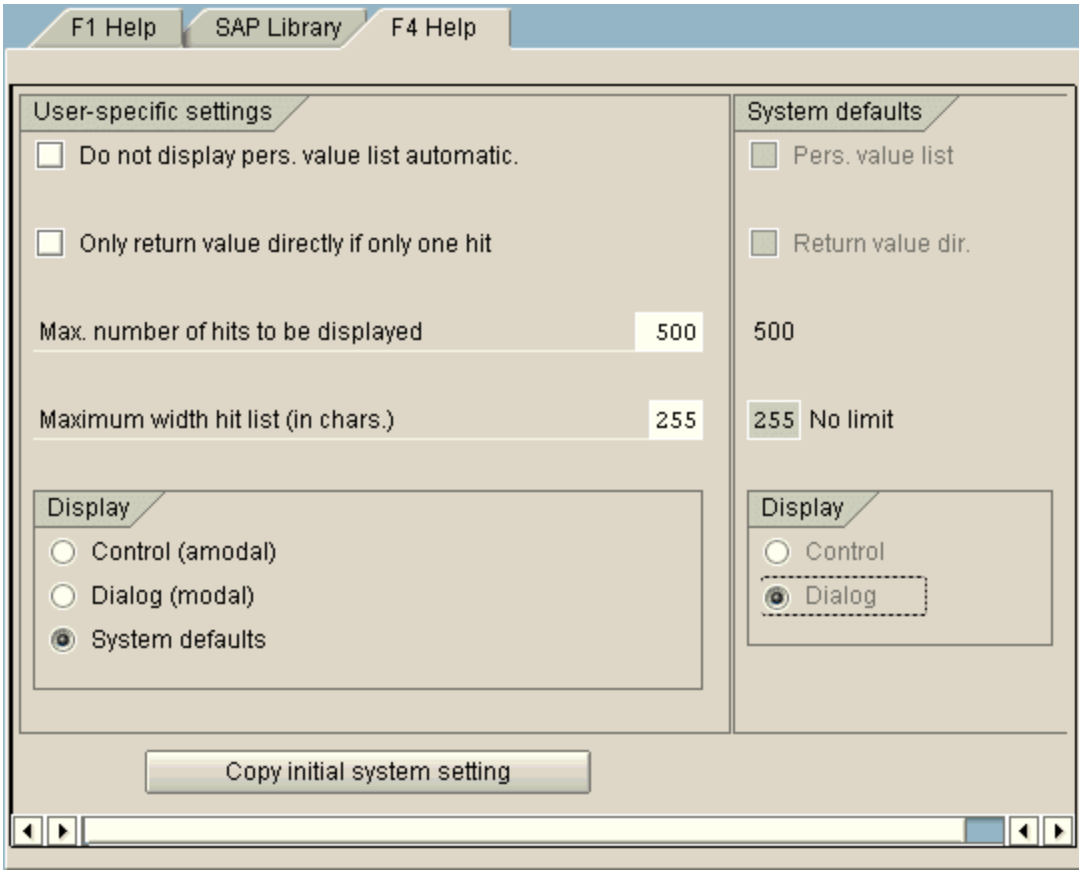
1. Select **Help > Settings**.
2. Click the **F1 Help** tab.
3. Select **in modal dialog box** in the Display section.

### Use Modal dialog boxes for F4 Help

> **Note:** This procedure can only be performed by the administrator.

Instruct the SAP GUI client to open the F4 help in a modal dialog box:

1. Make sure that all users have logged off from the server.
2. Select **Help > Settings**. Click the **F4 Help** tab.




3. In the Display section, select **System defaults**.
4. In the Display portion of the System defaults section, select **Dialog**.
5. Save the changes by clicking **Copy initial system setting**.
6. Verify that the status bar displays the message **Data was saved**.
7. Close the session and restart the service through the SAP Management Console.

## Replay SAP GUI Scripts

The following steps describe prerequisites to replaying SAPGUI scripts.

### Replace Masked Password

Replace the masked password in the **sapgui_logon** function generated during recording, with the real password. It is the second argument of the function, after the following user name

```
sapgui_logon("user", "pswd", "800", "EN");
```

For additional security, you can mask the password within the code. Right-click in the password text (the actual text, not *****) and select **Mask String**. VuGen inserts an **lr_unmask** function at the location of the password as follows:

```
sapgui_logon("user", lr_unmask("3ea037b758"), "800", "EN");
```

**Display SAP GUI During Replay (optional)**

When running a script for the first time, configure VuGen to show the SAP GUI user interface during replay, in order to see the operations being performed through the UI. Select **Replay > Runtime Settings > SAPGUI > General** node and select **Show SAP Client During Replay**. During a load scenario, disable this option, since it uses a large amount of system resources in displaying the UI for multiple Vusers.

# Run SAP GUI Scripts from the Controller

The following steps describe tips for running SAP GUI scripts in a LoadRunner scenario.

## LoadRunner Controller Settings

When working with a LoadRunner scenario, set the following values when running your script in a load test configuration:

- **Ramp-up.** One by one (to insure proper logon) in the Scheduler.
- **Think time.** Random think time in the runtime settings.

## Make Sure the Agent is Running in Process Mode

Make sure that the LoadRunner (or Performance Center) Remote Agent is running in Process mode. Service mode is not supported.

To check this, move your mouse over the agent's icon in the Windows task bar area, and read the description. If the description reads Load Testing Agent Service, it is running as a service.

The following steps describe how to restart the agent as a process.

1. Stop the agent. Right-click the LoadRunner Agent icon and select **Close**.
2. Run **magentproc.exe**, located in the **launch_service\bin** folder, under the LoadRunner installation.
3. To make sure that the correct Agent is launched the next time you start your machine, change the Start type of the Agent Service from Automatic to Manual. Then add a shortcut to **magentproc.exe** to the Windows Startup folder.
   - **Terminal Sessions.** Machines running SAP GUI Vusers may be limited in the number of Vusers that can run, due to the graphic resources available to that machine. To increase the number of Vusers per machine, open additional terminal server sessions on the Load Generator machines. Select **Agent Configuration** from **Start > All Programs > <product_name> > Advanced Settings**, and select the **Enable Terminal Service** option. You specify the number of terminal sessions in

the Load generator machine properties. For more information, see Configuring Terminal Services Settings.

> **Note:** When the LoadRunner Agent is running in a terminal session, and the terminal session's window is minimized, no snapshots will be captured on errors.

## Enhance SAP GUI Scripts

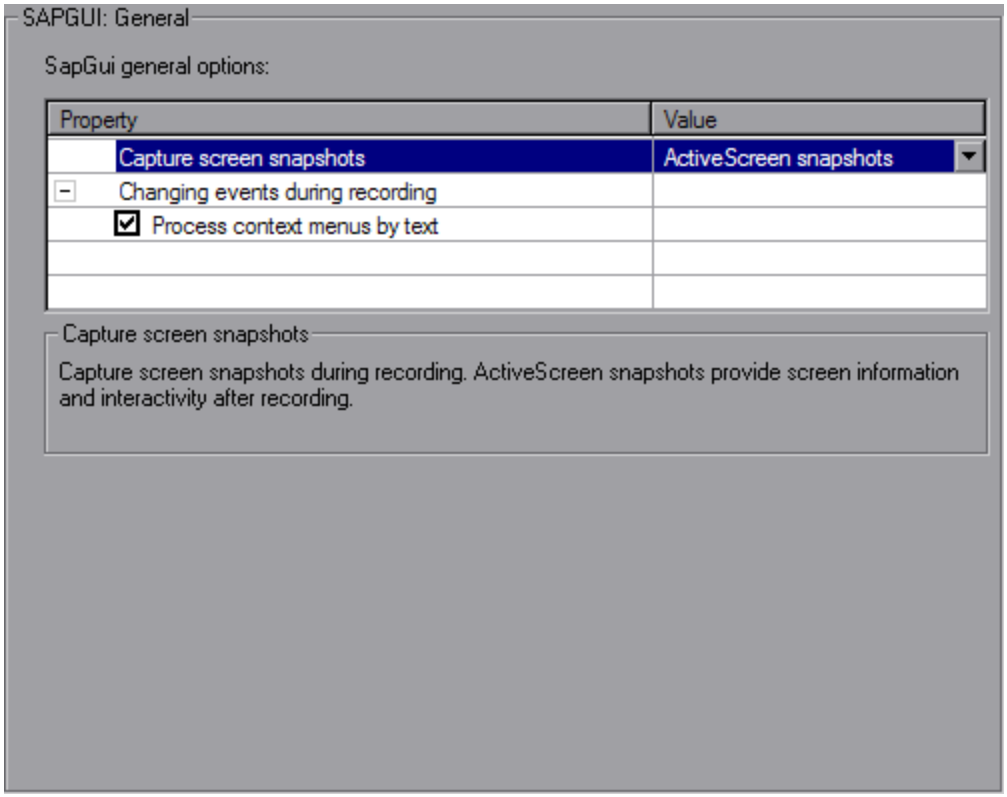
The following steps describe how to enhance SAP GUI protocol scripts.

### Insert Steps Interactively into a SAP GUI Script

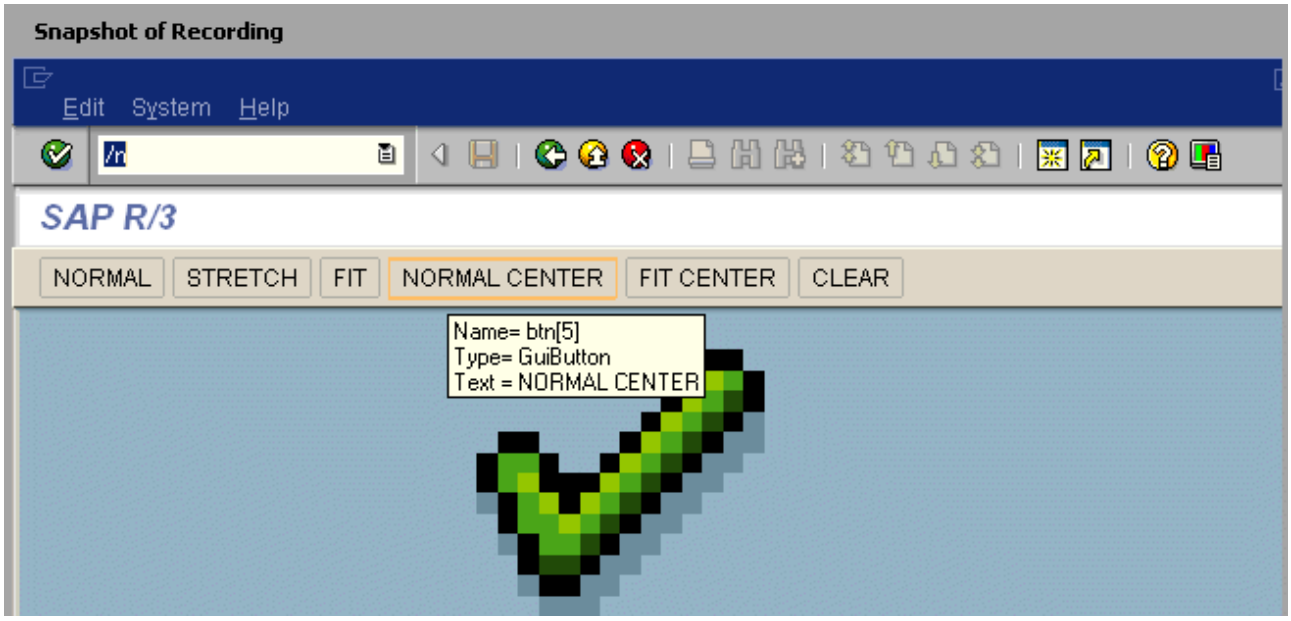
After recording, you can manually add steps to the script in either the **Editor** or **Step Navigator**. In addition to manually adding new functions, you can add new steps interactively for SAP GUI Vusers, directly from the snapshot. Using the right-click menu, you can add object-related steps.

When adding a step from within a snapshot, VuGen uses the Active Screen capability and determines the ID of each object in the SAP GUI client window (unless you disabled Active Screen snapshots in the "SAPGUI > General Recording Options" on page 220). The following steps describe how to insert a step interactively for a specific object.

1. Verify that you recorded the script when Active Screen snapshots were selected in the SAPGUI General node of the Recording Options (enabled by default).

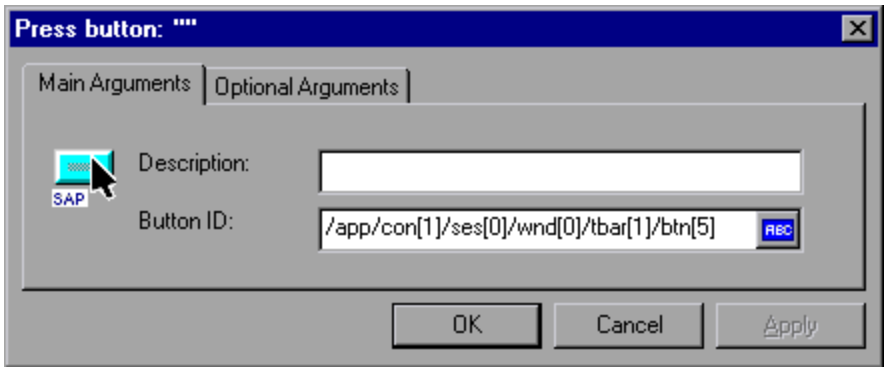


2. Click within the Snapshot pane.
3. Move the mouse over the object for which you want to add a function. Make sure that VuGen recognizes the object and encloses it with a box.




4. Right-click the object, click **Insert New Step**, and then select a step from the list of steps that are available for the object.

   The step's Properties dialog box opens, with the Control ID of the object when relevant. For example, if you add a **Press Button** step, for the normal center button as shown above, the Properties box displays the following ID:




5. Enter a name for the object in the **Description** box. Click **OK**. VuGen inserts the new step after the selected step.

> **Note:** You can get the Control ID of the object for the purpose of pasting it into a specific location. To do this, select Copy **Control ID** from the right-click menu. You can past it into a Properties box or directly into the code from the Script view.

### Add Verification Functions

When working with optional or dynamic windows or frames, you can use verification functions to determine if the window or object is available. An optional window is a window that does not consistently open during the SAP session. This function allow the Vuser script to continue running even if an optional window opens or an exception occurs.

The first example checks if a window is available. If the window is available, the Vuser closes it before continuing.

```
if (!sapgui_is_object_available("wnd[1]"))
        sapgui_call_method("{ButtonID}",
            "press",
            LAST,
            AdditionalInfo=info1011");
sapgui_press_button(.....)
```

The next example illustrates a dynamic object in the ME51N transaction. The Document overview frame is optional, and can be opened/closed by the **Document overview on/off** button.

The code checks the text on the Document overview button. If the text on the button shows Document overview on, we click the button to close the Document overview frame.

```
    if(sapgui_is_object_available("tbar[1]/btn[9]"))
    {
        sapgui_get_text("Document overview on/off button",
            "tbar[1]/btn[9]",
            "paramButtonText",
            LAST);
        if(0 == strcmp("Document overview off", lr_eval_string("{paramButtonText}")))
            sapgui_press_button("Document overview off",
                "tbar[1]/btn[9]",
                BEGIN_OPTIONAL,
                    "AdditionalInfo=sapgui1013",
                END_OPTIONAL);
    }
```

### Retrieve Information

When working with SAGUI Vusers, you can retrieve the current value of a SAP GUI object using the **sapgui_get_<xxx>** functions. You can use this value as input for another business process, or display it in the output log.

The following example illustrates how to save part of a status bar message in order to retrieve the order number.

1. Navigate to the point where you want to check the status bar text, and select **Insert New Step.** Select the **sapgui_status_bar_get_type** function. This verifies that the Vuser can successfully

retrieve text from the status bar.

2. Insert an **if** statement that checks if the previous statement succeeded. If so, save the value of the argument using **sapgui_status_bar_get_param**.

   This **sapgui_status_bar_get_param** function saves the order number into a user-defined parameter. In this case, the order number is the second index of the status bar string.

```
sapgui_press_button("Save (Ctrl+S)",
    "tbar[0]/btn[11]",
    BEGIN_OPTIONAL,
        "AdditionalInfo=sapgui1038",
    END_OPTIONAL);
sapgui_status_bar_get_type("Status");
if(0==strcmp(lr_eval_string("{Status}"),"Success"))
  sapgui_status_bar_get_param("2", " Order_Number ");
```

   During test execution, the Execution log indicates the value and parameter name:

```
Action.c(240): Pressed button " Save (Ctrl+S)"
Action.c(248): The type of the status bar is "Success"
Action.c(251): The value of parameter 2 in the status bar is "33232"
```

### Save Date Information

When creating scripts that use dates, your script may not run properly. For example, if you record the script on June 2, and replay it on June 3, the date fields will be incorrect. Therefore, you need to save the date to a parameter during text execution, and use the stored value as input for other date fields. To save the current date or time during script execution, use the **lr_save_datetime** function. Insert this function before the function requiring the date information. Note that the format of the date is specific to your locale. Use the relevant format within the **lr_save_datetime** function. For example, for month.day.year, specify "`%m.%d.%Y`".

In the following example, **lr_save_datetime** saves the current date. The **sapgui_set_text** function uses this value to set the delivery date for two days later.

```
lr_save_datetime("%d.%m.%Y", DATE_NOW + (2 * ONE_DAY),
             "paramDateTodayPlus2");
sapgui_set_text("Req. deliv.date",
        "{paramDateTodayPlus2}",
        "usr/ctxtRV45A-KETDAT",
        BEGIN_OPTIONAL,
            "AdditionalInfo=sapgui1025",
        END_OPTIONAL);
```

## Additional SAP Resources

For more information, see the SAP website at www.sap.com or one of the following locations:

- **SAP Notes**- https://websmp103.sap-ag.de/notes

  Note #480149: New profile parameter for user scripting on the front end

  Note #587202: Drag =; Drop is a known limitation of the SAP GUI interface

- **SAP Patches** - https://websmp104.sap-ag.de/patches

  SAP GUI for Windows - SAP GUI 6.20 Patch (the lowest allowed level is 32)

## Troubleshooting and Limitations for SAP

This section describes troubleshooting and limitations for SAP GUI and SAP-Web protocols.

### General SAP GUI limitations

- For LoadRunner users: If a business process in SAP GUI includes selecting an item from a Combo-list, the business process may not run properly in LoadRunner.

  **Workaround:** Add the path of the SAP GUI installation folder to the Windows PATH environment variable.

- For LoadRunner users: When replaying a script on Windows 10 or Windows 2016, if a business process in SAP GUI (7.30, 7.40, 7.50) includes accessing a link, the business process may not run properly in LoadRunner versions 12.55 and higher. This error is generated by SAP.

  **Workaround:**

  a. Add the path of the SAP GUI installation folder (for example, C:\Program Files (x86)\SAP\FrontEnd\SAPgui) to the Windows PATH environment variable, and restart the machine.

  b. Navigate to the above SAP GUI installation folder (...\SAPgui). Copy the **guirfc.dll** and paste into the **C:\Windows\SysWOW64** folder.

  c. Close SAP GUI and check that the SAPLogon application is not running.

- Recording is not supported for the **SAP GUI Security** dialog box.
- Recording is not supported for standard Windows dialog boxes (for example Save or Open) which are opened from the SAP GUI client.
- If you encounter a warning during SAP GUI recording: "Sizing conflicts exist on the screen..." it may affect replay.

  **Workaround:** Disable the warning in the SAP GUI application:

  a. Click the right-most button on the SAP GUI toolbar (or click Alt+F2) to open the Customize Local Layout screen.

  b. Select the **SAP Internal** sub-menu.

  c. Clear the **Enable dialog box for screen size** check box.

## I was able to record a script, but why does replay fail?

In LoadRunner, make sure that the LoadRunner Remote Agent is running in Process mode. Service mode is not supported. For more information, see "Run SAP GUI Scripts from the Controller" on page 686.

## Why were certain SAP GUI controls not recorded?

Some SAP GUI controls are supported only in their menu or toolbar contexts. Try performing the problematic task using a different means—through a menu option, context menu, toolbar, and so on.

## What is the meaning of the error popup messages that are issued when I try to run the script?

Certain SAP applications store the last layout for each user (such as which frames are visible or hidden). If the stored layout was changed since the script was recorded, this may cause replay problems. For Example, in the ME52N transaction, the **Document overview Off/On** button will change the number of visible frames.

If this occurs:

1. Navigate the transaction to the same point as it was during recording, before starting replay. You can use the Snapshot viewer to see the layout in which it was recorded.
2. Add statements to the script that bring the transaction to the desired layout during replay. For example, if an optional frame interferes with your replay, insert a verification function that checks if the frame is open. If it is open, click a button to close it. For verification examples, see "Enhance SAP GUI Scripts" on page 687.

## Can I use the single sign-on mechanism when running a script on a remote machine?

No, VuGen does not support the single sign-on connection mechanism. In your SAP GUI client, open the Advanced Options and clear the **Enable Secure Network Communication** feature. Note that you must re-record the script after you modify the Connection preferences.

## Can VuGen record all SAP objects?

Recording is not available for objects not supported by SAP GUI Scripting. See your recording log for information about those objects.

## Are all business processes supported?

VuGen does not support business processes that invoke Microsoft Office module controls, nor those that require the use of GuiXT. You can disable **GuiXT** from the SAP GUI for Windows client Options menu.

### When I go to the Auto Logon node of the Recording Options, why is the list of server names empty?

If this error occurs, copy the **saplogon.ini** file from **%APPDATA%\SAP\Common**, where **%APPDATA%** stands for the environment variable specifying the Application Data folder located directly below the user profile folder. Paste the file to the %WINDIR% folder (C:\Windows).

### See also:

- For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

# Siebel Web Protocol

## Siebel Web Protocol Overview

The Siebel-Web protocol is similar to the standard Web Vuser protocol, with several changes in the default settings to allow it to work with the Siebel Customer Relationship Management (CRM) application.

You record typical activities in your Siebel session. VuGen records the actions and generates functions with a **web_** prefix, that emulate your actions.

You can also generate a Siebel Web Vuser script by analyzing an existing network traffic file (capture file). This method may be useful for creating Vuser scripts that emulate activity on mobile applications. For details, see "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723.

## Siebel Web Recording Options and Runtime Settings

Before recording a Siebel Web Vuser script, set the following recording options:

- **Recording** node: **HTML-based script**

  HTML Advanced - Script type: **A script containing explicit URLs only**

  HTML Advanced - Non HTML-generated elements: **Do not record**
- **Advanced** node: Clear the **Reset context for each action** check box.

  Before running a Siebel Web Vuser script, set the following runtime setting:

  In the Runtime Settings, clear the **Simulate a new user on each iteration** check box in the **Browser Emulation** node.

## Record Transaction Breakdown Information

VuGen provides a diagnostic tool for understanding the transaction components in your test—**transaction breakdown**. Using transaction breakdown, you can determine where the bottlenecks are and the issues that need to be resolved.

When preparing your script for transaction breakdown, we recommend that you add think time steps at the end of each transaction using the ratio of one second per hour of testing. For more information about adding think time steps, see "Insert Steps into a Script" on page 354.

In order to record the transaction breakdown information, you need to modify your the parameterization functions in your script.

## Prepare Your Script for Transaction Breakdown

1. Identify the script parameterization replacement of the Session ID.

```
/* Registering parameter(s) from source task id 15
   // {Siebel_sn_body4} = "28eMu9uzkn.YGFFevN1FdrCfCCOc8c_"
   // */
   web_reg_save_param("Siebel_sn_body4",
       "LB/IC=_sn=",
       "RB/IC==;",
       "Ord=1",
       "Search=Body",
       "RelFrameId=1",
       LAST);
```

2. Mark the next **web_submit_data** function as a transaction by enclosing it with **lr_start_transaction** and **lr_end_transaction** functions.
3. Before the end of the transactions, add a call to **lr_transaction_instance_add_info**, where the first parameter, 0 is mandatory and the session ID has a SSQLBD prefix.

```
lr_start_transaction("LoginSQLSync");
    web_submit_data("start.swe_2",
        "Action=http://design/callcenter_enu/start.swe",
        "Method=POST",
        "RecContentType=text/html",
        "Referer=http://design/callcenter_enu/start.swe",
        "Snapshot=t2.inf",
        "Mode=HTML",
        ITEMDATA,
        "Name=SWEUserName", "Value=wrun", ENDITEM,
        "Name=SWEPassword", "Value=wrun", ENDITEM,
        "Name=SWERememberUser", "Value=Yes", ENDITEM,
        "Name=SWENeedContext", "Value=false", ENDITEM,
        "Name=SWEFo", "Value=SWEEntryForm", ENDITEM,
        "Name=SWETS", "Value={SiebelTimeStamp}", ENDITEM,
        "Name=SWECmd", "Value=ExecuteLogin", ENDITEM,
        "Name=SWEBID", "Value=-1", ENDITEM,
        "Name=SWEC", "Value=0", ENDITEM,
        LAST);
lr_transaction_instance_add_info(0,lr_eval_string("SSQLBD:{Siebel_sn_body4}"));
lr_end_transaction("LoginSQLSync", LR_AUTO);
```

> **Note:** To avoid session ID conflicts, make sure that the Vusers log off from the database at the end of each session.

## Siebel Web - Troubleshooting and Limitations

This section describes troubleshooting and limitations for Siebel Web Vuser scripts.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

### Recording High Interactivity Client

The Siebel High Interactivity client is only supported with a 32-bit Internet Explorer, version 9 and earlier. To record this type of session, check your browser version and downgrade if necessary. Alternatively, you can use proxy recording. For details, see "Recording via a Proxy - Overview" on page 233.

### Back or Refresh Error

An error message relating to **Back or Refresh** typically has the following text:

We are unable to process your request. This is most likely because you used the browser back or refresh button to get to this point.

**Cause:** The possible causes of this problem may be:

- The SWEC was not correlated correctly for the current request.
- The SWETS was not correlated correctly for the current request.
- The request was submitted twice to the Siebel server without the SWEC being updated.
- A previous request should have opened a frame for the browser to download. This frame was not created on the server probably because the SWEMethod has changed since the recording.

### Same Values

A typical Web page response to the **Same Values** error is:

@0`0`3`3``0`UC`1`Status`Error`SWEC`10`0`1`Errors`0`2`0`Level0`0`ErrMsg`The same values for 'Name' already exist. If you would like to enter a new record, please make sure that the field values are unique.`ErrCode`28591`

**Cause:** The possible cause of this problem may be that one of the values in the request (in the above example, the value of the Name field) duplicates a value in another row of the database table. This value needs to be replaced with a unique value to be used for each iteration per user. The recommended solution is to replace the row ID with its parameter instead insuring that it will be unique.

### No Content HTTP Response

A typical HTTP response for a **No Content HTTP Response** type error is:

HTTP/1.1 204 No Content

Server: Microsoft-IIS/5.0

Date: Fri, 31 Jan 2003 21:52:30 GMT

Content-Language: en

Cache-Control: no-cache

**Cause:** The possible causes of this problem may be that the row ID is not correlated at all or that it is correlated incorrectly.

### Restoring the Context

The typical Web page response to the **Restoring the Context** type error is:

@0`0`3`3``0`UC`1`Status`Error`SWEC`9`0`1`Errors`0`2`0`Level0`0`ErrMsg`An error happened during restoring the context for requested location`ErrCode`27631`

**Cause:** The possible causes of this problem may be that the rowid is not correlated or that it is correlated incorrectly.

### Cannot Locate Record

The typical Web page response to the **Cannot locate record** type error is:

@0`0`3`3``0`UC`1`Status`Error`SWEC`23`0`2`Errors`0`2`0`Level0`0`ErrMsg`Cannot locate record within view: Contact Detail - Opportunities View applet: Opportunity List Applet.`ErrCode`27573`

**Cause:** The possible causes of this problem may be that the input name SWERowId does not contain a row ID for a record on the Web page. This input name should have been parameterized. The parameter's source value may have changed its location.

### End of File

The typical Web page response to the **End of File** type error is:

@0`0`3`3``0`UC`1`Status`Error`SWEC`28`0`1`Errors`0`2`0`Level0`0`ErrMsg`An end of file error has occurred. Please continue or ask your systems administrator to check your application configuration if the problem persists.`ErrCode`28601`

**Cause:** The possible causes of this problem may be that the input name SWERowId does not contain a row ID for a record on the Web page. This input name should have been parameterized. The parameter's source value may have changed its location.

#### Unable to Retrieve Search Categories

The typical Web page response to the **Unable to Retrieve Search Categories** type error is:

**Cause:** A possible cause of this problem may be that the search frame was not downloaded from the server. This occurs when the previous request did not ask the server to create the search frame correctly.

# Silverlight Protocol

## Silverlight Protocol - Overview

> **Caution:** From LoadRunner 12.55, only replay is supported for this protocol. Support for this protocol will be discontinued in future versions.

Microsoft Silverlight is a web application framework that supports graphics, animations, and interactivity. The Silverlight protocol includes the Web - HTTP/HTML protocol as a subset, as well as a number of additional functions and runtime settings.

## Silverlight - Troubleshooting and Limitations

> **Caution:** This protocol is supported for replay only. Support for this protocol will be discontinued in future versions.

This section describes troubleshooting and limitations for the Silverlight protocol.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

- REST services do not generate Silverlight service calls. However they can be replayed.
- Duplex (Polling) Binding for WCF Web Services is not supported.
- Silverlight 4 and 5 clients are supported, however applications developed using the new communication features such as net.tcp binding are not supported.
- The VuGen snapshot viewer does not support Silverlight controls.
- The Silverlight Protocol does not support applications which use Japanese, Korean, Simplified Chinese, and Traditional Chinese.

# Teradici PCoIP (PC over IP) Protocol

The PCoIP (Teradici PC over IP) protocol supports testing on the Teradici Pervasive Computing Platform and VMware HorizonView.

Because only images are passed from the server, the user can perform **connect**, **disconnect**, **mouse click**, and **key press** actions only.

## PCoIP client

The PCoIP client, is shipped with LoadRunner (resides in <LoadRunner root>\bin\pcoip). The PCoIP client supports VMWare HorizonView.

Alternatively, you can download and install the official Teradici client software from Teradici (http://www.teradici.com/product-finder/client-downloads).

## Record a PCoIP test

1. Click the **Start Recording** button.
2. In the Start Recording dialog, select **Windows Application** under **Recording mode** and enter the PCoIP client path name in the **Application** field. For example,

   ```
   <LoadRunner root>\bin\pcoipo\pcoip_client.exe
   ```

3. Enter other information as appropriate to your test and click **Start Recording**.
4. Follow the directions on the Teradici client screens to connect to the server.
5. Record your business process from within the client.

> **Note:** A call to **Copy image to the clipboard** from the snapshot context menu may throw an **OpenClipboard Failed** exception. In that event, close all open PCoIP clients.

## Enhance the recorded script

During recording, strings (alphanumeric keyboard presses) are recorded until some other type of activity occurs, such as mouse clicks or non-alphanumeric key presses.

When typing is slow, VuGen may break up a typing session into a number of invocations of **pcoip_type()** interleaved with **lr_think_time()**. By manually editing the script, these calls can be concatenated. This will facilitate parameterization if you need the string to be substituted with a parameter.

## Add a timeout for finding an area of a snapshot

Use the **pcoip_sync_on_bitmap_ex** function to add a timeout period for finding the desired area (hash) of a snapshot.

1. Click the **pcoip_mouse_click** function that contains snapshot.
2. In the Snapshot panel, right-click the snapshot and select **Insert sync on bitmap**.
3. Using the mouse, select the desired zone.
4. In the dialog box that opens, define a timeout period (in seconds), and click **OK**.

The **pcoip_sync_on_bitmap_ex** is inserted before the **pcoip_mouse_click** function.

> **Note:** The hash value is automatically generated.

## Sample Script

This script enters "Hello World" in an open application.

```
Action()
{

	pcoip_set_auth("alex-qa", "mydomain", lr_unmask("56ec1f82347be574b867f74a72"));

	pcoip_connect("MYD-SERVER", "123.456.789.abc", "MYD-SERVER", "4172", 1);

	pcoip_set_display(980, 556);

	pcoip_key("NUM_LOCK_KEY", 0);

	pcoip_mouse_click(18, 537, LEFT_BUTTON, 0, "snapshot3");

	pcoip_key("ENTER_KEY", 0);

	pcoip_key("h", MODIF_SHIFT);

	pcoip_type("ello", 0);

	pcoip_key("w", MODIF_SHIFT);

	pcoip_type("orlx", 0);

	pcoip_key("BACKSPACE_KEY", 0);

	pcoip_key("d", 0);

	pcoip_key("ENTER_KEY", 0);

	pcoip_key("TAB_KEY", 0);

	pcoip_disconnect();

	return 0;
}
```

# TruClient Protocol

TruClient is a tool for recording Web-based applications. The TruClient engine records your actions as you navigate through your business process. It creates a script in real-time, allowing you to see the steps as they are performed in a sidebar.

For details, see the TruClient Help Center (select the relevant version).

If you are working locally, you can open the TruClient Help Center from the Windows Start menu, from **Micro Focus > LoadRunner > Documentation**.

## Convert TruClient scripts to TruClient - Web

Scripts recorded in TruClient version 12.02 and earlier can be replayed only in the browser on which they were recorded. You can convert these scripts to TruClient - Web so that they can be interactively replayed on any supported browser. After the converted scripts have been interactively replayed on a particular browser, the scripts can be used on that browser in load tests.

> **Note:** TruClient script conversion to Web - HTTP/HTML is not supported for converting to a JavaScript language script.

### Converting multiple scripts

You can convert multiple scripts either from VuGen or from the TruClient Launcher.

> **Tip:** You can also use this process for a single script if the script is contained in a parent folder that contains no other scripts.

1. To run the batch conversion tool, select **Tools > TruClient batch conversion tool**. The TruClient Scripts Converter dialog box is displayed.
2. In the **Source folder** field, define the path to the folder that contains the scripts to be converted. All of the scripts under the source folder must be single-browser scripts that can be converted by the tool.
3. The **Destination folder** define the path where the converted scripts will be saved.

### Converting a single script

VuGen provides an option to convert a single script. To convert a single script:

1. Open the script in VuGen.
2. Select **Tools > Convert Script to TruClient Web** to display the conversion dialog.
3. Define a name and location for the converted script.
4. Click Convert. The converted script is displayed in the Solution Explorer.

# TruWeb Protocol (Tech Preview)

TruWeb protocol focuses on the HTTP (transport) level, providing a lightweight, scalable, and cross-platform (Windows, Linux and MacOS) solution for web protocol testing.

You can run your TruWeb scripts with LoadRunner, Performance Center, and StormRunner Load.

Alternatively, you can you can use standalone TruWeb and run your scripts using the command line or an IDE configuration. You can download standalone TruWeb from the ADM Marketplace.

For more information, see TruWeb protocol in the online LoadRunner Help Center.

> **Important:** TruWeb is released as a tech preview feature; currently you do not need a license to run TruWeb. However, this may change for future versions.

# Web - HTTP/HTML Protocol

The Web - HTTP/HTML Vuser protocol is one of VuGen's *Web Vuser* and *Mobile* protocols. This section includes information that is specific to the Web - HTTP/HTML Vuser protocol. For information that is generic to all Web Vuser protocols, see "Web Protocols (Generic)" on page 729.

## Web - HTTP/HTML Protocol - Overview

> **Note:** This topic applies to Web - HTTP/HTML Vuser scripts only.

The Web - HTTP/HTML Vuser protocol emulates communication between a browser and Web server on an HTTP or HTML level.

### When should you use the Web - HTTP/HTML Vuser protocol?

You can use the Web - HTTP/HTML Vuser protocol for browser applications that include applets and VB script, and for non-browser applications.

Use the Web - HTTP/HTML Vuser protocol when the client and the server communication is done over http/s communication, and the complexity of the communication does not require content modification. If content modification is required, consider using the TruClient protocol. For further information about TruClient, see the TruClient Help Center (select the relevant version).

### Web - HTTP/HTML Vuser Technology

You use VuGen to develop Web - HTTP/HTML Vuser scripts. To record a Web - HTTP/HTML Vuser script, you navigate through a web site - performing typical user activities. VuGen records your actions and generates a Web - HTTP/HTML Vuser script. The script contains detailed information about the recorded traffic. When you run the script, the resulting Vuser emulates a user accessing the Internet.

For details, see "Web Vuser Technology" on page 730.

## Learn how to develop a Web - HTTP/HTML Vuser script

The table below displays a list of the LoadRunner documentation that relates to the process of developing a Web - HTTP/HTML Vuser script.

| Topic | Description |
|---|---|
| Creating a Web (HTTP/HTML ) Vuser script | See VuGen's generic documentation about creating Vuser scripts ["Creating Vuser Scripts - Overview" on page 136]. |
| Recording | In addition to the generic documentation about recording Vuser scripts ["Recording - Overview" on page 154], see:<br>• "Recording Levels - Overview" on page 229<br>• **Recording Options:**<br>  • "General > Script Recording Options" on page 188<br>  • "General > Protocol Recording Options" on page 185<br>  • "General > Recording - Recording Options" on page 186<br>  • "Network > Mapping and Filtering Recording Options" on page 204<br>  • "HTTP Properties > Advanced Recording Options" on page 195<br>  • "Correlations > Rules Recording Options" on page 171<br>  • "Correlations > Configuration Recording Options" on page 169<br>  • "Data Format Extension > Chain Configuration Recording Options" on page 177<br>  • "Data Format Extension > Code Generation Recording Options" on page 180 |
| Correlating | In addition to the generic VuGen documentation on correlating Vuser scripts ["Correlation Overview" on page 251], see:<br>• "Web Manual Correlation in Code" on page 290<br>• "Data Format Extensions (DFEs) - Overview" on page 741<br>• "Using the VuGen JavaScript Engine" on page 706 |

| Topic | Description |
|---|---|
| Replaying | In addition to the generic VuGen documentation about replaying Vuser scripts ["Replay Overview" on page 295], see:<br>• "Browser Emulation - Overview" on page 735<br>• "Working with Cache Data" on page 739<br>• Runtime Settings<br>  • General - Run Logic, Pacing, Log, Think Time, Additional Attributes, Miscellaneous<br>  • Network > Speed Simulation Node<br>  • Browser > Browser Emulation Node<br>  • Internet Protocol > Proxy Node<br>  • Internet Protocol > Preferences Node<br>  • Internet Protocol > Download Filters Node<br>  • Internet Protocol > ContentCheck Node |
| Debugging | See VuGen's generic documentation about debugging Vuser scripts ["Debugging Overview" on page 328]. |
| Parameterizing | In addition to the generic VuGen documentation on parameterizing Vuser scripts ["Parameterization Overview" on page 357], see:<br>• "Data Format Extensions (DFEs) - Overview" on page 741<br>• "Using the VuGen JavaScript Engine" on page 706 |
| Adding Load Testing functionality | In addition to the generic VuGen documentation on adding load testing functionality ["Enhancing a Script for Load Testing - Overview" on page 336], see:<br>• "Text and Image Verification (Web Vuser Scripts) - Overview" on page 731 |
| Viewing Test Results | See VuGen's generic documentation about viewing test results ["Replay Summary Pane" on page 131]. |

| Topic | Description |
|---|---|
| Misc | The following miscellaneous topics are applicable to Web - HTTP/HTML Vuser scripts:<br>• "Create Web - HTTP/HTML scripts from TruClient Scripts " on page 724<br>• "JavaScript Web HTTP Scripts" below<br>• "Convert a Web - HTTP/HTML Vuser Script into a Java Vuser Script" on page 713<br>• "Web Snapshots - Overview" on page 734<br>• "Record HTTP/2 " on page 721<br>• "Record Applications Using Smooth Streaming" on page 725<br>• "Create a PCAP File" on page 869 |

### See also:

- "Web Vuser Types" on page 731 for a full list of Web Vuser protocols

## JavaScript Web HTTP Scripts

You can record Web HTTP/HTML Vuser scripts in JavaScript. You can also regenerate a C language script to convert it to JavaScript. The JavaScript engine is Google's V8 engine and is ECMAScript 5 compatible.

> **Note:** VuGen's code regeneration overwrites all manual changes that you made to a recorded script; it only regenerates the recorded functions.

JavaScript Vuser scripts support asynchronous behavior and functionality. For details, see "Create an Asynchronous Vuser Script" on page 397.

JavaScript Vuser scripts support Virtual User Tables (VTS). For more information see "VTS and Parameterization" on page 359

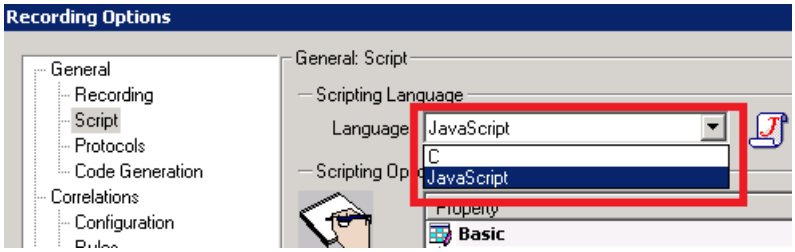
To run a script faster, you can disable JavaScript debugging at
**Runtime Settings -> Preference -> JavaScript**.

### Recording your Vuser Script in JavaScript

When you create a Web HTTP/HTML Vuser script, you are notified that you can choose to generate the script in JavaScript. See "General Options Tab" on page 109 to disable the message.

To record your script in JavaScript:

1. Click **Record > Recording Options**. Select **Script** in the General menu on the left.
2. Select **JavaScript** in the Scripting Language dropdown list.




3. Click OK.

After recording the script, the default C files are replaced by JS files.

> **Note:** Once you have recorded your script the Language option in the **Recording Options** dialog box is disabled and cannot be changed. You can change the language of a script after it is recorded only by regenerating it.

## Converting a C Script to JavaScript

You convert a C script to JavaScript by regenerating a recorded script. You cannot convert a script that you coded.

> **Caution:** If you changed a recorded script, your changes are lost when you regenerate.

To regenerate C scripts into JavaScript:

1. Click or **Record > Regenerate Script**. A warning appears that any changes to the code are overwritten.
2. Click **Options**. The Regenerate Options dialog box appears.
3. In the Script section, select the target language to convert the script to.
4. Click **OK** in the Regenerate Options dialog box. Click **OK** to approve regenerating the code.

## JavaScript Function Libraries

The native LoadRunner JavaScript library includes string functions, database connectivity functions and XML functions among others. Native library functions are available from four global objects:

- web: Web/HTTP Protocol API.
- lr: LoadRunner native API.
- vtc: VTC multiple connections API.
- lrvtc: VTC single connections API.

### Custom Libraries

In addition to LoadRunner standard functions, you can create JavaScript function libraries, or use JavaScript Language libraries such as underscore, lodash, moment, and so on.

Before you can invoke custom functions, add your custom library into the Extra files section. Your custom library must have a 'js' extension.

To activate auto-completion on your library, right click on the js file name and select **Add to Parsing List**.

> **Note:** If you regenerate your script in another language, any additional files, such as a user-defined .js file, are removed from the parsing list.

### External Libraries

In addition, the **lr.require** and **lr.loadLibrary** functions enable you to import JavaScript files at runtime. Third-party libraries are supported for libraries that are not MS-Windows or document-object based. Therefore, asynchronous functions and libraries, such as promise libraries, are not supported.

**lr.loadLibrary** is used to “include” a single file into the script. The code is evaluated and its definitions are added to the global scope of the runtime context. For example: `lr. loadLibrary (‘external.js’);`. **lr.loadLibrary** can be used for browser based libraries.

**lr.require** loads a CommonJS file or NPM module into a string, then wraps it with a function and evaluates it. Additional required files are loaded recursively along the “require” link.

> **Example:** var lodash = lr.require("..\\node_modules\\lodash");
>
> var sum = lodash.sum([1,2,3,4]);

### See also:

- "Debugging" on page 328
- The JavaScript section of the Function Reference (**Help > Function Reference**)
- Blog: Empowering LoadRunner with JavaScript or C as a scripting language

## Using the VuGen JavaScript Engine

> Applies to: Web - HTTP/HTML Vuser scripts written in C only

**What is the VuGen JavaScript Engine?**

Typically, Web - HTTP/HTML Vuser scripts contain C code. The built-in JavaScript Engine enables you to insert snippets of JavaScript code into the C code.

**What can I do with JavaScript in a Vuser script?**

You can insert JavaScript code into a Web - HTTP/HTML Vuser script to manipulate text strings that are included in the request and response messages that are sent between the client and server. Manipulating strings is often useful for correlation and parameterization purposes. Typical string

manipulations include converting decimal to hexadecimal, encoding and decoding Base64, URL encoding and decoding, and accessing object values inside JSON-formatted data.

> **Note:** It is possible to perform many of these string manipulation procedures by using the built-in DFEs (Data Format Extensions). For details, see "Data Format Extensions (DFEs) - Overview" on page 741.

Inserting JavaScript code into a Vuser script may also be useful when client-side logic is implemented in JavaScript. Inserting snippets of the original client-side JavaScript code into the Vuser script removes the requirement of having to re-write the JavaScript logic into C code to be included in the Vuser script.

You can use JavaScript code in a Vuser script to execute an **XMLHTTPRequest**. This allows you to generate and send HTTP or HTTPS requests using standard Javascript APIs. Such APIs include, for example, sending asynchronous requests, assigning callbacks to handle responses, reading responses in XML format. An **XMLHTTPRequest** used this way may replace a call to an action step such as **web_url** or **web_custom_request**.

**Why use JavaScript snippets?**

Although it may be possible to achieve the required functionality by using C code alone, including JavaScript in a Vuser script may be beneficial for the following reasons:

- JavaScript often offers a more intuitive, easier to implement solution than C.
- The JavaScript regular expression library simplifies the challenge of working with regular expressions.
- There are numerous JavaScript libraries that assist with string manipulation.
- Client-side logic is often implemented in JavaScript. Inserting snippets of the original JavaScript code removes the requirement of having to translate the JavaScript client logic into C code.

**Can I use the JavaScript Engine in Vuser scripts of all protocols?**

No, the JavaScript Engine enables you to insert JavaScript into Web - HTTP/HTML Vuser scripts only.

**What are some scenarios in which the JavaScript Engine may be useful?**

Including JavaScript code in a Vuser script may be useful in the following scenarios:

### Scenario 1: Converting a decimal number to its hexadecimal representation

In this scenario, the response that a Vuser sends to the server must include a 13-digit timestamp in hexadecimal format. For example, the date/time stamp "1234567891234" must be converted by the Vuser into hex and sent as "11F71FB0922". LoadRunner does not include any standard functionality to perform this conversion, and developing the required C code is not trivial. This problem can be resolved by inserting the following JavaScript code into the Vuser script:

```
web_js_run(
    "Code=getHexTimestamp();",
    "ResultParam=HexTS",
SOURCES,
    "Code=getHexTimestamp=function(){return new Date().getTime().toString(16).toUpperCase();}", ENDITEM,
LAST);

lr_output_message("[%s]",lr_eval_string("{HexTS}"));
```

### Scenario 2: Base64 encoding and decoding

The request and response messages that are sent between the client and server include data that is encoded using a Base64 coding scheme. Because the data is encoded, it is often difficult or impossible to parameterize or correlate the raw data. The data must be decoded before it can be parameterized or correlated, and then re-encoded before being sent to the server. By including JavaScript snippets in a Vuser script, you can access external JavaScript libraries that implement the required Base64 decoding and encoding functionality.

> **Note:** It is possible to implement Base64 decoding and encoding using the built-in Base64 DFE (Data Format Extension). For details, see "Data Format Extensions (DFEs) - Overview" on page 741.

### Scenario 3: URL encoding and decoding

In this scenario, the request and response messages that are sent between the client and server include URLs that are encoded using JavaScript URL encoding. By including a JavaScript snippet in a Vuser script, you can access the JavaScript **encodeURI()** and **decodeURI()** functions that perform the required decoding and encoding procedures.

> **Note:** It is possible to implement URL decoding and encoding using the built-in URL Encoding DFE (Data Format Extension). For details, see "Data Format Extensions (DFEs) - Overview" on page 741.

### Scenario 4: Accessing objects inside JSON-formatted data

In this scenario, the request and response messages that are sent between the client and server include data in JSON format. To access objects inside the JSON formatted data, you can include a JavaScript snippet inside the Vuser script to access the JavaScript **eval()** function.

> **Note:** It is possible to access objects inside JSON-formatted data by using the built-in JSON-To-XML DFE (Data Format Extension). For details, see "Data Format Extensions (DFEs) - Overview" on page 741.

### Scenario 5: Using XMLHTTPRequest

You can use JavaScript code in a Vuser script to execute an **XMLHTTPRequest** to download stock quotes from a specified site. For an example of how to execute an **XMLHTTPRequest**, see "JavaScript Engine: XMLHTTPRequest Example" on page 711.

### Scenario 6: Pre-existing client-side JavaScript code

To access a particular Web site, the Vuser must submit a user name and an encrypted password. The server sends a server hash to the browser to enable the browser to generate the required encrypted password. The code to generate the hash is complicated, and exists in JavaScript. This JavaScript code can be included in the Vuser script, removing the requirement to re-write the JavaScript logic into C code.

### What are the API functions that I can use in a Vuser script to execute Javascript code?

The following API functions are available for including JavaScript in a Vuser script:

1. **web_js_run:** Runs the specified JavaScript code.
2. **web_js_reset:** Clears the JavaScript context.

You use the **web_js_run** function to include JavaScript code in a Vuser script. Using the **web_js_run** function, you can either insert the required JavaScipt code into the Vuser script, or you can reference a file that contains the required JavaScript code.

### Example of inserted JavaScript code

The following is an example of how to include JavaScript code directly from the Vuser script:

```
web_js_run(
       "Code=xor((LR.getParam('buffer'), 0xFFFF));",
       "ResultParam=param",
LAST);
```

### Example of a referenced file that contains JavaScript code

The following is an example of how to include JavaScript code by calling a file that contains the JavaScript code:

```
web_js_run(
       "File=XMLHTTPRequest_sync_sample.js;",
       "ResultParam=param",
LAST);
```

For details on the above functions, and examples of how they can be used, see the Function Reference (**Help > Function Reference**).

### Can I use JavaScript to access any "internal" API functions?

JavaScript in a Vuser script gives you access to a number of "internal" API functions that can be called directly from a **web_js_run** function in the JavaScript code. These functions are used primarily for

managing parameters, but also enable you to log specified messages, record data, and run **XMLHTTPRequest**.

### Javascript-specific API functions

| API Function | Description | Arguments |
|---|---|---|
| LR.advanceParam (parameter) | Advances the specified parameter to the next value in the file. | **parameter.** The name of the parameter to advance. Must be a parameter of type file or unique number. |
| LR.setParam (name, value) | Saves a string to a parameter, creating the parameter if it does not exist. | **name.** The name of the parameter in which to save the value.<br>**value.** The value. |
| LR.freeParam (name) | Deletes a dynamic parameter at runtime, freeing its buffer. | **name.** The parameter name. |
| LR.getParam (name) | Returns the value of the specified parameter. | **name.** The parameter name. |
| LR.log(text, level) | Logs a message. | **text.** The message.<br>**level.** One of the following:<br>• "Error"<br>• "Warning"<br>• "Standard"<br>• "Extended"<br>• "Status"<br>**example:** LR.log("text", "Error"); |
| LR.userDataPoint (name, value) | Records a user-defined data point for analysis. | **name.** The name of the data point. Do not begin a data-point name with any of these strings: HTTP, NON_HTTP, RETRY, mic_, stream_, mms_<br>**value.** The numeric value. |

### How do I enable the Javascript engine for Vusers?

To run JavaScript from within a Vuser script, you must enable the JavaScript engine for the Vuser script. To enable the JavaScript engine, open the **Replay > Runtime Settings > Internet Protocol > Preferences** view. Go to the **JavaScript** section and select the **Enable running JavaScript code** option.

> **Note:** Enabling this option causes the creation of a JavaScript Runtime Engine, even if there are no JavaScript steps in the script.

### How do I configure the JavaScript engine?

You use the Vuser script's runtime settings to configure the product's custom JavaScript engine.

To access the JavaScript Engine (JSE) runtime settings, select **Replay > Runtime Settings > Internet Protocol > Preferences**, and expand the **JavaScript** section.

- **JavaScript Engine runtime size:** Specifies the size of the allocated JavaScript Engine Runtime memory, in kilobytes. This value may need to be increased when running a large number of Vusers.
- **JavaScript Engine stack size per-thread:** Specifies the size of each Vuser thread in the JavaScript Engine memory, in kilobytes. This value may need to be increased for large objects or deep stack calls.

For user interface details, see "Preferences View - Internet Protocol" on page 309.

**What is the connection between the VuGen's JavaScript Engine and VuGen's JavaScript Protocol?**

There is no connection between VuGen's JavaScript Engine and VuGen's JavaScript Protocol.

**Troubleshooting**

If you encounter difficulties when implementing JavaScript Engine support, review the items below for possible solutions.

1. Make sure that VuGen's JavaScript Engine is enabled. For details, see How do I enable the JavaScript Engine?
2. Javascript limits you to adding up to 9,000 operands in one function. For example, if you are combining strings, "str1"+"str2"+"str3"+..."str9000", you can only add up to 9,000 strings.
3. Memory issues
   - If the **Simulate a new user on each iteration > Clear cash on each iteration**runtime setting is selected, **web_js_reset** is called automatically at the start of each iteration.
   - If **Simulate a new user on each iteration > Clear cash on each iteration** is not set, avoid excessive memory consumption by inserting **web_js_reset** calls in your Vuser script at points where you no longer need the saved context.

   For details on the **web_js_reset** function, see the Function Reference (**Help > Function Reference**).

   For details on the runtime settings, see the hints below the option in the runtime settings view.
4. Performance issues

   If you are experiencing performance issues, modify the **JavaScript** runtime settings. For details, see "Preferences View - Internet Protocol" on page 309.

## JavaScript Engine: XMLHTTPRequest Example

> **Note:** This topic applies to Web - HTTP/HTML Vuser scripts written in C only.

VuGen's JavaScript Engine enables you to include JavaScript code in a Vuser script. For details on the JavaScript Engine, see "Using the VuGen JavaScript Engine" on page 706.

The example below shows how you can use a JavaScript **XMLHTTPRequest** object in a Web (HTTP/HTML) Vuser script. In this example, the **XMLHTTPRequest** object enables the Vuser to

download a stock quote from finance.example.com, and then to save the value to a parameter for future use.

The script section below shows a **web_js_run** function that has been inserted into a Vuser script. The **web_js_run** function includes a reference to a file called XMLHTTPRequest_sync_sample.js. This file contains the JavaScript code that executes the **XMLHTTPRequest** function.

```
web_js_run(
  "Code=getQuotes(LR.getParam('symbol'));",
  "ResultParam=param",
SOURCES,
"File=XMLHTTPRequest_sync_sample.js", ENDITEM,
LAST);
```

The contents of the XMLHTTPRequest_sync_sample.js file are shown below.

```
var req2;

function getQuotes(mySymbol)
{
var myURL="http://download.finance.example.com/d/quotes.csv?s="+mySymbol+"&f=l1&e=.csv";

  req2 = false;
  // branch for native XMLHttpRequest object
  try {
      req2 = new XMLHttpRequest();
    } catch(e) {
      req2 = false;
    }
  if(req2) {
    req2.open("GET", myURL, false);
    req2.send("");

  }
  return 1*req2.responseText;
}
```

- For additional examples of code used with the JavaScript Engine, see the Function Reference (**Help > Function Reference**).
- For information about the **XMLHTTPRequest** object, see http://www.w3schools.com/ajax/ajax_xmlhttprequest_send.asp.

# Convert a Web - HTTP/HTML Vuser Script into a Java Vuser Script

> **Note:** This topic applies to Web - HTTP/HTML and Java Vuser scripts only.

VuGen provides a utility that enables you to convert a Web - HTTP/HTML Vuser script into a Java Vuser script. This also allows you to create a hybrid Vuser script for both Web and Java.

1. Create an empty **Java Vuser** script and save it.
2. Create an empty **Web (HTML/HTTP)** Vuser script and save it.
3. Record a session into the Web (HTML/HTTP) Vuser script.
4. Replay the Web (HTML/HTTP) Vuser script. When it replays correctly, cut and paste the entire script into a text editor and save it as a text file (.txt).
   In the text file, modify any parameter braces from the Web type, "`{ }`" to the Java type, "`< >`".
5. Open a DOS command window and go to the <Installation_folder>/ **dat** folder.
6. Type the following command:

   ```
   <Installation Folder>\bin\sed -f web_to_java.sed filename > outputfilename
   ```

   where **filename** is the full path and filename of the text file you saved earlier, and **outputfilename** is the full path and filename of the output file.
7. Open the output file, and copy its contents into your Java Vuser script action section at the desired location.
   If you are pasting the contents into an empty custom Java template (Java Vuser type), modify the line containing `public int action()` as follows:

   ```
   public int action() throws Throwable
   ```

   This change is done automatically for recorded Java users (RMI and CORBA).
8. Parameterize and correlate the Vuser script as you would with an ordinary Java script, and run the script.

# Create a Script for a REST API

In recent years, Representational State Transfer (REST) has become a popular model for software architecture, especially for Web-based applications. Today, most large websites such as Twitter, Google, Flickr, and so forth, use REST APIs. Using the **web_rest** function, you can create a load test script for a REST API.

## REST API components

A REST API call consists of the following components:

| | |
|---|---|
| **Uniform Resource Identifier (URI)** | A string comprised of the host, the path of the functional component, and the Query string, with key-value pairs. For example :<br>`http://www.shopping.mydomain.com/en_US/home-office/-/products/Tablets/Tablets?SearchParameter=ElitePad` |
| **Method** | The action to perform, such as GET, POST, PUT, and DELETE. |
| **Data** | The data to send to the server, usually in JSON format. |

All of these components are included as parameters of a **web_rest** function. For details, see the Function Reference (**Help > Function Reference**).

## Generate **web rest** functions

To generate **web_rest** functions, do one of the following:

- Select **Design > Insert in Script > REST API**
- Right-click in the Script Editor and select **Insert > REST API**

The REST Step Properties dialog box that opens includes various tabs that help you to generate and check the required **web_rest** functions.

| Tab | Function |
|---|---|
| **Build Step** | Define the **web-rest** step:<br>• Specify the required URL and select a method.<br>The request body editor area changes slightly depending on whether the selected method can have a body attached to it or not.<br>• Specify the required URL parameter key-value pairs. As you add key-value pairs, VuGen adds them to the URL above.<br>Click **+Add** to add additional key-value pairs.<br>• Specify the required parameters for the headers. You can enter any string as the header name, or you can select one of the common headers from the drop down list.<br>Click **+Add** to add additional header parameters.<br>• Specify the required values for the body parameters. Click **+Add** to add additional body parameters. |
| **Preview** | Shows a preview of the request code that is generated, based on the data that is contained in the Build Step tab. This is the code that is sent to the server when the resulting **web_rest** function is run. |
| **Step Results** | Shows the HTTP response that was returned by the server after the most recent step run. The response is divided between the Body, Cookies, and Headers tabs.<br>The status of the response appears in the top right corner of the dialog box. |

- After you have entered the required attributes for the **web_rest** function, you can click **Run Step** to run the step to test that it functions correctly. The HTTP response to the step run is displayed in the Step Results tab. The HTTP status of the test run appears in the top right corner of the dialog box.
- After you have entered and checked the required attributes of the step, click **Insert Step** to insert the **web_rest** function into your script, at the cursor location.

> **Note:** After you click **Insert Step** to insert the step into your script, VuGen adds the step to the step history list. Click **Step History** (located on the left of the REST Step Properties dialog box) to show the step history list.
>
> You can select any step in the list to insert the step details into the Build Step tab. You can then modify the step, run it, and insert it into the script.
>
> A separate step history list is maintained for each computer on which VuGen is installed. If required, click **Clear History** to clear the step history list.

## Generate a web.rest function in JavaScript

To generate a web.rest function in JavaScript :

1. Record your script in JavaScript. See "Recording your Vuser Script in JavaScript" on page 704.

2. Press **Ctrl+Shift+W** to launch the REST Step Properties dialog box and define the web.rest function. See "Generate web rest functions" on the previous page.

## Edit an existing **web_rest** function

To open the REST Step Properties dialog box to edit an existing **web_rest** function, right-click inside the function and select **Show Arguments**.

## Create script steps for a Web - HTTP/HTML script that calls a REST API

1. Create a Web (HTTP/HTML ) Vuser script. For details see, "Creating Vuser Scripts - Overview" on page 136.
2. Record the REST API application while you perform typical business processes in the application.

   After you finish recording, add new **web_rest** calls using the following format, where the attributes are **URL**, **Method**, **ResType**, **Body**, and **BodyFilePath**:

   ```
   web_rest("<request_name>",
    "<Attribute_List>",
    LAST);
   ```

3. (Optional) To reference a file with the JSON data instead of entering the actual text:
   a. In the Solution Explorer, right-click the **Extra Files** node and select **Add Files to Script** to add the .json data file.

b. Replace the **Body** argument with **BodyFilePath=<file_name.json>**.

c. Allow JSON files. Select **Tools > Options > Scripting > Script Management** and add .json to the Allowed Extensions list.

### Examples

The following example shows a REST API function that updates values using a **PUT** action:

```
web_rest("update customer info",
 "URL=http://myServer/customers/{cust_id}",
 "Method=PUT",
 "ResType=JSON",
 "Body={ \"address\":\"yyyy\", … }",
LAST);
```

The next example shows a REST API function that uses data from a file when updating values using a **POST** action:

```
web_rest("create new customer",
 "URL=http://myServer/customers/",
 "Method=POST",
 "ResType=JSON",
 "BodyFilePath = c:\\my_data\\data.json" /* BodyFile for large content */
LAST);
```

## Using Emulation to Record Mobile Applications

Using an emulator to test mobile applications often provides a robust solution for many mobile devices. To test mobile devices using emulation, you install a third party emulation application on your local computer, and then record events using the "Windows Application" method in VuGen's Start Recording dialog box. (For details, see "Start Recording Dialog Box" on page 240.) After selecting the "Windows Application" method, the Start Recording dialog box requires the following emulator settings in order to start the emulator:

- **Emulator to record:** Enter this in the **Application** field.
- **Command line:** Enter this in the **Program arguments** field.
- **Working directory:** Enter this in the **Working directory** field.

The table below shows emulator download sites and examples of emulator settings for various mobile operating systems:

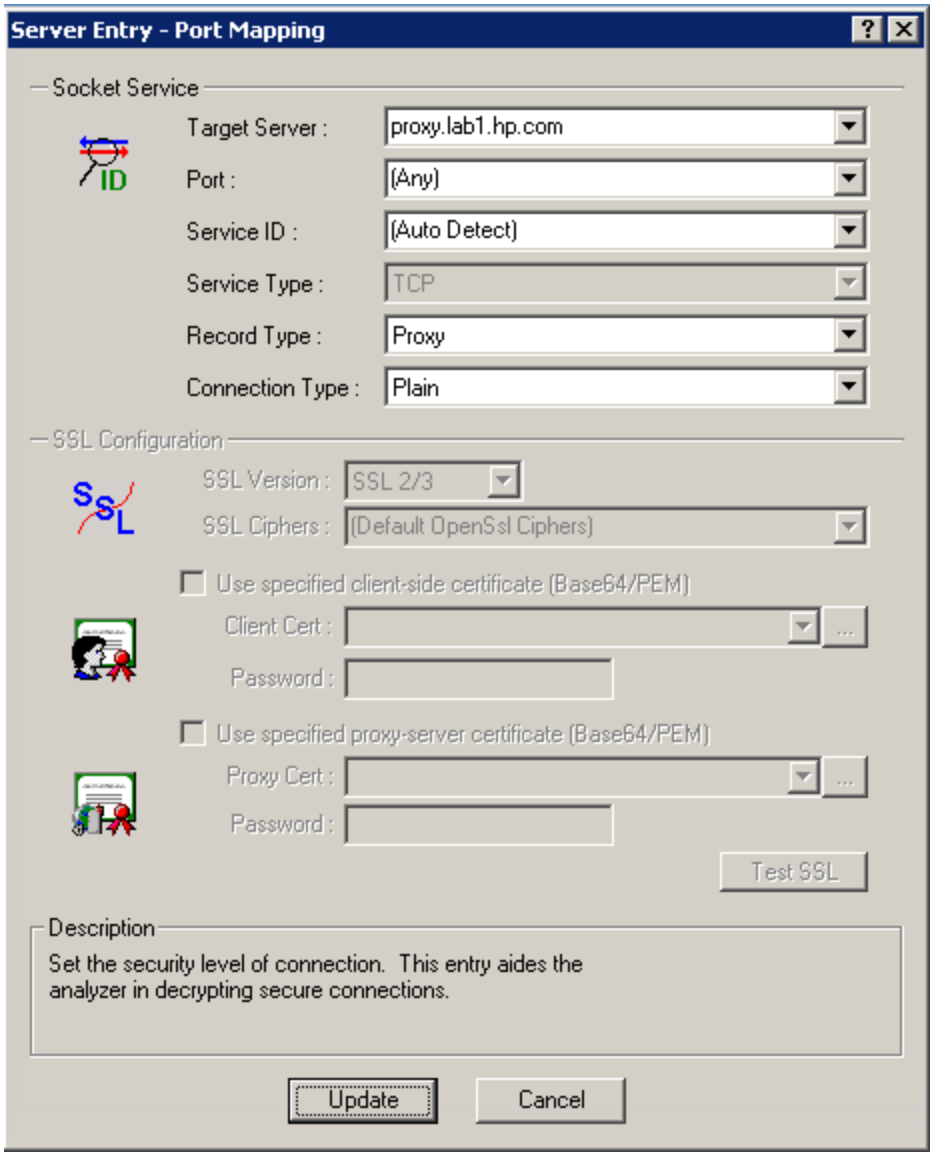
| Link to Download SDK | Emulator to record (example) | Command line (example) | Working directory (example) |
|---|---|---|---|
| | | | |
| **Android** | | | |
| Android Emulator Download | C:\Program Files\Android\android-sdk-windows\tools\emulator.exe | @Android_v2.2 | C:\Program Files\Android\android-sdk-windows\tools |
| **Blackberry** | | | |
| Blackberry Emulator Download | C:\Program Files\Research In Motion\BlackBerry Smartphone Simulators 6.0.0\6.0.0.337 (9800)\fledge.exe | /app=Jvm.dll /handheld=9800 /session=9800 /app-param=DisableRegistration /app-param=JvmAlxConfigFile:9800.xml /data-port=0x4d44 /data-port=0x4d4e /pin=0x2100000A | C:\Program Files\Research In Motion\BlackBerry Smartphone Simulators 6.0.0\60.0.337 (9800) |
| **Windows 8** | | | |
| Windows Mobile Emulator Download | C:\Program Files\Microsoft SDKs\Windows Phone\v7.0\Tools\XDE Launcher\XdeLauncher.exe | "Windows Phone 7" "Windows Phone 7 Emulator" | C:\Program Files\Microsoft SDKs\Windows Phone\v7.0\Tools\XDE Launcher |

## Recording with a Google Android Emulator

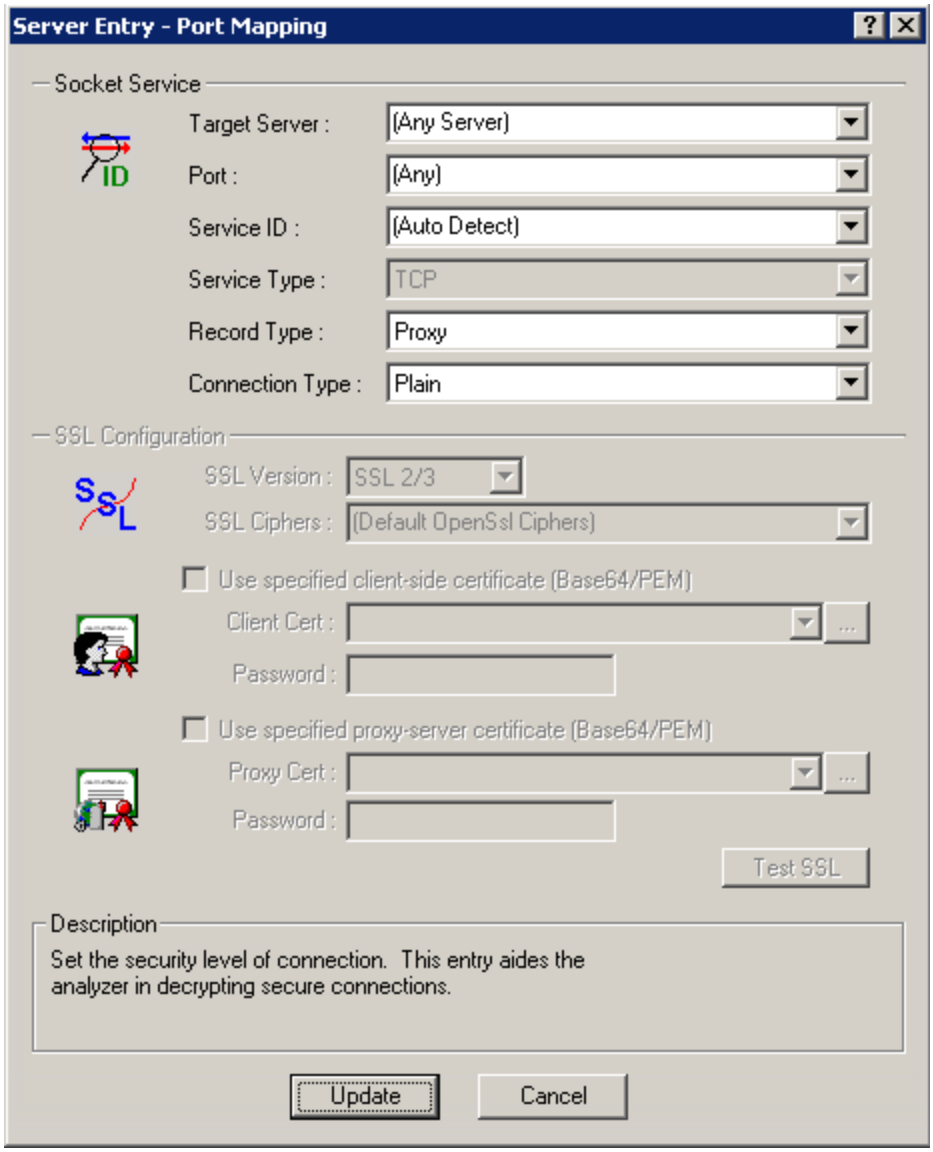
### Google Android Emulator Version 2.0 and above

If you experience problems when recording with Google Android Emulator Version 2.0 and above, apply the following workaround:

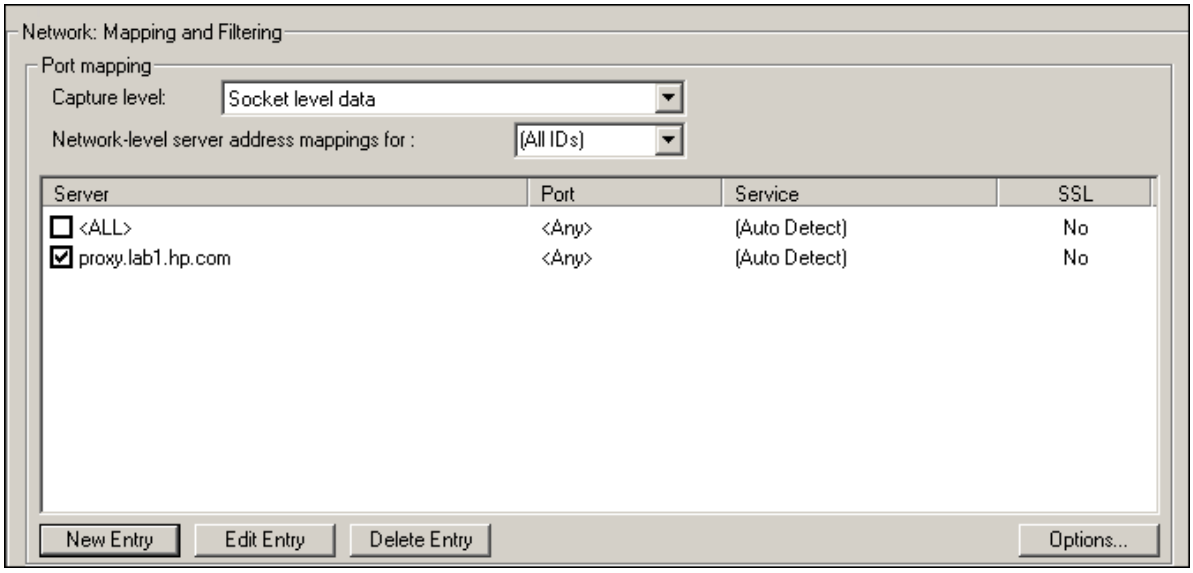
1. Enter a new Port Mapping by selecting **Recording Options > Network > Mapping and Filtering** and select **New Entry** in the **Port Mapping** section.
2. Specify a Target server and port.




3. Enter a second Port Mapping entry without changing any details.



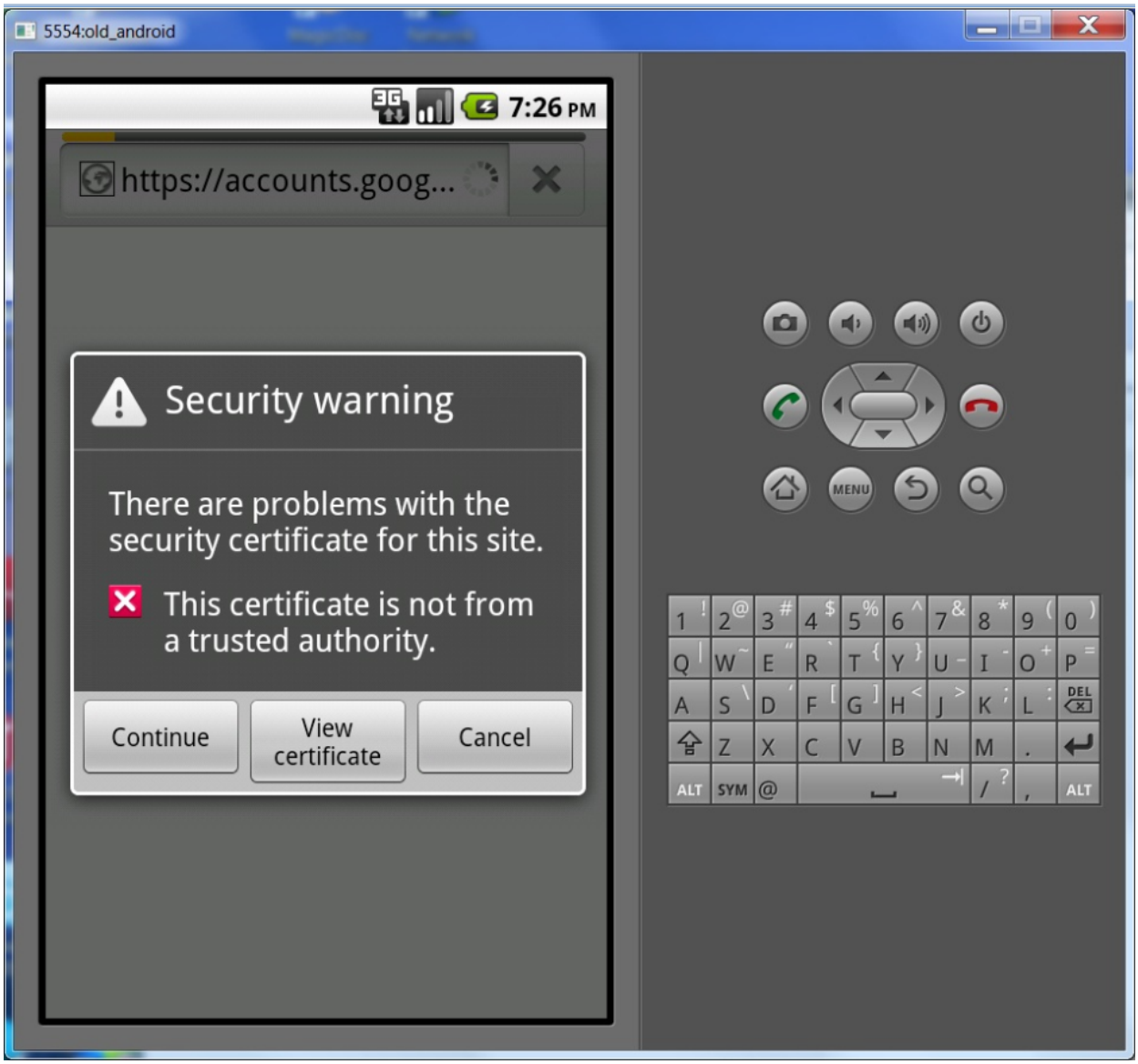

4. Disable the second entry so all traffic is handled by the first entry in the Port Mapping.




## Google Android Emulator - Resolved limitation

The following limitation was resolved in Google Android Emulator Version 4.03:

While recording an SSL site, you may encounter a warning message stating that there are problems with the certificate for the site. Click **Continue** to proceed with the recording.




## Record HTTP/2

HTTP/2 was standardized as a successor to HTTP 1.1 in May 2015. This version does not replace HTTP. Instead, it improves the way HTTP requests and responses are sent.

HTTP/2 allows multiplexing and concurrency, letting a client send multiple requests successively on the same TCP connection. This eliminates the need for multiple connections between the client and server.

In HTTP/2, the header size is substantially smaller. In addition, the server can push information to the client, even before it was requested.

All of these changes contribute to HTTP/2 being faster and more efficient than its predecessor.

**Tip:** To determine if the Web site you are testing supports HTTP/2, perform an Internet search

for an HTTP/2 detection tool. Most common browsers support HTTP/2 over TLS.

### To record a Web - HTTP/HTML script that uses the HTTP/2:

1. Create a Web (HTTP/HTML ) Vuser script. For details see, "Creating Vuser Scripts - Overview" on page 136.
2. Select **Record > Recording Options > Network > Maps and Filtering > Options** to open the Advanced Port Mapping Settings dialog box.
3. In the **SSL Version** dropdown, select **TLS ALPN**. Click **Update**.
4. Start recording and perform typical business processes on your application.
5. To confirm that recording was done with HTTP/2, examine the headers that were recorded in the script.

   To verify that HTTP/2 was used for replay, enable **Advanced Trace** in the **Log** runtime settings. After replay, see the Replay log and note the references to HTTP/2.

## Record and Replay on servers with SNI Enabled

Server Name Indication (SNI) is an extension of the TLS protocol, in which a client indicates which hostname it is attempting to connect to at the start of the handshaking process. This allows multiple certificates to be sent from a single server. It is often used on cloud servers, such as AmazonCloud.

You can control whether VuGen sends an SNI extension to SSL handshakes using:

**web_set_sockets_option("TLS_SNI",** *<value>***);**

Where:

| Value | Description |
|---|---|
| "1" | **Default.** Enable extension and send the server name value derived from the URL (host name) |
| "0" | Disable extension. Do not send an SNI extension with SSL handshakes. |
| "<server name>" | Manually specify the SNI extension value for the next handshake only. |

### Additional SNI Guidelines

VuGen uses the following guidelines for SSL handshakes:

**During recording:**

- If SNI support is detected, no special step is added.
- If SNI is not detected, a
  `web_set_sockets_option("TLS_SNI", "0");`
  statement is added to the beginning of the script.

**During replay:**

- When the `<server_name>` option is used, the specified server name is used for the next handshake only.
  Subsequent handshakes revert to the standard enabled (1) or disabled (0) behavior.
- If no `web_set_sockets_option("TLS_SNI", <value>);` statement exists, or if a `web_set_sockets_option("TLS_SNI", "1");` exists, the default SNI extension is sent with each SSL handshake.
  Therefore, scripts created prior to support of this option will run with SNI enabled.
  To prevent this behavior, add `web_set_sockets_option("TLS_SNI", "0");` to your script.

## Create a Vuser Script by Analyzing a Captured Traffic File

> **Note:** This task is applicable to the following Vuser types: Web - HTTP/HTML, Flex, SAP - Web, and Siebel - Web.

This topic describes how to use an existing network traffic file (capture file) to generate a Vuser script. This method may be useful for creating Vuser scripts that emulate activity on mobile applications.

> **Note:**
> - HAR file entries must include the **startedDateTime** field.
> - In addition to the procedure described below, you can create a Vuser script by right-clicking a captured traffic file (.pcap, .cap, .saz, or .har ) in Windows Explorer, and then selecting **Create VuGen script**.

1. Create a new Vuser script in VuGen.
   Note that this functionality is available only for those Vuser protocols listed in the note at the top of this topic.
2. Click **Start Recording** to open the Start Recording dialog box.
3. From the list of Recording methods, select **Captured Traffic File Analysis**.
4. Locate and select your capture file (**pcap**, **cap**, **saz** (Fiddler), or **har**).
   For details about creating a new capture file in a Windows, Linux, or mobile environment, see "Create a PCAP File" on page 869.
5. Specify the client side filter. This is the IP address of the *client* whose traffic you want to use to generate the Vuser script. VuGen typically detects the client side filter by analyzing the capture file.
6. You can use the recording options to specify server side filters by clicking **Recording Options > Mapping and Filtering**.
7. Add a list of the SSL attributes for the servers being analyzed. Use the **Add**, **Edit**, and **Remove** buttons to manage the entries in the list. The **Add** button opens the Add SSL Attribute dialog box, allowing you to add a server and specify its IP address, port, certificate file, and password if required.

> **Note:** The SSL attribute list is visible only after you select a **pcap** capture file that contains SSL data and requires a certificate. This list is not available for **har** and **saz** files — instead, configure the SSL through Fiddler.

8. Click **Start Recording**. VuGen analyzes the capture file, and generates a Vuser script.

## Create Web - HTTP/HTML scripts from TruClient Scripts

Script development with TruClient is fast, but more Vusers can be run with VuGen Web - HTTP/HTML than with TruClient. You can combine the advantage of fast script development, and the advantage of running many Vusers, by developing your script with TruClient and then converting it to a Web - HTTP/HTML script.

> **Note:** During the conversion, comments and APIs are added to the Web - HTTP/HTML protocol script that document the conversion process.

1. Create a TruClient script. See **Record a TruClient Web script** in the TruClient Help Center (select the relevant version).
2. Save the script and close the TruClient Side Bar.
3. From the VuGen toolbar, click the

button to convert the script.
4. After the script is generated, review the script, keeping in mind that the Web - HTTP/HTML protocol records on the transport level. For example, you may need to address correlation or parameter issues in your converted script.

When you add a converted script to a scenario, VuGen offers to create a new Vuser group for the script, provided that the original TruClient script is in the specified path. Having a separate group enables you to view its results separately.

**Limitations**

Conversion of TruClient scripts to Web - HTTP/HTML scripts does not support converting steps that call 127.0.0.1 (localhost).

## Record Streaming Media in Web - HTTP/HTML

The Web streaming functions emulate communication between a client and a server that provides streaming media using one of the streaming protocols, HTML5 or HLS.

If the application under test runs on mobile devices (browser or APPs), use proxy recording. If the application's client is a desktop browser, application or plugins, you can use either hook-based or proxy based recording.

Set timeouts, number of retries, and configure snapshots and logging in the Run-Time Settings under **Network > Streaming**. Make sure your timeouts do not conflict with the settings of **Internet Protocol >**

**Preferences > HTTP.** Step time in streaming media includes the play time, so set the Internet Protocol timeouts to allow for playing.

The downloaded media is not decoded. Instead, it is consumed by a player emulator. You control the emulator with the streaming functions:  wait, seek, play, and pause. Other streaming functions get and set attributes of the stream. This protocol does not record user actions on a media player widget. Except for open and close, all the functions are added manually.

### See also:

- VuGen Function Reference. These functions are under **Web Vuser Functions > Alphabetical List of Web Vuser Functions**. The names all start with "web_stream_".

  To open the Function Reference from a machine with LoadRunner installed, from the Windows Start menu, go to **Micro Focus > LoadRunner > Documentation > Function Reference**.
- "Streaming State Statistics Monitoring" on page 1221
- "Streaming State Statistics Graphs" on page 1467

## Record Applications Using Smooth Streaming

Smooth Streaming is an Internet Information Services (IIS) Media Services extension that provides high-quality video streaming to clients over HTTP. Smooth Streaming adapts the stream rate and quality by monitoring the local bandwidth and video playback performance of the client while traditional streaming delivers the content at a fixed rate and quality.

### To prepare a script for load testing for applications that use Smooth Streaming:

1. Create a Web (HTTP/HTML ) Vuser script. For details see, "Creating Vuser Scripts - Overview" on page 136.
2. Look for the “Manifest” request at the start of the streaming communication:

```
web_custom_request("Manifest",
        "URL=http://mediadl.microsoft.com/mediadl/iisnet/smoothmedia/E
xperience/BigBuckBunny_720p.ism/Manifest",
        "Method=GET",
        "Resource=0",
        "RecContentType=text/xml",
        "Referer=",
        "Snapshot=t11.inf",
        "Mode=HTTP",
        LAST);
```

3. Following the “Manifest” request, you should find a number of streaming requests:

```
    web_custom_request("Fragments(video=0)",
        "URL=http://mediadl.microsoft.com/mediadl/iisnet/smoothmedia/E
xperience/BigBuckBunny_720p.ism/QualityLevels(350000)/Fragments(video=
0)",
        "Method=GET",
        "Resource=1",
        "RecContentType=video/mp4",
        "Referer=",
        "Snapshot=t12.inf",
        LAST);

    web_custom_request("Fragments(audio=0)",
        "URL=http://mediadl.microsoft.com/mediadl/iisnet/smoothmedia/E
xperience/BigBuckBunny_720p.ism/QualityLevels(64000)/Fragments(audio=0
)",
        "Method=GET",
        "Resource=1",
        "RecContentType=video/mp4",
        "Referer=",
        "Snapshot=t13.inf",
        LAST);
```

Create and configure parameters to emulate different bandwidths than the ones that were recorded. For example:

a. In the streaming request, replace QualityLevel with a parameter named 'qualityLevel'.

```
web_custom_request("Fragments(video=0)",
        "URL=http://mediadl.microsoft.com/mediadl/iisnet/smoothmedia/E
xperience/BigBuckBunny_720p.ism/QualityLevels({qualityLevel})/Fragment
s(video=0)",
        "Method=GET",
        "Resource=1",
        "RecContentType=video/mp4",
        "Referer=",
        "Snapshot=t12.inf",
        LAST);
```

b. Configure the 'qualitylevel' values that will be used during each iteration of the load test in the "Parameter Properties Dialog Box" on page 376.

4. Replay the script and verify that the size of the response from each request corresponds to the value of the parameter that was sent.

## Common Web Recording Problems

This topic covers some common issues you may have when recording web scripts and provides possible causes and solutions.

## No events are being recorded

- **Problem:** The events counter on the recording toolbar keeps increasing, while the generated script is empty.
- **Possible Cause:** VuGen's recording mechanism may be unable to identify HTTP data.
- **Solution:** Ensure that your application indeed uses HTTP Web traffic.

  If your application uses SSL connections, make sure you have the correct SSL version (SSL 2, SSL 3, TLS) in the Port Mapping dialog box (**Record > Recording Options > Network > Port Mapping > Options**).

  In the Advanced Port Mapping Settings dialog box, make sure SSL is enabled and select the correct version.

## Very few events are being recorded

- **Problem:** The events counter shows less than five events, while the application keeps getting data from the server.
- **Possible Cause:** VuGen's recording mechanism is unable to capture activity over the network.
- **Solution:** Ensure that your application really does provide some network traffic, that it actually sends and receives data through the IP network.

  If an antivirus program is running, turn it off during recording. Check the recording log for any clues about the recording failure. Messages such as "connection failure" or "connection not trapped" can be a sign of the incorrect Port Mapping settings.

  In addition, if you are recording on a Chrome or Firefox browser, make sure that all the instances of the browser are closed prior to recording.

## Specific events are not being recorded

- **Problem:** Certain events are not being recorded.
- **Possible Cause:** VuGen has classified this event as an insignificant action. By default, the Web - HTTP/HTML protocol only records client requests that return an HTTP response status of 2xx or 302, and discards all other requests. If a request returns a response that was discarded, such as 301, VuGen will not generate a step.
- **Solution:** Modify the registry to include the missing status. Locate the following registry key:

  ```
  [HKEY_CURRENT_USER\Software\Mercury Interactive\Networking\Multi Settings\QTWeb\Recording]
  ```

  Add the following string value to it:

  ```
  "GenerateApiFuncForCustomHttpStatus"="301"
  ```

## Application hangs during recording

- **Problem:** The recorded application becomes unresponsive during the recording.
- **Possible Cause:** VuGen's recording mechanism is unable to connect to the application's server.

- **Solution:** Check the Recording Log in the Output pane (select Recording from the drop down list of logs) for a message about the Request Connection: Remote Server.

Open the Port Mapping dialog box (**Record > Recording Options > Network > Port Mapping > Options**) and add an entry **(New Entry)** for the application's server. Clear the check box adjacent to that entry. This will ensure that the above IP and port are not recorded—the application connects to them without any LoadRunner involvement.

If communication with the server is essential to the business process, keep the entry checked.

### Wrong Server Certificate

- **Problem:** During the recording the recorded application shows an error message about a wrong server certificate.
- **Possible Cause:** The problem is caused by the inability of the client side to verify the validity of the server certificate.
- **Solution:** The LoadRunner Certificate Authority (CA) file should be added to the machine's "Trusted Root Certificate Authorities" certificate store (in case of a Java client application, LoadRunner's CA should be added to Java's trusted CA list using the keytool). This file is provided with LoadRunner, and is called **wplusCAOnly_Expiration_2022.crt**, located in the <LoadRunner root>\bin\certs folder.

  To add it to the store, double-click on the file to open the certificate. Then click **Install Certificate...** to open the Certificate Import Wizard. Use the **Place all certificates in the following store** option and select **Trusted Root Certification Authorities**. When the wizard is completed, you should be able to record the application.

### See also:

- Blog: 5 tips to solve the most common problems seen in Web recording

## Troubleshooting and Limitations - Web - HTTP/HTML Protocol

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

- Certain POST requests may require HTTP headers which are not automatically generated in the recorded script. To add headers, use the `web_add_header` API function. For details, see the Function Reference (**Help > Function Reference**).
- Port mapping configurations is not supported in the Proxy Recording mode.
- VuGen cannot get a client certificate from Internet Explorer 10 while recording a session.
  **Workaround:** Provide a client certificate in the port mapping settings.
- When strong private key protection is set on a certificate and the WinInet mode is used during the replay, you may be required to manually enter authentication details when replaying the script.
- In previous versions of LoadRunner, the C type "char" was considered a "signed char". In LoadRunner 11.50 and later, it is considered as an "unsigned char". If you used "char" without specifying whether it

is signed or unsigned, and performed arithmetic operations on this variable, the results may be different when comparing current results with those from previous versions of LoadRunner.

- When using certificates, if your script refers to the certificate by a wrong index number, such as **web_set_certificate_ex("CertIndex=2", LAST)**, the authentication may fail.
  **Solution:** Regenerate the script (not for WinInet mode recording), or manually set the correct certificate index. To determine this, in Internet Explorer, open the Certificates list **(Tools > Internet options> Content > Certificates)** and use the order of the certificates as they are listed, beginning with 1.
- When recording a Java application with SSL, you need to install the LoadRunner CA certificate. To install the certificate:
  a. Copy the certificate file **wplusCAOnly_Expiration_2022.crt** from the **<LoadRunner root>\bin\certs** folder, to the location of the Java keystore file, *cacerts*. This example assumes the typical folder, **C:\Program Files (x86)\Java\jre6\lib\security**.
  b. Rename the file to wplus.crt
  c. Open the command prompt, and go to the keystore file's folder, by typing `cd C:\Program Files (x86)\Java\jre6\lib\security`.
  d. Run `C:\Program Files (x86)\Java\jre6\bin\keytool -list -keystore cacerts` (no quotes).
  e. You should see the list of all installed certificates. When prompted for a password, enter **changeit** (no quotes).
  f. Run `C:\Program Files (x86)\Java\jre6\bin\keytool -import -file wplus.crt -keystore cacerts -alias LoadRunnerCA -trustcacerts` (no quotes). When prompted for a password, enter **changeit** (no quotes).
  g. Record your secure application.

### See also:

- "Troubleshooting and Limitations for Recording with VuGen" on page 248
- "Common Web Recording Problems" on page 726

# Web Protocols (Generic)

The **Web Protocols** section includes information that is generic to all Web Vuser protocols. For information that is specific to a given Web Vuser protocol, see the documentation for that specific protocol.

## Web Protocols - Overview

> **Note: Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" on page 731.

You use VuGen to develop Web Vuser scripts. While you navigate through a site performing typical user activities, VuGen records your actions and generates a Vuser script. When you run the script, the resulting Vuser emulates a user accessing the Internet.

Suppose you have a web site that displays product information for your company. The site is accessed by customers and potential customers. You want to make sure that the response time for any customer query is less than a specified value (for example, 20 seconds)—even when many users (for example, 200) access the site simultaneously. You use Vusers to emulate this scenario, where the web server services simultaneous requests for information. Each Vuser could do the following:

- Load the home page
- Navigate to the page containing the product information
- Submit a query
- Wait for a response from the server

You can distribute several hundred Vusers among the available testing machines, each Vuser accessing the server by using its API. This enables you to measure the performance of the server under the load of many Vusers.

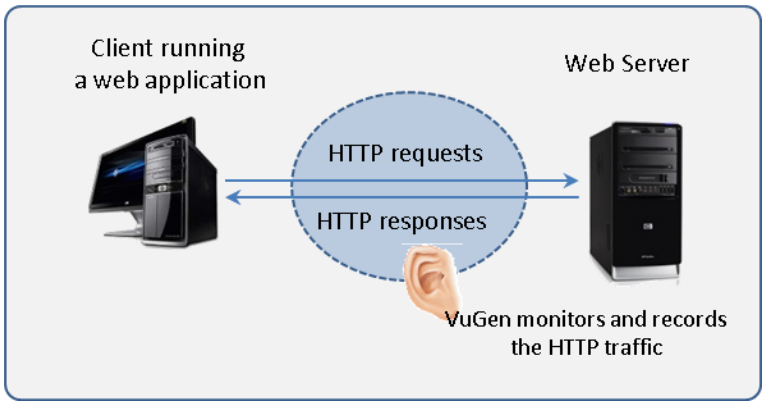
### See also:

- "Web Vuser Technology" below

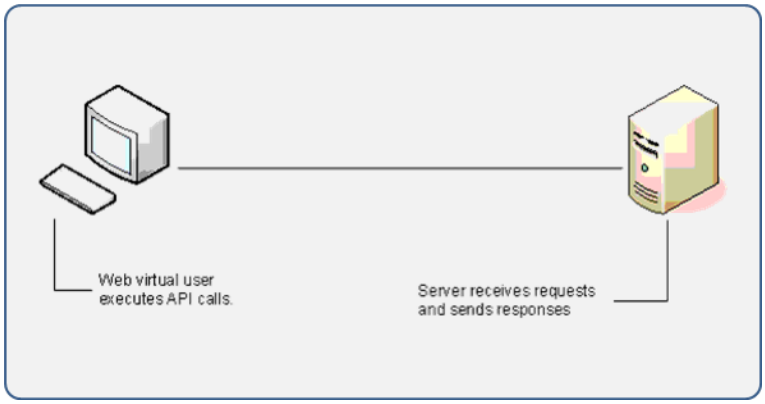
## Web Vuser Technology

> **Note: Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" on the next page.

VuGen creates Web Vuser scripts by monitoring and recording the web traffic that flows between a web browser and a web server while you perform typical business processes. The web traffic includes HTTP requests sent by the browser to the server, and the HTTP responses returned by the server.




When you run a Web Vuser script, the resulting Vuser communicates directly with the web server without relying on a browser or client software. To perform this communication, the Vuser script sends web API functions directly to the web server.




## Web Vuser Types

LoadRunner enables you to build and run web-based Vuser scripts using a variety of Web Vuser protocols. Following is a list of LoadRunner's Web Vuser protocols:

- Flex
- Java over HTTP
- Oracle - Web Applications
- SAP – Web
- Siebel – Web
- TruClient - TruClient - Web, TruClient - Native Mobile, TruClient - Mobile Web
- Web - HTTP/HTML

## Text and Image Verification (Web Vuser Scripts) - Overview

> **Note: Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" above.

VuGen enables you to add checks to your Web Vuser scripts. A check verifies the presence of a specific object in a Web page. The object can be either a text string or an image.

Checks enable you to determine whether or not your Web site is functioning correctly while it is being accessed by many Vusers—that is, does the server return the correct Web pages? This is particularly important while your site is under the load of many users, a period when the server is more likely to return incorrect pages.

For example, assume that your Web site displays information on the temperatures in major cities around the world. You use VuGen to create a Vuser script that accesses your Web site.

The Vuser accesses the site and executes a text check on this web page. For example, if the word **Temperature** appears on the page, the check passes. If **Temperature** does not appear because, for example, the correct page was not returned by the server, the check fails. Note that the text check step appears before the URL step. This is because VuGen registers, or prepares in advance, the search

information relevant for the next step. When you run the Vuser script, VuGen conducts the check on the web page that follows.

Although the server may display the correct page when you record the script and when a single Vuser executes the script, it is possible that the correct page will not be returned when the server is under the load of many Vusers. The server may then be overloaded and may therefore return meaningless or incorrect HTML code. Alternatively, in some instances when a server is overloaded, the server returns a **500 Server Error** page. In both of these cases, you can insert a check to determine whether or not the correct page is returned by the server.

> **Note:** Checks increase the work of a Vuser, and therefore you may be able to run fewer Vusers per load generator. You should use checks only where experience has shown that the server sometimes returns an incorrect page.

### See also:

- "Understanding Web Text Check Functions" below
- "Add Text Checks and Image Checks (Web Vuser Protocols)" on the next page

## Understanding Web Text Check Functions

> **Note: Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" on the previous page.

### web_reg_find

When you add a text check to a Web Vuser script, VuGen adds a **web_reg_find** function to the script. This function registers a search for a text string in an HTML page. Registration means that the Vuser does not execute the search immediately—rather the Vuser performs the check only after executing the next Action function, such as **web_url**.

> **Note:** If you are working with a concurrent functions group, the **web_reg_find** function is executed only at the end of the grouping.

In the following example, the **web_reg_find** function searches for the text string "Welcome". If the string is not found, the next action function fails and the script execution stops.

```
web_reg_find("Text=Welcome", "Fail=Found", LAST);

web_url("Step", "URL=...", LAST);
```

In addition to the **web_reg_find** function, you can use other functions to search for text within an HTML page including:

### web_find

This function is used primarily used for backward compatibility. **web_find** differs from the **web_reg_find** function in that **web_find** is limited to HTML-based scripts (see **Recording Options > Recording** tab). It also has less attributes, such as instance, allowing you to determine the number of times the text appeared. When performing a standard text search, **web_reg_find** is the preferred function.

### web_global_verification

This function lets you search the data of an entire business process. In contrast to **web_reg_find**, which applies only to the next Action function, this function applies to **all** subsequent Action functions such **web_url**. By default, the scope of the search is NORESOURCE, searching only the HTML body, excluding headers and resources.

**web_global_verification** is ideal for detecting application level errors that are not included the HTTP status codes. This function is not limited to an HTML-Based script (see **Recording Options > Recording** tab).

## Add Text Checks and Image Checks (Web Vuser Protocols)

> **Note: Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" on page 731.

### Add a Text Check While Recording

1. In the web browser, select the desired text.
2. Click the **Insert Text Check** button on the VuGen Recording toolbar. VuGen adds a **web_reg_find** function to the script.

> **Note:** When recording a multi-protocol script, you can only insert a text check in the Web part of the script.

### Add a Text Check After Recording

1. In the Snapshot pane, display a snapshot that contains the text you want to verify.
2. In the Snapshot pane toolbar, click **HTTP Data** to display the HTTP Data view of the snapshot.
3. In the snapshot, select the text you want to verify. The text must be located in a response section of the snapshot - not in a request section.
4. Right-click and select **Add Text Check Step** from the menu.
5. Modify the options in the Find Text dialog box. For details on the dialog box options, press F1

when in the dialog box to open the Function Reference.

6. Click **OK** to insert the function into the Vuser script.

### Add Other Text Checks After Recording

1. In the script editor, locate the position where you want to insert the check.
2. Select **View > Toolbox** to open the Toolbox.
   a. To insert a **web_reg_find** function, in the Toolbox, under **Services**, select **web_reg_find**.
   b. To insert a **web_global_verification** function, in the Toolbox, under **Services**, select the required **web_global_verification** function.
3. Drag the selected function to the required location in the Editor.
4. Enter the required details in the dialog box that opens. For details on the dialog box options, press F1 when in the dialog box to open the Function Reference.
5. Click **OK** to insert the function into the Vuser script.

### Add an Image Check After Recording

1. In the Editor, locate the position where you want to insert the check.
2. Select **View > Toolbox** to open the Toolbox.
3. In the Toolbox, under **Web Checks**, select **web_image_check**.
4. Drag the selected **web_image_check** function to the required location in the Editor.
5. Enter the required details in the Image Check Properties dialog box. For details on the dialog box options, press F1 when in the dialog box to open the Function Reference.
6. Click **OK** to insert the function into the Vuser script.

### See also:

- "Text and Image Verification (Web Vuser Scripts) - Overview" on page 731

## Web Snapshots - Overview

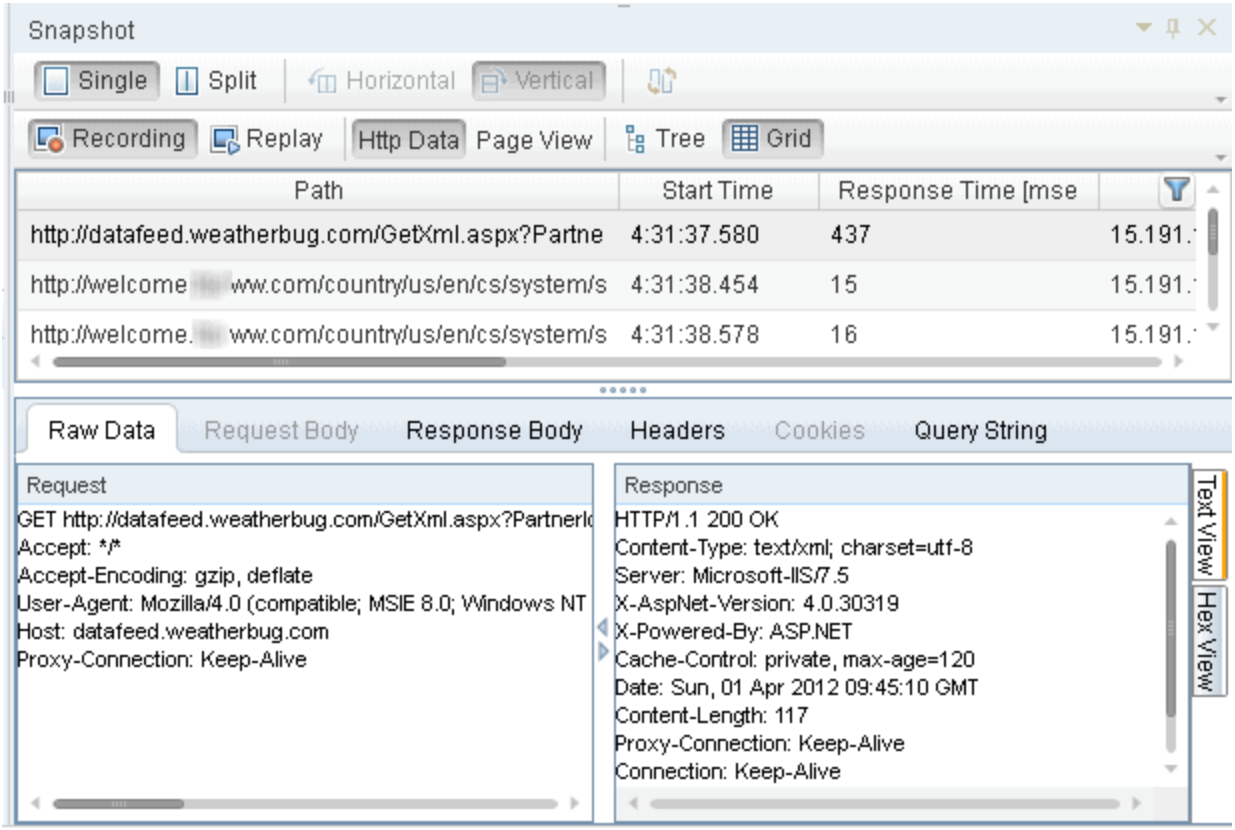
> **Note: Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" on page 731.

Web Vuser scripts use VuGen's Snapshot pane.

- For details on how to work with the Snapshot pane, see "Work with Snapshots" on page 297.
- For details on the Snapshot pane UI, see "Snapshot Pane" on page 85.

The snapshots show detailed information about some of the steps in the Vuser script. Each snapshot can be displayed using either the **Page** view or the more detailed **HTTP Data** view.

The HTTP Data view displays each HTTP transaction in either a tree view or a grid view. The transaction data is broken up into response data, request data, headers, cookies, and query strings.




Data in the snapshots can be displayed in a number of formats: Data view, Text view, and Hex view.

You can split the Snapshot pane to display two snapshots - typically a record snapshot and the corresponding replay snapshot. If both snapshots are displayed in the HTTP Data view, you can click the Sync button on the Snapshot pane toolbar to synchronize the data that is displayed in the two snapshots. For details, see "Work with Snapshots" on page 297.

Correlations and parameters can be created on response data by selecting the desired text and right-clicking.

For data that is difficult to work with (such as binary data), VuGen offers a variety of Data Format Extensions that can transform certain data types into more readable formats. Data that has been formatted by a Data Format Extension can be displayed in its original or formatted state. For more information, see "Data Format Extensions (DFEs) - Overview" on page 741.

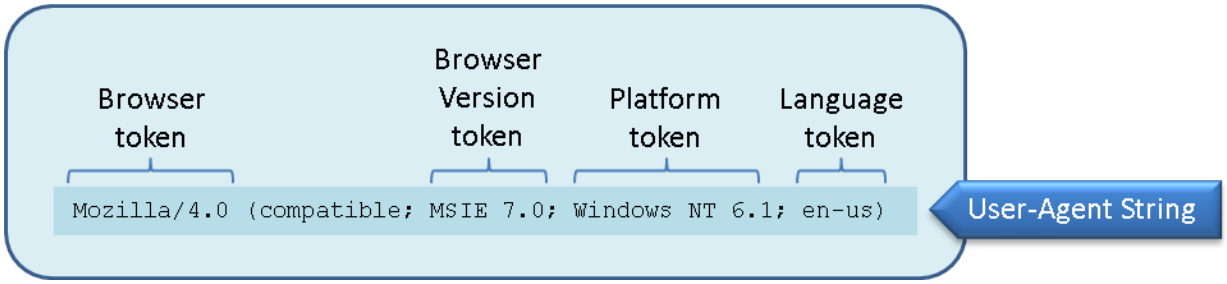
## Browser Emulation - Overview

> **Note: Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" on page 731.

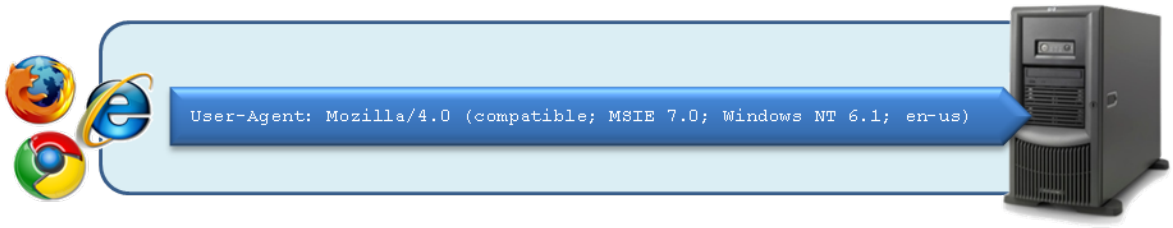
When you run a Web Vuser that accesses a web-site, the Vuser does not use an actual browser to access the site. Instead, the Vuser *emulates* a browser accessing the site. To enable the emulation, the Vuser uses a user-agent string.

## What is a User-Agent String?

When a browser sends a request to a server, the browser sends a **user-agent string** that identifies itself to the server. The identifying details in the user-agent string are included in various **tokens**. These tokens provide various details such as which browser is being used, the version of the browser, and the operating system on which the browser is running.




The user-agent string is included in a **User-Agent** header that is part of every request that is sent by the browser to the server. Servers can use the information that is contained in the user-agent string to provide content that is tailored for the specific browser.




Below is an example of a User-Agent header that contains a user-agent string that is sent to a server. In this example, the user-agent string identifies the browser as Internet Explorer 7.0, running under Windows 7.

```
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1)
```

## Emulating a specific browser

When you run a Web Vuser that accesses a web-site, the Vuser *emulates* a browser accessing the site. To enable the emulation, the Vuser creates a user-agent string that includes various tokens.

- The *Browser* and *Version* tokens define the browser to emulate and the version of the emulated browser.
- The *Platform* token defines the operating system on which the emulated browser is running.
- The *Language* token defines the language for which the emulated browser is localized.

The user-agent string is included in a **User-Agent** header in every request that the Vuser sends to the server that hosts the web-site.

## What information is included in Web Vuser's user-agent string?

1. When a Vuser script is first created, the browser in the user-agent string is Internet Explorer 6, and the operating system is Windows.
2. After the code in a Vuser script is generated, the browser in the user-agent string is changed to the browser that was used when the script was recorded, and the operating system is changed to the operating system that was used when the script was recorded.

After the code in a Vuser script is generated, you can use the Vuser script's runtime settings to specify an emulated browser that is different from the browser that was used when the script was recorded. There are scenarios in which this enables you to more accurately emulate the real-world situation. When you specify the emulated browser, you can specify the browser type and version, the platform on which the Vuser runs, and the language for which the browser is localized. VuGen creates a user-agent string that includes the details that you specify. If required, you can edit the user-agent string to create a customized user-agent string.

### To specify a different browser to emulate:

1. Open the Vuser script.
2. Click **Replay > Runtime Settings**.
3. In the runtime settings dialog box, click **Browser > Browser Emulation**, and then select **Use Browser**.
4. Select from the lists of available options to specify the browser to emulate.

> **Note:** The **User-Agent String** that is displayed is updated after each selection that you make.

### To customize the user-agent string:

1. Open the Vuser script.
2. Click **Replay > Runtime Settings**.
3. In the runtime settings dialog box, click **Browser > Browser Emulation**, and then select **Use Custom**.
4. Modify the user-agent string as required.

## Maximum number of concurrent connections

When a browser accesses a web-site, the browser maintains a number of concurrent connections with the web server that hosts the web-site. Therefore, when you access a web page that contains many different objects, such as images, Javascript files, frames, data feeds etc, the browser may try to improve performance by downloading several of the objects simultaneously. The maximum number of concurrent connections is dependent on the browser type, and the version of the browser. For example,

Internet Explorer 6 limits the number of concurrent connections to 2; Internet Explorer 8 and Firefox 3+ limit the number of concurrent connections to 6.

When you run a Web Vuser, by default, the maximum number of concurrent connections that the Vuser can maintain with a web-server is defined by the browser that is specified in the Vuser script's user-agent string.

### To override the default maximum number of connections value:

You can use the **MAX_CONNECTIONS_PER_HOST** option in the **web_set_sockets_option** function. For details, see the Function Reference (**Help > Function Reference**).

The following example sets the maximum number of concurrent connections to 10:

```
web_set_sockets_option("MAX_CONNECTIONS_PER_HOST", "10");
```

> **Note:** When you run a Web Vuser script, the maximum number of concurrent connections that the Vuser can maintain with the server appears in the Replay log.

## Perform Load Testing with nCipher HSM

This task describes how to set up a script to load test an environment with an nCipher HSM (Hardware Security Module).

1. **Prerequisite:** Generate the client certificate file (client_cert.pem) with the private key (client_key.pem) pointing to the private keys stored in the HSM. Make sure that you can connect to your Web server (A.com) with a generated CA file (ca-certs.pem). The successful openssl command should have the following form:

   ```
   openssl s_client –connect A.com:443 –CAfile ca-certs.pem –cert client_
   cert.pem –certform PEM –key client_key.pem – keyform PEM –engine CHIL
   ```

2. Set up the **PATH** environment variable:
   a. Add the **nfhwcrhk.dll** file, usually located in *C:\Program Files (x86)\nCipher\nfast\toolkits\hwcrhk*, to the PATH environment variable.
   b. Restart VuGen to apply the changes.
3. Enable nCipher key retrieval
   a. In the runtime settings, go to the **Internet Protocol > Preferences** view and expand the **Authentication** section.
   b. Select the **Enable retrieving keys from nCipher HSM** option. For details, see Preferences View - Internet Protocol.
4. Edit the Vuser script

   Add the following content to the **vuser_init** section of your Vuser script.

```
web_set_sockets_option(SSL_VERSION, "TLS");
web_set_sockets_option(DEFAULT_VERIFY_PATH, <full_path>\ca-certs.pem);
web_set_certificate_ex (
 "CertFilePath=<full_path_to_client_cert_file>/client_cert.pem",
"CertFormat=PEM",
 "KeyFilePath= <full_path_to_client_private_key_file> /client_key.pem",
"KeyFormat=PEM",
LAST);
```

## Working with Cache Data

> **Applies to:** Web Vuser protocols
> For a list of Web Vuser protocols, see "Web Vuser Types" on page 731.

Web browsers maintain a cache of objects that are downloaded by the browser. This lets the browser load objects faster when accessing a website because it uses objects from the cache instead of downloading the objects again.

When you run a Vuser script, by default, the Vuser starts with an empty cache. This implies that the Vuser must download each object the first time that the object is required. To better emulate a real-world situation, you can provide the Vuser with a pre-defined cache at any stage while the Vuser is running. The Vuser can then access required objects directly from the cache, without having to download them.

To provide a Vuser with a pre-defined cache, first you create a file that contains the cache, and then you load the cache into the Vuser. You use the **web_dump_cache** function to create the cache file, and the **web_load_cache** function to load the cache file into the Vuser.

### Create the cache file

You use the **web_dump_cache** function to create a cache file. The **web_dump_cache** function creates a file that contains all the objects that exist in the Vuser cache when the **web_dump_cache** function is executed. Insert the **web_dump_cache** function into a Vuser script, typically towards the end of the *Action* section of the script. This will ensure that the Vuser cache contains the required objects when the **web_dump_cache** function is run to create the cache file. After inserting the **web_dump_cache** function, run the script to build the Vuser cache and create the cache file.

> **Note:** You need to run the **web_dump_cache** function only once to generate the required cache file. Typically, this run is not part of a scenario. After the cache file has been created, when you run the Vuser script as part of a scenario, there is no need to execute the **web_dump_cache**

function. Therefore you should comment-out the **web_dump_cache** function in the Vuser script.

You use the **FileName** argument in the **web_dump_cache** function to specify the name and location of the file to create. The **FileName** path can be either absolute (e.g. "FileName=c:\\MyDir\\User1.cache") or relative to the current Vuser directory (e.g. "FileName=Iteration1.cache").

- **Absolute path names:** Use absolute path names if you do not want the cache file to be linked to the script. For example, if you wish to use a different cache on each host, use an absolute path.
- **Relative path names:** If you use a relative path name, the cache file is created inside the Vuser directory. When you copy the Vuser script, save it to a new location, or copy it to a load generator host, the cache file is also copied. Relative paths are independent of drive mappings and network locations.

The file extension of the cache file is always ".cache". The ".cache" extension is added if it is not specified in the **FileName** argument. For example, if you specify "FileName=Iteration2.txt", the cache file is called "FileName=Iteration2.txt.cache".

File names in the **FileName** argument can be parameterized so that different Vusers or different iterations can use different cache files. For example, "FileName=Iteration{param}.cache"

In the following example, the **web_dump_cache** function creates a cache file for each VuserName parameter running the script. The cache files are located in c:\temp.

**Example:**

```
web_dump_cache("paycheckcache","FileName=c:\\temp\\{Vuser
         Name}paycheck", "Replace=yes", LAST)
```

If you run a single Vuser user ten times, VuGen creates ten cache files in the following format, where the "Kunnn" prefix is the VuserName value:

```
Ku001paycheck.cache
Ku002paycheck.cache
Ku003paycheck.cache
...
```

## Load the cache file into a Vuser

The **web_load_cache** function loads a cache file into a Vuser. The **FileName** argument in the **web_load_cache** function specifies the name and location of the cache file to load. The specified cache file must exist before the **web_load_cache** function is executed. Therefore, you can run the **web_load_cache** function only after running the **web_dump_cache** function to create the cache file.

In the following example, the **web_load_cache** function loads the **{VuserName}paycheck** cache files from **c:\temp**.

**Example:**

```
web_load_cache("ActionLoad","FileName=c:\\temp\\{VuserName}paycheck",LAST)
```

### See also:

- For details on cache functions, see the Function Reference (**Help > Function Reference**).
- For details on how to implement cache functions, see "Insert Caching Functions" below.

## Insert Caching Functions

**Note: Applies to:** Web Vuser protocols
For a list of Web Vuser protocols, see "Web Vuser Types" on page 731.

Caching functions allow you to create a cache file that contains the Vuser cache, and then to load the cache file data into a Vuser.

1. Insert the **web_dump_cache** function into your Vuser script.
2. Run the script to create the cache file.
3. Insert the **web_load_cache** function into your script - before the Vuser actions.
4. Comment-out the **web_dump_cache** function.
5. Run and save the script.

### See also:

- Working with Cache Data
- Function Reference (**Help > Function Reference**)

# Data Format Extensions (DFEs) - Overview

Data Format Extension support—or DFE support for short—enables easier scripting of web applications by providing the ability to decode and encode formatted data that is exchanged between the client and the server. This enables easier correlation and parameterization of the generated Vuser scripts.

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

## When to use DFEs

When you record a Vuser script, VuGen records the HTTP requests and responses that are passed between the client and the web server. The data in these HTTP requests and responses is often

encoded. For example, some of the data may be in binary format. The encoded data may be in the HTTP query string, headers, body, or cookies. When the encoded data is included in a Vuser script, the resulting script will contain data that is difficult to decipher. This makes it difficult to identify text strings that can be used for parameterization and correlation.

The script segment below shows a section of a Vuser script that was generated while recording business processes on a GWT-based application. Notice how some sections of the script contain encoded data and are therefore difficult to decipher.

LoadRunner uses data format extensions (DFEs) to resolve the difficulties that arise from encoded data in Vuser scripts. DFE support allows easier creation of Vuser scripts by providing the ability to decode the encoded data that is exchanged between the client and the server. By providing the decoded format of the data, the information is presented in the Vuser script in a readable format that enables you to correlate and parameterize the script as required. When the script is replayed, the DFE support re-encodes the modified Vuser script, and enables the Vuser to send the correctly encoded request to the server.

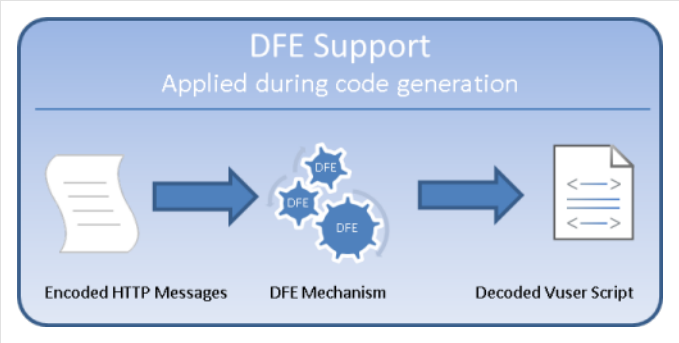
LoadRunner includes a number of pre-defined DFEs. Each DFE is able to decode and encode a specific type of data. For example, the GWT DFE decodes GWT data to XML format when a script is generated, and it encodes XML-formatted data to GWT-formatted data before the script is replayed. For a full list of the pre-defined DFEs, see "Data Format Extension List" on page 750.

When a DFE is applied to a Vuser script and the script is then regenerated, the DFE modifies the script and replaces the encoded data with decoded data. For details on how the script is modified, see "How DFEs Modify a Vuser Script" on page 749.

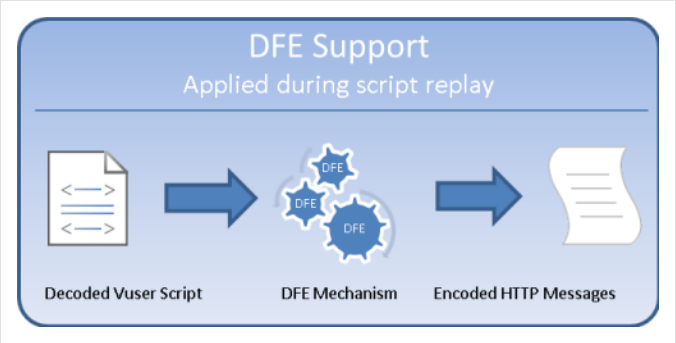
## DFE support

You enable DFE support for each Vuser script that requires decoding of encoded data. When DFE support is enabled, the DFE support is applied in the following circumstances:

- Each time the script is generated (after recording) or regenerated. The DFEs are applied to decode the encoded data to produce a script that is easy to decipher.




- Each time the script is run. The DFEs are applied to re-encode the decoded data to produce HTTP messages with encoded data in a format that is expected by the server.



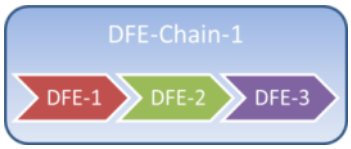

**Note:** In addition to applying DFEs when a script is generated or replayed, it is possible to apply a DFE to a selected string in a Vuser script. For details, see "Applying DFEs to a String" on page 751.

In some scenarios, decoding of encoded data must be performed in a number of stages, until the fully decoded data is produced. Each stage in the conversion process is performed by applying a specified DFE. For example, encoded data from a response may be decoded by applying three DFEs - first DFE-1, then DFE-2, and then DFE-3. In each stage, the output from one DFE is the input to the next DFE, until the fully decoded data is produced.

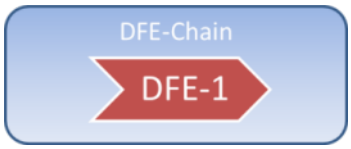
## DFE chains

The series of DFEs that are required to decode encoded data is defined in a **chain**. For example, you could create a chain called **DFE-Chain-1** that includes three DFEs: DFE-1, DFE-2, and DFE-3. The

sequence of the DFEs inside a chain is significant—the sequence indicates the order in which the DFEs are applied to the encoded data.




If only a single DFE is required to decode encoded data, the DFE must still be included in a chain.




## Assigning DFE chains

HTTP messages can be divided into a number of sections, including a body, headers, cookies, and a query string. After you define the DFE chains that will be applied to decode and encode a Vuser script, you must specify to which sections of the HTTP messages the DFE chains will apply. Because each HTTP message has only one **Body** section and one **Query String** section, you can specify only a single DFE chain to apply to each of these sections. In contrast, each HTTP message can contain numerous headers and cookies. Consequently, you can specify a particular DFE chain to apply to each header and cookie. For details, see "Apply DFE Chains to Sections of the HTTP Message" on page 748.

## Replaying Vuser scripts that contain DFEs

When you replay a Vuser script that contains DFE functionality, various messages are added to the Replay log in VuGen's Output pane. Make sure to check these messages to ensure that the DFE functionality is correctly implemented. For further details, see "Troubleshooting - Data Format Extension (DFE) " on page 761.

## See also:

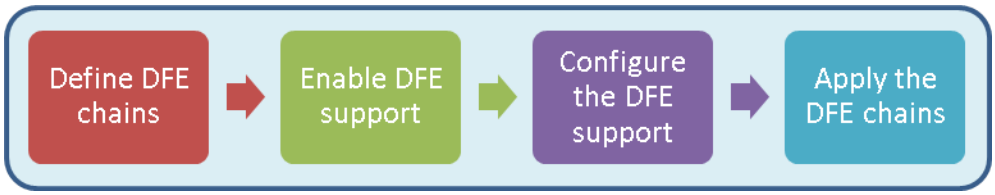
- "Implement Data Format Extension (DFE) Support" below
- "Implementing GWT-DFE Support" on page 759
- For details on creating custom DFEs (advanced users only), see the Software Support Knowledge Base.
- For details on using the VuGen JavaScript Engine to encode and decode data using common JavaScript libraries, see "Using the VuGen JavaScript Engine" on page 706.

## Implement Data Format Extension (DFE) Support

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

To implement the DFE support for a Vuser script, you must perform the steps shown in the diagram below:




## See also:

- "Data Format Extensions (DFEs) - Overview" on page 741
- "Implementing GWT-DFE Support" on page 759

## Define a Chain of DFEs

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

In order to implement DFE support for a Vuser script, you must first define the DFE chains that are applied by the Vuser. Defining the DFE chains is the first step in implementing DFE support.

### Adding a DFE chain

1. In VuGen, open the Vuser script.
2. Click **Record** > **Recording Options** > **Data Format Extension** > **Chain Configuration**.

   For details on the dialog box options, see "Data Format Extension > Chain Configuration Recording Options" on page 177.

3. Under **Chains**, click the **New Chain** button and create a new chain. The new chain is listed in the **Chains** pane.

### Adding DFEs to the new DFE chain

1. In the **Chains** pane, select the chain to which you want to add DFEs.
2. In the **Chain: <chain name>** pane, click the **Add DFE** button . For details on the **Chain: <chain name>** pane, see "Data Format Extension > Chain Configuration Recording Options" on page 177.
3. In the Add Data Format Extension dialog box, select the DFE that you want to add to the chain and click **OK**.

   When you add the **GWT** DFE or the **Prefix Postfix** DFE to a chain, you must supply additional configuration details about the DFE. For more information, see the documentation about the specific DFE.
4. After you add a DFE to the chain, select the appropriate option from the **Continue Processing** list. For details on the **Continue Processing** option, see "Data Format Extension > Chain Configuration Recording Options" on page 177.
5. Add additional DFEs to the chain as required.

### See also:

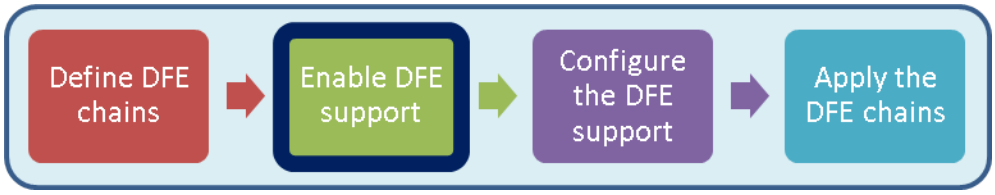
- "Data Format Extensions (DFEs) - Overview" on page 741
- "Enable DFE Support" below
- "Implement Data Format Extension (DFE) Support" on page 744

## Enable DFE Support

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

After you define the DFE chains that are available to a Vuser script, you must enable the DFE support, as described below. Enabling DFE support is the second step in implementing DFE support.




1. In VuGen, open the Vuser script.
2. Click **Record** > **Recording Options** > **Data Format Extension** > **Code Generation**.

   For details on the dialog box options, see "Data Format Extension > Chain Configuration Recording Options" on page 177.
3. Select the **Enable data format extension** check box.

## See also:

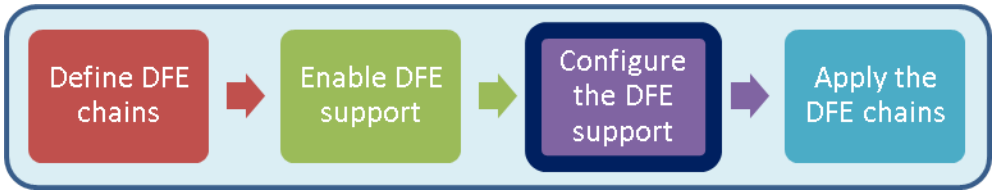
- "Configure DFE Support" below
- "Define a Chain of DFEs" on page 745
- "Implement Data Format Extension (DFE) Support" on page 744
- "Data Format Extensions (DFEs) - Overview" on page 741

## Configure DFE Support

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

After you enable DFE support for a Vuser script, you can configure the DFE support as described below. Configuring DFE support is the third step in implementing DFE support.




1. Open the Vuser script in VuGen.
2. Click **Record** > **Recording Options** > **Data Format Extension** > **Code Generation**.

   For details on the dialog box options, see "Data Format Extension > Code Generation Recording Options" on page 180.
3. Under **Configuration**, from the **Format** list, select the parts of the Vuser script to which the DFEs will be applied.
   - **Code and snapshots**. Applies DFEs to convert the Vuser script code and the snapshot data.
   - **Snapshots**. Applies DFEs to convert the snapshot data only - not the Vuser script code.
4. Select the **Verify formatted data** check box to check the results of the data conversion by converting the converted data back to the original state, and then verifying that it matches the original data.

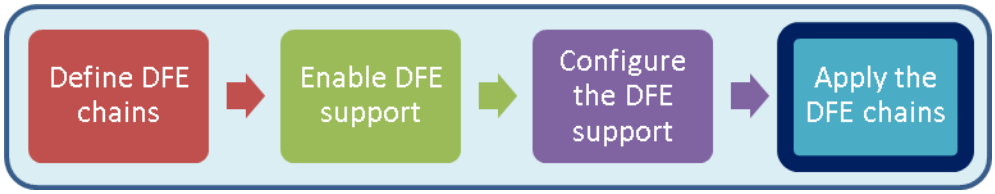
- For details on the dialog box options, see "Data Format Extension > Code Generation Recording Options" on page 180.
- After configuring the DFE support, you can assign the DFE chains to specific sections of the HTTP messages. For details, see "Apply DFE Chains to Sections of the HTTP Message" on the next page.
- For an overview of the process of implementing DFE support, see "Implement Data Format Extension (DFE) Support" on page 744.
- For additional details about DFEs, see "Data Format Extensions (DFEs) - Overview" on page 741.

# Apply DFE Chains to Sections of the HTTP Message

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

After you have defined the DFE chains that are available to a Vuser script, enabled and configured DFE support, you must define to which sections of the HTTP messages to apply the DFE chains, as described below. Applying DFE chains to message sections is the last step in implementing DFE support.




1. Open the Vuser script in VuGen.
2. Click **Record** > **Recording Options** > **Data Format Extension** > **Code Generation**.

   For details on the dialog box options, see "Data Format Extension > Chain Configuration Recording Options" on page 177.
3. [Optional] In the <**message section**> pane, click **Body** add then select the chain that will be applied to the message body.
4. [Optional] In the <**message section**> pane, click **Headers** add then select the chains that will be applied to the message headers.

   **Note:** For VuGen to correctly assign the chain to a specific header, the entry in the **Name** column must be exactly the same as the name of the header in the message.
5. [Optional] In the <**message section**> pane, click **Cookies** add then select the chains that will be applied to the message cookies.

   **Note:** For VuGen to correctly assign the chain to a specific cookie, the entry in the **Name** column must be exactly the same as the name of the cookie in the message.
6. [Optional] In the <**message section**> pane, click **Query String** add then select the chain that will be applied to the message query string.

**Note:** Whereas you can modify only the default chain for the **Body** and **Query String** sections, you can add multiple chains for the **Headers** and **Cookies** sections.

### See also:

- "Implement Data Format Extension (DFE) Support" on page 744
- "Data Format Extensions (DFEs) - Overview" on page 741

## How DFEs Modify a Vuser Script

> **Applies to:**
>
> - Web - HTTP/HTML Vuser scripts
> - Web - HTTP/HTML steps inside Flex Vuser scripts

When a DFE is applied to a Vuser script and the script is then regenerated, the DFE causes various modifications to be made to the script, as follows:

- VuGen replaces the original encoded text string with a parameter.
- VuGen inserts a **web_convert_from_formatted** function before the function that contains the new parameter. The **web_convert_from_formatted** function contains the decoded value of the original encoded text.

The script section below shows a **web_custom_request** function that was generated without DFE support. The **Body** tag in the function includes a text string that is base64 encoded, **Body=TW9uZGF5**. Because the value of the **Body** tag is encoded, it is difficult to change its value if required for correlation or parameterization purposes.

```
web_custom_request("echo_post.asp",
    "URL=http://example.devlab.ad/cgi-bin/temp/echo_post.asp",
    "Method=POST",
    "Resource=0",
    "RecContentType=text/html",
    "Referer=",
    "Snapshot=t1.inf",
    "Mode=HTTP",
    "Body=TW9uZGF5",
    LAST);
```

After applying Base64 DFE support, the value of the **Body** tag in the regenerated script is replaced with a parameter called DFE_PARAM, "**Body={DFE_PARAM}**, as shown below.

```
/*TODO: A Correlation scan needs to be performed.*/
    web_convert_from_formatted("FormattedData/EscapedBinary=<HP_EXTENSION name=\"Base64\">Monday</HP_EXTENSION>",
        "TargetParam=DFE_PARAM",
        LAST);

    web_custom_request("echo_post.asp",
        "URL=http://example.devlab.ad/cgi-bin/temp/echo_post.asp",
        "Method=POST",
        "Resource=0",
        "RecContentType=text/html",
        "Referer=",
        "Snapshot=t1.inf",
        "Mode=HTTP",
        "Body={DFE_PARAM}",
        LAST);

    return 0;
```

In addition, the modified code also includes a **web_convert_from_formatted** function. The function indicates that the decoded value of the originally encoded string is **Monday**. It is now simple to change the value from **Monday** to any other day by simply changing the decoded value in the **web_convert_from_formatted** function.

## Data Format Extension List

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

The following table lists the pre-defined LoadRunner DFEs (Data Format Extensions). For more information about DFEs, see "Data Format Extensions (DFEs) - Overview" on page 741.

| Data Format Extension | Description |
|---|---|
| Base64 Extension | Decodes strings that are encoded with a Base64 encoder. |
| Binary to XML Extension | Transforms Microsoft WCF binary XML into XML format. |
| GWT Extension | Transforms GWT data to XML format. |
| Java To XML Extension | Transforms serialized Java objects to XML. |

| | |
|---|---|
| JSON to XML Extension | Transforms JSON data to XML format. |
| Prefix Postfix Extension | Enables you to cut data from the beginning and/or end of a string which you do not want decoded. You can add and customize as many prefix/postfix extensions as required. Each postfix/prefix extension created should have a unique display name and tag name. |
| Remedy to XML Extension | Transforms Remedy request data into XML format.<br>Note: This extension does not transform Remedy response data - which is JavaScript code. |
| URL Encoding Extension | Decodes strings that are encoded with URL encoding format. |
| XML Extension | Receives data and checks to see if it conforms with XML syntax. This check allows VuGen to perform correlations based on XPath and to display snapshot data in an XML viewer. |
| XSS Extension | Enables you to test sites that use Cross Site Scripting (XSS) defense code. |

## Applying DFEs to a String

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

You can apply DFEs to a Vuser script to decode encoded data in the script. You can apply the DFEs:

- to specified sections of the Vuser script, when the script is generated or regenerated. For details, see "Implement Data Format Extension (DFE) Support" on page 744.
- to a single encoded string in the Vuser script, as described in this topic.

For an overview of DFEs, see "Data Format Extensions (DFEs) - Overview" on page 741.

**Note:** After you apply a DFE to a string in a Vuser script, VuGen adds an entry to the **Tasks** tab. The added entry indicates that a correlation scan should be performed. For details on performing a correlation scan, see "Correlating" on page 251.

To apply a DFE to an encoded string in a Vuser script:

1. Open the script in VuGen, and select the encoded text string.
2. Right-click inside the selection, click **Decode with DFE**, and click the name of the chain that contains the DFEs to decode the encoded string. For details on how to define a DFE chain, see "Define a Chain of DFEs" on page 745.

   VuGen replaces the selected text with a parameter, and adds a **web_convert_from_formatted** function that contains the decoded equivalent of the originally selected text.

> **Note:** To change the name that VuGen assigns to new parameters, right-click some encoded text in the Vuser script, click **Decode with DFE > Advanced**, and then specify the parameter name in the **Target Parameter** box. VuGen will add a counter to the parameter name, and increment the counter as required.

## Google Web Toolkit - Data Format Extension (GWT-DFE)

The Google Web Toolkit (GWT) is a set of tools that allow web developers to create complex JavaScript front-end applications. The code is developed in Java and is open source. GWT supports, among other features: asynchronous remote procedure calls, history management, bookmarking, UI abstraction, internationalization, and cross-browser portability.

> **Applies to:**
>
> - Web - HTTP/HTML Vuser scripts
> - Web - HTTP/HTML steps inside Flex Vuser scripts

### Overview

The GWT-DFE support helps to generate Vuser scripts for GWT-based web sites that use the GWT-RPC mechanism. When you record a GWT-RPC based web site without enabling the GWT-DFE support, the resulting Vuser script may contain significant amounts of data that is cryptic and therefore difficult to decipher, as shown in the code segment below:

```
web_custom_request("gwtservice",
    "URL=http://lazarus.devlab.ad:8081/GwtComplexObject/org.ega.Main/sampleservice/gwtservice",
    "Method=POST",
    "Resource=0",
    "RecContentType=application/json",
    "Referer=http://lazarus.devlab.ad:8081/GwtComplexObject/",
    "Snapshot=t3.inf",
    "Mode=HTML",
    "EncType=text/x-gwt-rpc; charset=utf-8",
    "Body=6|0|7|http://lazarus.devlab.ad:8081/GwtComplexObject/org.ega.Main/|99EB9620EEB0D48791FBC5BF95BC6368|"
    "org.ega.client.sampleservice.GWTService"
    "|myMethod|org.ega.client.data.InputData/74817998|LoadRunner|11.52|1|2|3|4|1|5|5|6|7|",
    LAST);
```

The cryptic formatting of the data makes it difficult to identify text strings to be used for correlation, parameterization, and verification.

> **Tip:** The presence of numerous pipes in the recorded data indicates that the recorded site may be a GWT-based web site that uses the GWT-RPC mechanism.

When you enable GWT-DFE support, VuGen is able to decode much of the complex data in the HTTP responses and requests. This enables VuGen to generate Vuser scripts that contain data in XML format. In addition, the original coded data contains only values, without the associated names of the data fields. After applying GWT-DFE, the resulting XML-formatted data includes both the names and the values of the data fields. The XML-formatted data in the scripts is therefore easier to decipher, making the scripts easier to correlate, parameterize, and use for verification purposes.

To enable VuGen to decode the complex data in the HTTP communication, you must identify the *.war* file that is used by the web application. Occasionally, an application uses *.jar* files. All considerations below about *.war* files apply equally to *.jar* files when used instead of a *.war* file.

The *.war* file contains the logic used by GWT to encode and decode the information in the HTTP communication. VuGen needs access to the *.war* file so that VuGen can perform similar encoding and decoding procedures. Typically, these *.war* files are located on the application server, under the web applications folder.

> **Note:** Make sure that the *.war* file that you associate with the Vuser script is the most up-to-date *.war* file for your application. The *.war* file is changed each time changes are made to the web application. GWT-DFE support will function correctly only if the most up-to-date *.war* file is available.

- For an introduction to using DFEs in Vuser scripts, see "Data Format Extensions (DFEs) - Overview" on page 741.
- For details on how to implement DFE support, see "Implement Data Format Extension (DFE) Support" on page 744.
- For a full list of the supported DFEs, see "Data Format Extension List" on page 750.

> **Note:**
>
> - GWT-DFE provides an automatic solution for GWT specific (STRONG_NAME_HEADER) correlations.
> - VuGen is able to parse Client Fields with GWT-DFE.

Example of code generated with and without GWT-DFE support

| **Original Script - without GWT-DFE Support** |
| --- |
| 6\|0\|11\|http://localhost:8081/MyTestApp/testapp/\|624C899BB846618A2E7F49092<br><br>8212946\|com.test.client.GreetingService\|greetServeCompAns\|com.test.client.ComplexObject/198661839<br><br>\|GWT User\|inside object\|java.util.HashSet/1594477813\|java.lang.String/2004016611\|add string1\|<br><br>string2\|1\|2\|3\|4\|1\|5\|5\|1001\|1999\|6\|5\|321\|1234\|7\|0\|8\|0\|8\|2\|9\|10\|9\|11\| |

**Script after applying GWT-DFE Support**

```
<HP_EXTENSION name="GWT_DFE_1">
<com.hp.dfe.GWT__Request>
<moduleBaseURL>http://localhost:8081/MyTestApp/testapp/</moduleBaseURL>
<rpcRequest>
<flags>0</flags>
<method>
<class>com.test.client.GreetingService</class>
<name>greetServeCompAns</name>
<parameter-types>
<class>com.test.client.ComplexObject</class>
</parameter-types>
</method>
<parameters>
<com.test.client.ComplexObject>
<anIntField>1001</anIntField>
<anotherIntField>1999</anotherIntField>
<name>GWT User</name>
<objectInComposingField>
<anIntField>321</anIntField>
<anotherIntField>1234</anotherIntField>
<name>inside object</name>
<stringsSet/>
</objectInComposingField>
<stringsSet>
<string>add string2</string>
<string>add string1</string>
</stringsSet>
</com.test.client.ComplexObject>
</parameters>
<serializationPolicy
class="com.google.gwt.user.server.rpc.impl.StandardSerializationPolicy"/>
```

| **Script after applying GWT-DFE Support** |
| --- |
| </rpcRequest><br></com.hp.dfe.GWT__Request><br></HP_EXTENSION> |

### Auto-detection of GWT Remote Procedure Calls (RPCs)

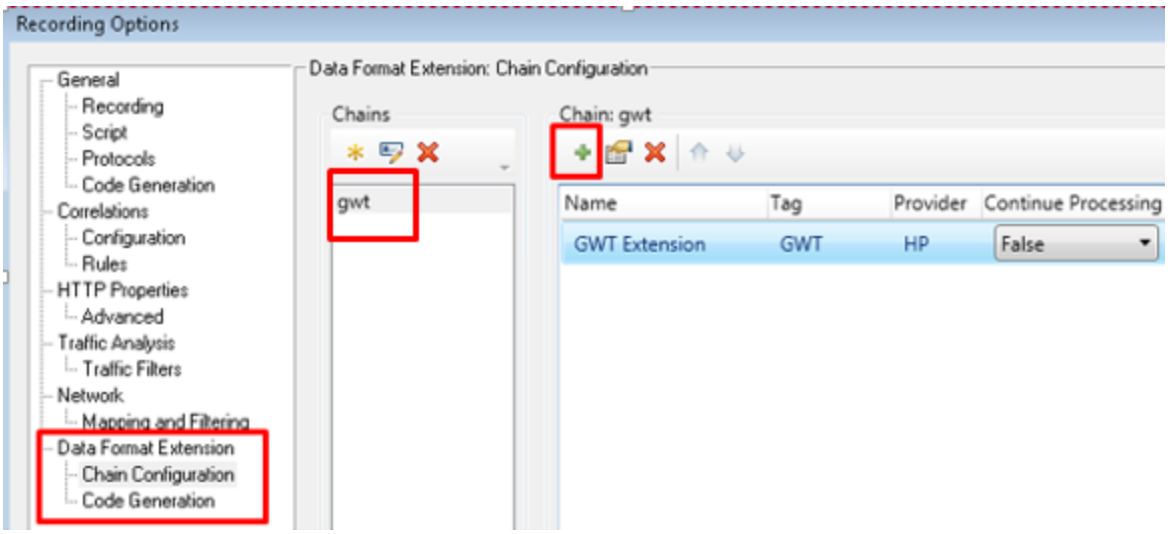
When VuGen generates or regenerates a Vuser script, VuGen scans the HTTP headers in the requests that are sent to the server. If VuGen detects both a **x-gwt-module-base** text string and a **x-gwt-permutation** text string in any of these HTTP headers, VuGen displays a warning in the VuGen Error tab. The warning recommends that you enable GWT-DFE for the Vuser script.

> **Note:** VuGen will continue to issue the above warning - each time the script is generated or regenerated - until the GWT-DFE is enabled.

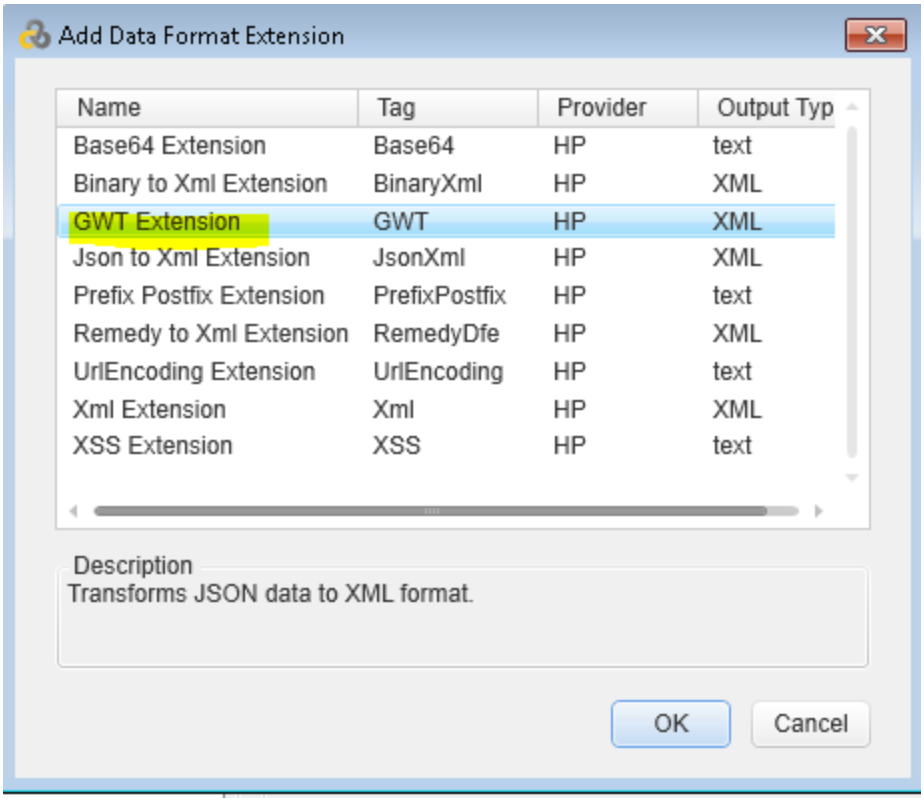
### How to apply a GWT Data Format Extension to your script before recording

To apply GWT-DFE before recording, perform the following actions:

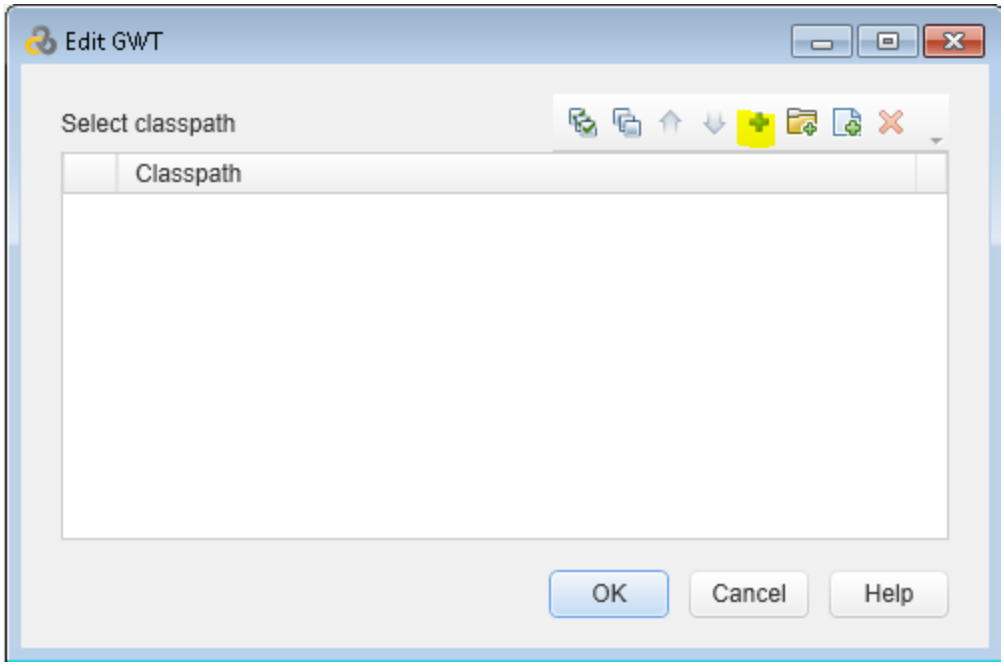
1. Open Recording Options (VuGen --> File --> Recording options)
2. Go to the "Data Format Extension --> Chain Configuration" tab
3. Add a new chain (press the green '+' sign)



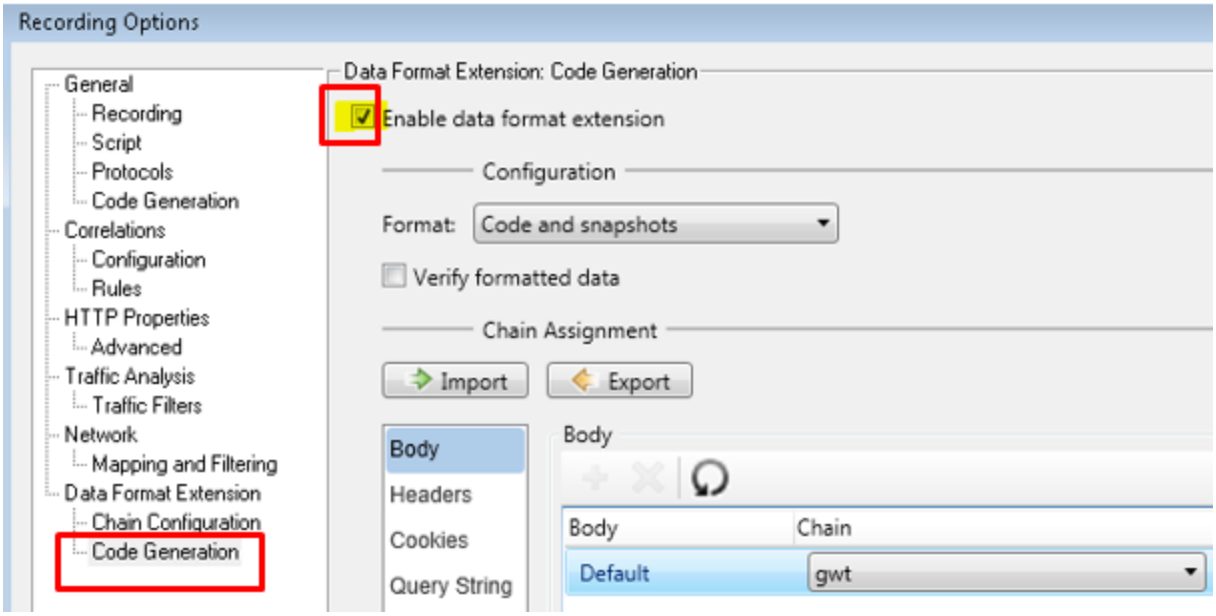

4. In the dialog that opens, select **GWT Extension** and click **OK**.




5. A new window will then open.
   To add the most recent WAR / JAR files, press the '+' button and add them from the file system.




6. Enable Data Format Extension




7. Record or regenerate the GWT script.

After generating the script using GWT-DFE, you will get a readable LoadRunner API that includes the request inside an XML.

## Tips and troubleshooting

The exact same versions of the war and jar files that are on the server must be used by LoadRunner during code generation and replay.

If there are differences, LoadRunner gives an indication during Code Generation and Replay:

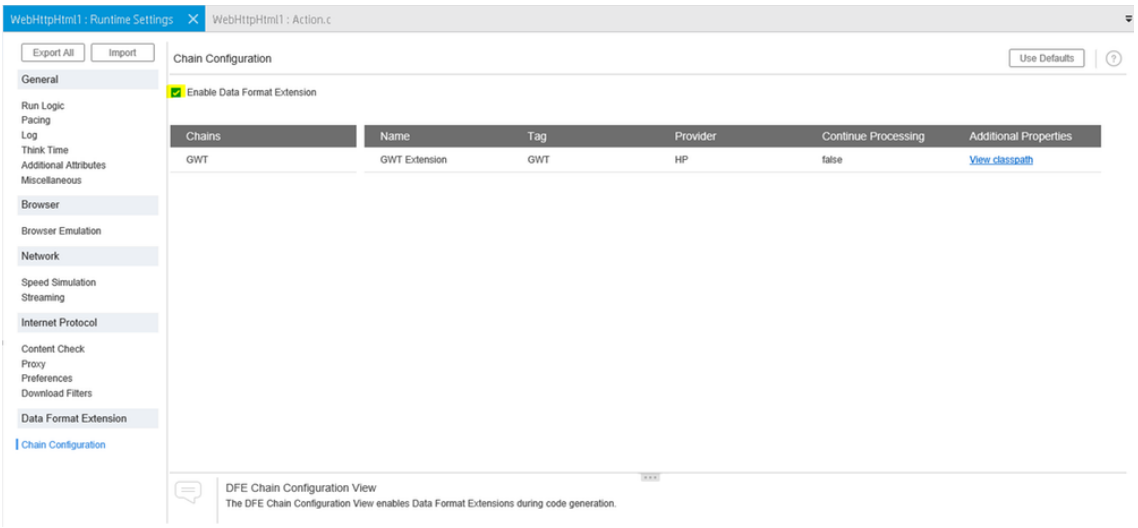
The follow errors can occur if the war and jar files do not match:

- Class not found
- Failed to handle WAR entry.
- Failed to de-serialize method
- Missing method
- The correct GWT serialization policy could not be found.

To fix issues, ensure that:

- The GWT DFE is enabled and the chains are configured correctly, including all .JAR, .CLASS and .GWT.RPC files used by the GWT application. After configuring the Chain, regenerate the script.
- The Data Format Extension is enabled in runtime settings **Data Format -> Chain Configuration ->**

**Enable Data Format Extension**.




## See also:

- "Implement Data Format Extension (DFE) Support" on page 744
- "Implementing GWT-DFE Support" below
- Blog: Introduction to GWT (Google Web Toolkit) DFE (Data Format Extension) in LoadRunner
- Blog: The technical aspects of GWT (Google Web Toolkit) DFE (Data Format Extension) in LoadRunner

## Implementing GWT-DFE Support

**Applies to:**

- Web - HTTP/HTML Vuser scripts
- Web - HTTP/HTML steps inside Flex Vuser scripts

This topic provides additional information about implementing DFE support, as described in "Implement Data Format Extension (DFE) Support" on page 744.

### Prerequisites for implementing GWT-DFE support

LoadRunner includes OpenJDK 7 JRE. However, some applications may be compiled for JVM1.8 and higher. If your application is compiled with JDK1.8 or higher, replace the <;LoadRunner>\lib\openjdk32\jre folder with your own OpenJDK 8 or higher JRE before recording a GWT application.

### Recording GWT-DFE Headers

As part of the GWT-DFE support implementation process, it is necessary to specify that VuGen record **x-gwt-permutation** headers when recording business processes. This procedure, as described below,

can be performed at any stage of the GWT-DFE support implementation process.

1. Create a Web - HTTP/HTML protocol script .
2. Select **Record > Recording Options > HTTP Properties > Advanced** and then click **Headers**.
3. In the Headers dialog box, select **Record headers in list**.
4. From the **Headers** list, select the **x-gwt-permutation header** check box.

Applying GWT-DFE chains

When you implement any DFE support, you must apply the DFE chains to specific sections of the HTTP communication. The basic process is described in "Apply DFE Chains to Sections of the HTTP Message" on page 748. This topic includes information required when assigning chains while implementing GWT-DFE support.

When you assign the chains while implementing GWT-DFE support, you must specify the classpath entries that are associated with the application that is operated by the Vusers. To assign the classpath entries, you must have access to the GWT WAR folder that is used by your development team. The WAR folder includes the following file types:

- *.gwt.rpc files
- *.jar files
- *.class files

**Specifying the classpath entries**

1. Select **Record > Recording Options > Data Format Extension > Chain Configuration**.
2. Under **Chains**, click to create and name a new DFE chain.
3. Click .
4. In the Add Data Format Extension dialog box, select **GWT Extension** and click **OK**.
5. In the **Add GWT** dialog box, specify the classpath entries:
   a. If the classpath entries are contained in a single *.war* file, click and then specify the location of the *.war* file.

> **Note:** If you have write permissions in the folder containing the .war file, it automatically creates a new folder with the extracted contents. It adds the specific classes/jars to VuGen in the following structure:
>
> <ServerDir>\<applicationDir>\<MyGWTApplication>\<SomeDirContaining .gwt.rpc file>
>
> - WEB-INF\classes
> - WEB-INF\lib\gwt-servlet.jar

- WEB-INF\lib\gwt-servlet-deps.jar
- WEB-INF\lib\log4j.jar
- WEB-INF\lib\<AdditionalAUTRelatedJarFile>.jar

If you do not have write access, it will just add the .war file without extracting its contents.

b. If the classpath entries are not contained in a single *.war* file:

- Click to add the folder that contains .gwt.rpc files.
- Click to add the application **classes** folder.
- Click to add the application **JAR** files from the **WEB-INF\lib** folder.

**Note:** If the location of the classpath entries is not the same on the computer on which the script was recorded and the computer on which the script will be replayed, then you must modify the runtime settings for the script. Select **Replay > Runtime Settings > Data Format Extension > Chain Configuration** and specify the location of the classpath entries on the computer on which the script will be replayed.

6. Select **Recording Options > Data Format Extension > Code Generation**.
7. Select the **Enable data format extension** check box.
8. Under **Configuration**, select **Code and snapshots** from the **Format** list.
9. Under **Chain Assignment**, select **Body** and select a chain. Select **Headers** and choose the same chain.
10. Click **OK**.

## See also:

- "Data Format Extensions (DFEs) - Overview" on page 741
- Blog: Introduction to GWT (Google Web Toolkit) DFE (Data Format Extension) in LoadRunner
- Blog: The technical aspects of GWT (Google Web Toolkit) DFE (Data Format Extension) in LoadRunner

## Troubleshooting - Data Format Extension (DFE)

This section describes troubleshooting tasks for Vuser scripts that contain DFE functionality.

- You may encounter the following errors with DFE functions on 64-bit Linux environments:

```
Error -27040: Data Format Extension: Creating custom chain failed: Extension
"UrlEncoding" was not found.
Error -35063: The "DFEs" argument is invalid. Check that the provided extensions
have their configuration files defined.
```

  **Solution:** Install the 32-bit version of **keyutils-libs.so** (keyutils-libs.i686) on your system, if it does not already exist.

  If you still encounter issues, it may be because the DFEs you are using are not supported on Linux. In this case, disable DFE support in the script, and remove all DFE steps from the script.

- **Replay log: Warning -27040: Data Format Extension: Convert: Empty string returned from extension {Extension name}**
  When you replay a Vuser script that contains DFE functionality, various messages are added to the Replay log in VuGen's Output pane. The above message indicates that when the Vuser script was replayed, the result of the specified **web_convert_from_formatted** step in the script was an empty string. For some DFEs, returning an empty string from a **web_convert_from_formatted** step is the correct behavior. However, if the Vuser script includes GWT-DFE functionality, the above message may indicate one or both of the following:
  - Some of the required classpath files are not included in the runtime settings for the Vuser script.
  - Some of the required classpath files do not exist in the specified location on the Load Generator.

  For details on how to resolve these issues, see "Implementing GWT-DFE Support" on page 759.

- If you have implemented your own version of DFE, the definition of class HTTPEntity in DfeDefinitions.h file has been updated in LoadRunner 11.50. No code change is required, but all DFE extensions need be recompiled.

# Web Services Protocol

## Web Services - Adding Script Content

### Web Service Testing Overview

SOA systems are based on Web Services, self-contained applications that can run across the Internet on a variety of platforms. The services are built using Extensible Markup Language (XML) and Simple Object Access Protocol (SOAP). They serve as building blocks, thereby enabling the rapid development and deployment of new applications.

Using VuGen, you create Vuser scripts for testing your SOA environment. You can use a test generation wizard to automatically generate scripts, or you can create the scripts manually.

### Adding Web Service Script Content - Overview

Web Services scripts let you test your environment by emulating Web Service clients.

After creating an empty Web Services script, as described in "Create a New Script Dialog Box" on page 150, you add content through one of the following methods: recording, manually inserting Web Service calls, importing SOAP, or by analyzing server traffic.

### Recording a Web Services Script

By recording a Web Services session, you capture the events of a typical business process. If you have already built a client that interacts with the Web Service, you can record all of the actions that the client performs. The resulting script emulates the operations of your Web Service client. After recording, you can add more Web Service calls and make other enhancements to the script.

When you record an application, you can record it with or without a Web Service WSDL file. If you include a WSDL file, VuGen allows you to create a script by selecting the desired methods and entering values for their arguments.VuGen creates a descriptive script that can be updated when there are changes in the WSDL.

If you record a script without previously importing a service (not recommended) VuGen creates SOAP requests instead of Web Service call steps. SOAP request arguments are less intuitive and harder to maintain.

For more information, see "Add Content" on page 771.

### Adding New Web Service Calls

You can create a Web Services script by manually adding Web Service calls. You design the call based on operation, transport, arguments, and other properties.

For more information, see "Add Content" on page 771.

### Importing SOAP Requests

VuGen lets you create Web Service calls from SOAP files. If you have a SOAP request file, you can load it directly into your script. VuGen imports the entire SOAP request (excluding the security headers) with the argument values as they were defined in the XML elements. By importing the SOAP, you do not need to set argument values manually as in standard Web Service calls.

For example, suppose you have a SOAP request with the following elements:

```
- <soap:Body soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  - <q1:AddAddr xmlns:q1="http://tempuri.org/AddrBook/message/">
            <Addr href="#id1" />
    </q1:AddAddr>
   - <q2:Addr id="id1" xsi:type="q2:Addr"
xmlns:q2="http://tempuri.org/AddrBook/type/">
        <name xsi:type="xsd:string">Tom Smith</name>
             <street xsi:type="xsd:string">15 Elm Street</street>
             <city xsi:type="xsd:string">Pheonix</city>
            <state xsi:type="xsd:string">AZ</state>
            <zip-code xsi:type="xsd:string">97432</zip-code>
            <phone-numbers href="#id2" />
```

```
            <birthday xsi:type="xsd:date">1983-04-22</birthday>
     </q2:Addr>
...
```

When you import the SOAP request, VuGen imports all of the values to the Web Service call. To view the values, in the Step Navigator, right-click the step and then click **Show Arguments**.

To create a new Web Service call based on a SOAP request, you must first import a WSDL file. If a WSDL is not available, or if you want to send the SOAP traffic directly, you can create a SOAP Request step. You specify the URL of the server, the SOAP action, and the response parameter.

In the Editor, the SOAP Request step appears as a **soap_request** function, described in the Function Reference (**Help > Function Reference**).

For more information, see "Add Content" on page 771.

### Analyzing Server Traffic

The main focus when testing enterprises and complex systems, is to measure the performance from the client end. Ordinarily, VuGen records the actions you perform in the application or browser, and generates a script emulating the client actions and requests to the server.

In certain test environments, you may be unable to record the client application to retrieve the requests to the server. This may be a result of the server acting as a client, or because you do not have access to the client application. In these cases, you can create a script using VuGen's **Analyze Traffic** feature.

The **Analyze Traffic** feature examines a capture file containing the server network traffic, and creates a script that emulates requests sent to or from the server.

For more information, see "Create a Script by Analyzing Traffic (Web Services)" on page 775.

## Script Integration

You can use the completed script to test your system in several ways:

- **Functional Testing.** Run the script to see if your Web services are functional. You can also check to see if the Web service generated the expected values. For more information, see "Web Services - Preparing Scripts for Replay" on page 791.
- **Load Testing.** Integrate the script into a LoadRunner Controller scenario to test the performance of your system under load. For more information, see the *Controller* documentation.
- **Production Testing.** Check your Web service's performance over time through a Business Process Monitor configuration. For more information, see the *Business Process Monitor* documentation.

## Web Service Call Attachments

When transferring binary files such as images over SOAP, the data must be serialized into XML. Serialization and deserialization can cause a significant amount of overhead. Therefore, it is common to send large binary files using an attachments mechanism. This keeps the binary data intact, reducing the parsing overhead.

Using attachments, the original data is sent outside the SOAP envelope, eliminating the need to serialize the data into XML and making the transfer of the data more efficient.

The formats used for passing a SOAP message together with binary data are MIME (Multipurpose Internet Mail Extensions) and the newer, more efficient DIME (Direct Internet Message Encapsulation) specifications. VuGen supports DIME for all toolkits, but MIME only for the Axis toolkit. To use MIME attachments for the .NET toolkit, see "User Handler Examples" on page 799.

VuGen supports the sending and receiving of attachments with SOAP messages. You can send Input (Request) or save Output (Response) attachments. For task details, see "Add Content" on page 771.

Output attachments are used to save the response as an attachment. You can choose one of the following options: **Save All Attachments** or **Save Attachment by Index**.

When you specify **Save All Attachments,** VuGen creates three parameters for each attachment based on the parameter name that your specify: a parameter containing the attachment data, the content type of the attachment, and a unique ID for the attachment.

For example, if you specify the name **MyParam** in the **Content** field, the parameter names for the first attachment would be:

```
MyParam_1
MyParam_1_ContentType
MyParam_1_ContentID
```

When you specify **Save Attachments by Index**, you specify the index number and name of the parameter in which to store the attachment. The parameter name that you specify for **Content**, is used as a prefix for the Content type and Content ID parameters.

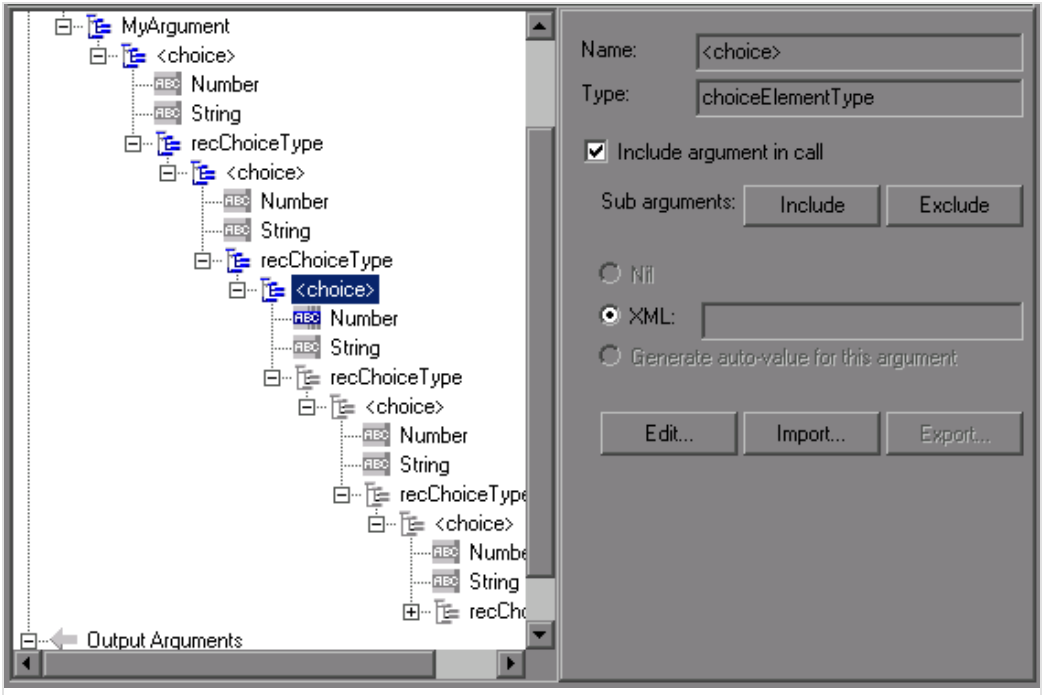
## Special Argument Types

VuGen handles special argument types, as follows:

| Argument Type | Description |
|---|---|
| **Derived Types** | Supported for WSDLs. When setting the properties for a Web Service Call, VuGen allows you to use the base type or derived type for the argument. After you select a type, VuGen updates the argument tree node to reflect the new type. For details, see "New Web Service Call Dialog Box" on page 778. |

| Argument Type | Description |
|---|---|
| **Abstract Types** | A declaration type declared by the programmer. When an element or type is declared to be **abstract**, it cannot be used in an instance document. Instead, a member of the element's substitution group, provided by the XML schema, must appear in the instance document. In such a case, all instances of that element must use the **xsi:type** to indicate a derived type that is not abstract.<br><br>When VuGen encounters an Abstract type, it cannot create an abstract class and replay will fail. In this case, VuGen displays a warning message beneath the **Type** box, instructing you to replace the Abstract type with a derived type. |

| Argument Type | Description |
|---|---|
| **Optional Elements** | In WSDL files, optional parameters are defined by one of the following attributes:<br><br>`minoccurs='0'`<br><br>`nillable='true'`<br><br>**minoccurs = 0** indicates a truly optional element that can be omitted. **Nillable** means that the element can be present without its normal content, provided that the nillable attribute is set to 'true' or 1. By default, the **minoccurs** and **maxoccurs** attributes are set to 1.<br><br>In the following example, **name** is mandatory, **age** is optional, and **phone** is nillable.<br><br>`<s:element minOccurs="1" name="name" type="s:string" />`<br>`<s:element minOccurs="0" name="age" type="s:int" />`<br>`<s:element minOccurs="1" name="phone" nillable="true" type="s:string" />`<br><br>## Option availability per parameter type<br><br>**Mandatory**<br><br>• Nil radio button: disabled<br>• Include arguments in call: disabled<br><br>**MinOccurs=0**<br><br>• Nil radio button: disabled<br>• Include arguments in call: enabled<br><br>**Nillable**<br><br>• Nil radio button: enabled<br>• Include arguments in call: disabled<br><br>## To include a specific optional argument in the service call:<br><br>Click the node and select **Include Argument in Call**. The nodes for all included arguments are colored in blue. Arguments that are not included are colored in gray.<br><br>If you include an element on a parent level, it automatically includes all mandatory and nillable children elements beneath it. If it is a child element, then it automatically includes the parent element and all other mandatory or nillable elements on that level. If you specify **Generate auto-value** to a parent element, VuGen provides values for those child elements that are included beneath the parent. |

| Argument Type | Description |
|---|---|
| | **Note:** VuGen interprets whether elements are mandatory or optional through the toolkit implementation. This may not always be consistent with the element's attributes in the WSDL file. |
| **Choice Optional Elements** | In a WSDL, defines a set of elements where only one of them appears in the SOAP message. In some cases, one of the Choice elements is optional, while the others are not. You can select the Choice element and still prevent its optional element from appearing in the SOAP envelope. In the **Step Navigator**, select the Choice element, and clear the **Include argument in call** option. In Script view, delete the line that defines the Choice argument. |
| **Recursive Elements** | Using the Properties dialog box, you can control the level of recursive elements to include in the Web Service call.<br><br>To exclude a certain level and exclude those below, select the lowest parent node that you want to include and select **Include Argument in Call**. VuGen includes the selected nodes, its mandatory children, and all of its parent nodes.<br><br>In the following example, three levels of the Choice argument are included—the rest are not. Excluded nodes are grayed out.<br>  |

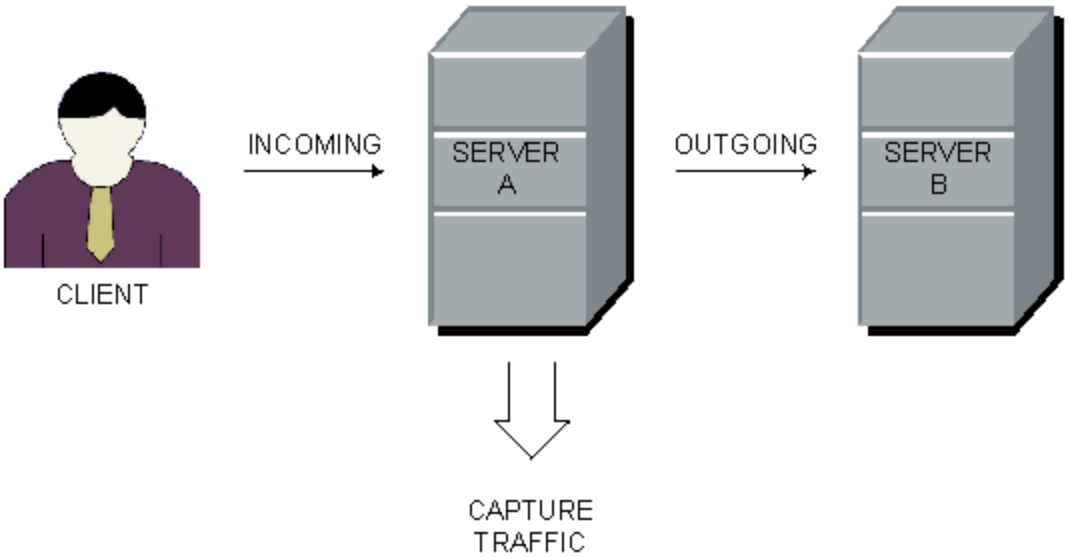
| Argument Type | Description |
|---|---|
| **Base64 Arguments** | An encoding method used to represent binary data as ASCII text. Since SOAP envelopes are plain text, you can use this encoding to represent binary data as text within SOAP envelopes.<br><br>When VuGen detects a WSDL element of **base64Binary** type, it lets you provide an encoded value. You can specify a value in two ways:<br><br>• **Get from file.** Reference a file name.<br>• **Embed encoded text.** Specify the text to encode.<br><br>For details, see "Process Base64 Data - Simple Data Dialog Box" on page 787. |

## Server Traffic Scripts Overview

The main focus when testing enterprises and complex systems, is to measure the performance from the client end. Ordinarily, VuGen records the actions you perform in the application or browser, and generates a script emulating the client actions and requests to the server.

In certain test environments, you may be unable to record the client application to retrieve the requests to the server. This may be a result of the server acting as a client, or because you do not have access to the client application. In these cases, you can create a script using VuGen's **Analyze Traffic** feature.

The **Analyze Traffic** feature examines a capture file containing the server network traffic, and creates a script that emulates requests sent to or from the server. The steps in creating a script by analyzing server traffic are described in "Create a Script by Analyzing Traffic (Web Services)" on page 775.




There are two types of emulations: **Incomingtraffic** and **Outgoing traffic**.

**Incoming traffic** scripts emulate situations in which you want to send requests to the server, but you do not have access to the client application, for example, due to security constraints. The most accurate solution in this case is to generate a script from the traffic going **into** the server, from the side of the client.

When you specify an Incoming server network traffic, you indicate the IP address of the server and the port number upon which the application is running. VuGen examines all of the traffic going into the server, extracts the relevant messages, and creates a script. In the above diagram, if the client is unavailable, you could create an Incoming script to emulate the requests coming into **Server A**.

**Outgoing Traffic** scripts emulate the server acting as a client for another server. In an application server that contains several internal servers, you may want to emulate communication between server machines, for example between **Server A** and **Server B** in the above diagram. The solution in this case is to generate a script from the traffic sent as output **from** a particular server.

When you create an Outgoing traffic script, you indicate the IP address of the server whose outgoing traffic you want to emulate, and VuGen extracts the traffic going out of that server. In the above diagram, an Outgoing script could emulate the requests that **Server A** submits to the **Server B**.

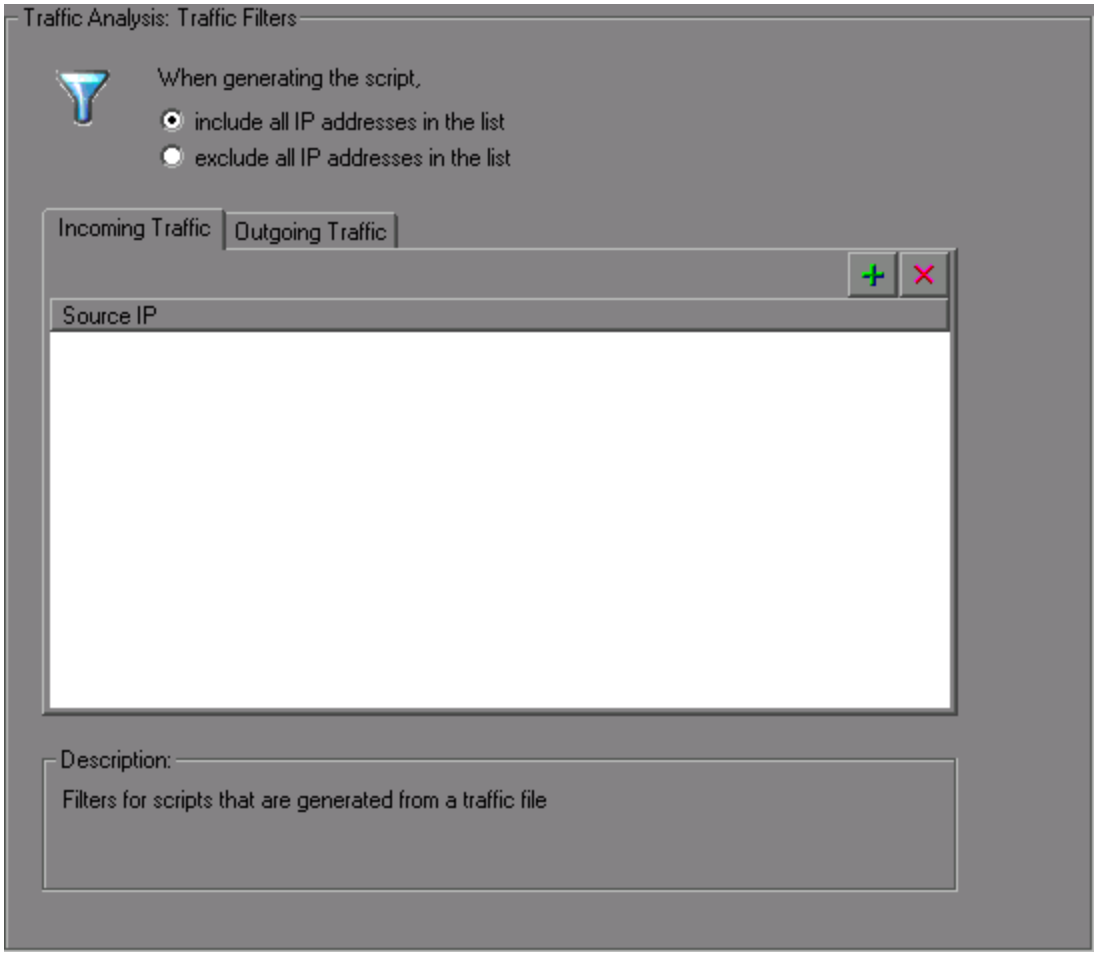
- "Create a PCAP File" on page 869
- "Filtering Traffic" below
- "Data on Secure Servers" on the next page

## Filtering Traffic

You can provide a filter to drill down on specific requests going to or from a server, by specifying its IP address and port.

> **Tip:** Several external capture tools allow you to filter the IP addresses while capturing the traffic.

You filter the requests by choosing the relevant host IP addresses. The filter can be inclusive or exclusive—you can include only those IPs in the list, or include everything except for those IPs that appear in the list.




For more information, see "Create a Script by Analyzing Traffic (Web Services)" on page 775.

## Data on Secure Servers

To analyze traffic from a secure server, you must provide a certificate containing the private key of the server.

If the traffic is SSL encrypted, you must supply a certificate file and password for decryption. If you want traffic from multiple servers to be reflected in the script, you must supply a separate certificate and password for each IP address that uses SSL.

For more information, see "Create a Script by Analyzing Traffic (Web Services)" on page 775.

## Add Content

This topic describes how to add content, such as Web Service calls, to a Web Services Vuser script.

### Prerequisites

Create an empty Web Services Vuser script. Click **File > New Script and Solution** and choose the **Web**

**Services** protocol. You can create either a single-protocol or multi-protocol Vuser script.

### Record a Session - Optional

1. Click the **Start Recording** button on the VuGen toolbar or press Ctrl+R to open the Recording Wizard > Specify Services screen.

   For user interface details, see "Specify Services Screen" on page 775.
2. Add services to the list. Click **Import** to load a WSDL for the test. Indicate the location of the WSDL file.

   For user interface details, see "Import Service Dialog Box" on page 822.
3. Click **Next**. Specify the location of the application and any other relevant arguments. See the "Specify Application to Record Dialog Box" on page 776.

### Add a New Service Call - Optional

1. Import a service. Click **Manage Services** to access the Import dialog box.

   For user interface details, see "Import Service Dialog Box" on page 822.
2. Click the cursor at the desired location in your script (**Editor**) or in the test steps (**Step Navigator**).
3. Click the **Add Service Call** button. The New Web Service Call dialog box opens.
4. In the Select Web Service Call section, select a **Service**, **Port Name**, and **Operation**.
5. To specify an endpoint other than the default **Target Address**, select **Override Address** and insert the new endpoint to which you want to submit the requests.
6. Expand the nodes and specify argument values. To create sample values for all Input arguments, select the **Input Arguments** node and click **Generate**.

   > **Tip:** To instruct VuGen to interpret a string as text, and not markup data, you can use a CDATA section. For example, suppose you want to specify an input string "<3558>&abc". Instead of using escape characters such as **<3558>&abc**, add a CDATA section in the following format: **<![CDATA[<3558>&abc]]>**.

   To edit, import, or export the element's XML structure, see "Assign Values to XML Elements" on the next page.
7. To parameterize an input argument, click the node and select the **Value** option. Click the ABC icon and proceed with parameterization. For more information, see "Parameterization" on page 357.
8. Select the **Transport Layer Configuration** node to specify advanced options, such as JMS transport for SOAP messages (Axis toolkit only), asynchronous messaging, or WS Addressing. For more information, see "Send Messages over JMS" on page 803.

### Add Attachments - Optional

1. To add an attachment to an input argument, choose an operation in the left pane. Select **Add to request (Input)**. VuGen prompts you to enter information about the attachment and adds it to the

method's tree structure. For details, see the "Add Input Attachment Dialog Box" on page 785.

2. To specify an output attachment through which to store output arguments, choose an operation in the left pane. Select **Save received (Output)**. Select the desired option: **Save All Attachments** or **Save Attachment by Index** based on their index number—beginning with 1. For details, see "Web Service Call Attachments" on page 764.
3. To edit the properties of either an Input or Output attachment, click the attachment in the left pane, and enter the required information in the right pane.

### Specify SOAP Headers - Optional

Select the **CustomSOAP Header** node in the left pane and enable the **Use SOAP header** option. You must individually specify SOAP headers for each element. To compose your own, click **Edit** and edit the XML. To import an XML file for the SOAP header, click **Import**.

### Import SOAP - Optional

1. Import a service if one is available. Click **Manage Services** to access the Import dialog box. For more information, see the "Import Service Dialog Box" on page 822.
2. Click the **Import SOAP** button to open the Import SOAP dialog box.
3. Browse for the XML file that represents your SOAP request.
4. Select the type of step you would like to generate: **Create Web Service Call** or **Create SOAP Request**. In order to create a Web Service Call, you must first import at least one WSDL that describes the operation in the SOAP request file. To view the SOAP before loading it, click **View SOAP**.
5. Click **Load** to import the XML element values.

   For a **Web Service Call**, set the properties for the Service call as described in the "New Web Service Call Dialog Box" on page 778.

   For a **SOAP Request**, provide the URL and the other relevant parameters.
6. For a Web Service Call, if there are multiple services with same operation (method) names, select the service whose SOAP traffic you want to import.
7. Click **OK** to generate the new step within your script.

### Analyze Server Traffic - Optional

To create a script by analyzing a file containing a dump of the server traffic, click **Analyze Traffic**.

For details, see "Server Traffic Scripts Overview" on page 769.

## Assign Values to XML Elements

This task describes how to work with XML elements by manually editing the code, importing an external file, and exporting the XML for later use.

1. Prerequisite

   Import a service and create a new Web Service call. Alternatively, in the Step Navigator, right-click a step and select **Show Arguments**.

2. Select the element.

   In the left pane, select a complex type or array argument. In the right pane, click **XML**. The XML field shows the XML code as a single string.

3. Import a file (optional).

   To import a previously saved XML file, click **Import** and specify the file's location.

4. Edit the XML elements (optional).

   To edit the XML structure and element values, click **Edit**. The XML Editor opens. To import a previously saved XML file, click **Import File.**

   - To manually edit the code, click the **Text View** tab.
   - To modify the XML through a graphical interface, click the **Step Navigator**. Use the shortcut menu to add children and sibling elements and rename the node. Click **Insert** from the shortcut menu to add a new element before or after the selected one.

   > **Tip:** To instruct VuGen to interpret a string as text, and not markup data, you can use a CDATA section. For example, suppose you want to specify an input string "<3558>&abc". Instead of using escape characters such as **<3558>&abc**, add a CDATA section in the following format: **<![CDATA[<3558>&abc]]>**.

5. Export a file (optional).

   To save your XML data to a file so it can be used for other tests, click **Export** and specify a location.

## Generate a Test Automatically

This task describes how to create requirements or tests for checking your service.

1. Open the wizard:

   Select **File > New** to open the New Virtual User dialog box. Select **SOA Test Generator** in the left pane and click **Create**.

2. Add a service:

   Proceed to the next screen and click **Add** to import at least one service. If your service is not ready yet, you can use an emulated service. For details, see "Add and Manage Services" on page 816. Click **Next**.

3. Select testing aspects:

   Expand the nodes and select the desired testing aspects. Click **Next**.

4. Specify a location:

   Specify a test name and a location for the test scripts: **ALM** or a **local file system**. If you specified ALM, click **Connect** to log on to the server and **Browse** to locate the test node.

5. Complete the test generation:

   Review the summary and include or exclude any scripts from the generation. Click **Generate**.

6. Open the scripts:

In the final screen, review the list of generated scripts and indicate which ones to open. Click **Finish**.

## Create a Script by Analyzing Traffic (Web Services)

This task describes how to create a Web Services Vuser script using a network traffic file. You can use a capture file of the following types: our **pcap**, **pcap**, **lrcap**, or **saz** (Fiddler)

1. Prerequisite: Create a capture file on a Windows platform. (Optional)

   Locate your capture file or create a new one using an external tool, such as Wireshark. For details on creating a capture file, see "Create a PCAP File" on page 869.
2. Open the Analyzing Traffic wizard.

   Create a new Web Services Vuser script and click **Analyze Traffic**. The "Specify Services Screen" opens.
3. Import a service and add it to the list. (Optional)
   a. Click **Import** to load a WSDL file. For details, see the "Import Service Dialog Box" on page 822.
   b. Click **Next**.
4. Specify traffic information.
   a. Browse for the capture file.
   b. Indicate whether you want to analyze **Incoming** or **Outgoing** traffic and specify the server whose traffic you want to analyze.
   c. Select the section of the script into which you want to load the traffic: **vuser_init**, **Action**, or **vuser_end**.
5. Filter the IP addresses. (Optional)

   Click the **Filter Options** button to open the Recording options and indicate which IP addresses to ignore or include. For details, see "Traffic Analysis > Traffic Filters Recording Options" on page 220.
6. Configure the SSL. (Optional)

   Click the **SSL Configuration** button to add SSL certificates. This is necessary in order to analyze traffic from a secure server. For details, see the "SSL Configuration Dialog Box" on page 791.

## Specify Services Screen

This dialog box enables specify the basic details needed to begin recording a Web Services script.

| | |
|---|---|
| **To access** | **VuGen > Start Record** button |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|

| | |
|---|---|
| Details... | Opens the Manage Services dialog box for providing further information about the service. For more information, see "Manage Services Dialog Box" on page 817. |
| Import | Opens the Import Service dialog box. For more information, see the "Import Service Dialog Box" on page 822. For user interface details, see the "Import Service Dialog Box" on page 822. |
| Delete | Removes the selected service from the list. |
| **<services list>** | A list of the available services:<br>• **Service Name.** The native name of the service.<br>• **WSDL Location.** The source of the WSDL. |

## Specify Application to Record Dialog Box

This dialog box enables specify the basic details needed to begin recording a Web Services script.

| | |
|---|---|
| **To access** | **VuGen > Start Record** button, **Next** |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Advanced Options... | Opens the Recording Options dialog box. For user interface details, see "Recording Options" on page 164. |
| **Record default Web browser** | Records the default browser actions. Specify the starting URL or click the Browse button to navigate to a location.<br><br>**Note:** The Web Services protocol only supports IE as the default browser. |
| **Record any application** | Records any Win32 application. You can also specify the following:<br>• **Program to record.** Select the browser, internet application, or Win32 application to record<br>• **Program arguments** (Win32 Applications only). Command line arguments for the application. For example, if you specify plus32.exe with the command line options peter@neptune, it connects the user Peter to the server Neptune when starting plus32.exe.<br>• **Working directory.** A working folder for the application (only when required by the application). |

| | |
|---|---|
| **Record into action** | The section into which you want to record: **vuser_init**, **Action**, or **vuser_end**. For actions you want to repeat, use the **Action** section. For initialization steps, use **vuser_init**. |
| **Record application startup** | In the following instances, it may not be advisable to record the startup:<br>• If you are recording multiple actions, in which case you need to perform the startup in only one action.<br>• In cases where you want to navigate to a specific point in the application before starting to record.<br>• If you are recording into an existing script. |

## Import SOAP Dialog Box

This dialog box enables you to create a test step based on a SOAP file.

| | |
|---|---|
| **To access** | Use one of the following:<br>• Click <br>• **SOA Tools > Import SOAP** |
| **Relevant tasks** | "Add Content" on page 771<br>"Adding Web Service Script Content - Overview" on page 762 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| ... | **Browse.** Locate the XML file containing SOAP traffic. |
| Load | Loads the element values from the SOAP file. |
| Manage Services | Opens the Manage Services dialog box for importing and configuring services. |
| View SOAP | Opens the SOAP file in a browser for viewing. |
| **<Call type>** | The type of call to generate in the script/test:<br>• **Web Service Call**. Requires the import of a service.<br>• **SOAP Request.** Generates a soap_request step in the script. |

| | |
|---|---|
| **SOAP Request Properties** | The properties of the SOAP request (only visible for **SOAP Request** type calls). Specify the following:<br>• **URL.** The URL or IP address of the server to which to submit the request.<br>• **SOAP Action.** The SOAP action to include in the request (applicable if there are multiple actions).<br>• **Response Parameter.** A parameter name to store the response of the SOAP or Web Service call request. |

## New Web Service Call Dialog Box

This dialog box lets you create and configure a new Web Service call.

| | |
|---|---|
| **To access** | Open a Web Service Vuser script and then click **SOA Tools > Add Service Call** or click the **Add Service Call** button on the VuGen toolbar. |
| **Important Information** | To access the Web Service call properties for existing Web Service calls, select a step in the **Step Navigator** and choose **Properties** from the shortcut menu. |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<service argument tree> (left pane)** | An expandable tree hierarchy of the Service containing the following nodes:<br>• <operation name><br>• Transport Layer Configuration<br>• Custom SOAP Header<br>• Input Arguments<br>• Output Arguments |
| **<parameter values> (right pane)** | Enables you to set and select values for each of the left pane's nodes. |

| | |
|---|---|
| **Select Web Service Call** | Lets you set the following items:<br>• **Service.** A drop-down list of all of the imported services, with the name derived from the WSDL.<br>• **Port Name.** A drop-down list of available ports through which to send the request.<br>• **Operation.** A drop-down list of the service's operations.<br>• **Target Address**. The default endpoint of the service.<br>• **Override Address.** Allows you to enter an alternate endpoint address in the **Target Address** box. |

### <Operation Name> Node

Allows you to generate sample values for the operation's input arguments and add attachments.

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Method** | The name of the selected operation (read-only). |
| **Description** | Free text area for entering a description of the service. |
| **Attachments** | Handles input and output attachments:<br>• **Add to Request (Input).** Attaches a file or parameter value to the request.<br>• **Save Received (Output).** Saves the response to a parameter. |
| **Step properties** | Lists the following service call properties and their values:<br>• WSDL file location<br>• Service<br>• Port name<br>• Target address<br>• SOAP action<br>• SOAP namespace |

### Transport Layer Configuration Node

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **HTTP/S Transport** | Sets the transport method to HTTP or HTTPS transport. |
| **Async Support** | Marks the Web Service call as an asynchronous message activated by an event:<br>**Async Event.** An arbitrary name for the event.<br>**Note:** Add a **Web Service Wait For Event** step to the script, to instruct the replay engine to wait for the event. |
| **WSA Support** | Enables WS-Addressing. Use one of the following options for a reply:<br>• **WS-A Reply.** An IP address of the server to reply to when the event occurs.<br>• **Autodetect.** Reply to the current host when the event occurs. This is useful when running the same script on several machines.<br>**Tip:** To use WS-Addressing calls in synchronous mode, leave the **Async Event** box empty. In Script view, remove the **AsyncEvent** argument. This instructs the replay to block script execution until the complete response is received from the server. |
| **JMS Transport** | Sets the transport method to JMS for *synchronous* messages. For details, see "Testing Web Service Transport Layers Overview" on page 792.<br>**Note:** For JMS *asynchronous* messages, manually add a **JMS Send Message Queue** or **JMS Receive Message Queue** step to the script, to set up the message queue information. |
| **Override JMS Queues** | Enables you to provide the request and response queues. |
| **Request Queue** | The queue name for the request message. |
| **Response Queue** | The queue name for the response message. |

### Custom SOAP Header Node

Lets you specify additional application-generated header elements to include in the SOAP envelope of an HTTP message. For task details, see "Add Content" on page 771.

User interface elements are described below:

| UI Element | Description |
|---|---|
| Edit... | Opens an XML editor that lets you view and edit the SOAP header XML code. |
| Import... | Opens the **Select XML File to Import** dialog box. |
| Export... | Opens the **Export SOAP Header into File** dialog box. |
| | Opens the **Select or Create New Parameter** dialog box. |
| **Use SOAP Header** | Includes a SOAP header in the HTTP request. |
| **Header** | The header source:<br>• For an imported file: Header element as it appears in the imported file.<br>• For a parameter: The parameter name (in curly brackets) |

### Input Arguments Node

Lets you set the properties and generate values for all input arguments.

User interface elements are described below:

| UI Element | Description |
|---|---|
| Include All | Includes all of the method's arguments in the Web Service call. |
| Reset | Resets the arguments to their original state. It removes their inclusion in the call, and sets them to the values in the WSDL. |
| Generate | Generates sample data for all of the input arguments. |
| Edit Argument | Opens the pane for editing the selected argument's value. |
| **Name** | The name of the operation (read-only) |
| **Argument List** | A list of the input arguments.<br>• Simple parameters<br>• Arrays (shows the top level only). |

### <Input Argument Name> Node

When selecting an input argument, the right pane allows you to specify argument values.

User interface elements are described below:

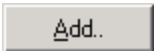
| UI Element | Description |
|---|---|
| Add.. | Opens the "Add Array Elements Dialog Box" on page 786 for adding a new array element to the input argument (only visible when selecting a parent node in an input array). |
| Delete... | Removes the selected array element in the input argument (only visible when selecting a parent node in an input array). |
| Include | Includes the sub-arguments of the selected argument, in the Web Service call. This is only enabled for an argument with sub-arguments with the **Include argument in call** option enabled. |
| Exclude... | Excludes the sub-arguments of the parent argument, from the Web Service call. |
| Edit... | Opens an XML editor for editing the XML code containing the argument values. The only changes saved are the element values and the number of array elements. |
| Import... | Opens the **Select XML File to Import** dialog box, |
| Export... | Opens the **Export argument XML into file** dialog box. |
| | Opens the **Select or Create Parameter** dialog box. |
| **Name** | The name of the argument or array. |
| **Include argument in call** | Include the argument in the call. For arrays, click **Include** to add the sub arguments to the call. To exclude all omittable arguments, click **Exclude**. |
| **Type** | The argument type as defined in the WSDL. When the WSDL contains derived types, this box becomes a drop-down list. For details, see "Special Argument Types" on page 765. |
| **Nil** | Sets the Nillable attribute to **true**. |

| | |
|---|---|
| **XML (for arrays only)** | • **XML.** Enables the **Edit**, **Import,** and **Export** buttons. By editing the XML, you can manually insert argument values. Click on the **ABC** icon to replace the entire XML structure with a single XML type parameter. **Note**: This import operation handles XML files that were previously exported—not standard SOAP files.<br>• **Generate auto-value for this argument.** Inserts automatic values for all children elements.<br>• **Add/Delete**. Adds or removes elements from the array. |
| **Value (for non -array elements)** | The argument value. To parameterize this value, click the abc icon (only available for non-arrays). |
| **Generate auto-value for this argument** | Generates a sample value for the selected argument. |

### Input Attachments Node

Lets you add attachment to the input arguments. This is only visible if you enabled **Add to request (Input)** in the operation name (top level parent) node.

User interface elements are described below:

| UI Element | Description |
|---|---|
|  | Open the Add Input Dialog box. In this dialog box, you specify the source, content-type, and content-ID of the attachment. |
| Attachment format | The format of the attachment, such as **MIME**, **DIME**, and so forth. |

### <Input Argument Attachment> Node

Lets you set the properties for input attachments. This is only visible if you enabled **Add to request (Input)** in the operation name (top level parent) node.

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Deletes the selected attachment parameter. If you have saved the attachments by index, it only removes the selected item. |
| **Take data from** | The source of the attachment—either a file, or an existing parameter. |

| | |
|---|---|
| **Content-type** | The content type of the parameter: **Detect automatically** or specify a value such as **application/octet-stream**. |
| **Content-ID** | A unique content ID for the parameter: **Generate automatically** or specify a value. |

### Output Arguments Node

Lets you see the properties of all output arguments.

User interface elements are described below:

| UI Element | Description |
|---|---|
| Edit Argument | Opens the pane for editing the selected argument's value. |
| **Name** | The name of the operation (read-only). |
| **Argument List** | A list of the output arguments and the corresponding parameters storing the values. |

### <Output Argument Name> Node

Lets you specify a parameter for storing the value of the output argument.

User interface elements are described below:

| UI Element | Description |
|---|---|
| Add.. | Opens the "Add Array Elements Dialog Box" on page 786 for adding a new array element to the output argument (only visible when selecting a parent node in an output array). For details, see the "Add Array Elements Dialog Box" on page 786. |
| Delete | Removes the selected array element in the output argument (only visible when selecting a parent node in the output array). |
| **Name** | The name of the output argument or array. |
| **Save returned value in parameter** | Saves the value of the selected argument to a parameter. To specify a custom parameter name, modify the default Param_<*arg_name*> in the **Parameter** field. |
| **Nil** | Sets the value of the current argument to **nil=true.** |
| **XML (for arrays only)** | XML code containing the argument values. To parameterize this value, click the abc icon (only available for arrays). |

### Output Attachments Node

Lets you set the properties for output attachment parameters. This is only visible if you enabled **Save received (Output)** attachments in the operation name (top level parent) node.

User interface elements are described below:

| UI Element | Description |
|---|---|
| Add.. | Adds a new index-based output argument. This is available only when choosing **Save Attachments by Index**. |
| **Save All Attachments** | Saves all output attachments to a parameter with the following properties:<br>• **Content.** An editable name for the parameter storing the attachment<br>• **Content-type.** The type of parameter (read-only).<br>• **Content-ID.** A unique ID for the parameter (read-only). |
| **Save Attachments by Index** | Saves the output attachments to index-based parameters. To set the index, select one of the parameters and modify the index number in the right pane. |

### <Output Argument Attachment> Node

Lets you set the properties for output attachments. This is only visible if you enabled **Save received (Output)** attachments in the operation name (top level parent) node.

User interface elements are described below:

| UI Element | Description |
|---|---|
| Delete Attachment | Deletes the selected attachment parameter. If you have saved the attachments by index, it only removes the selected item. |
| **Index** | An index number for the parameter. This field is only enabled when you select **Save Attachments by Index** in the Output Attachments node. |
| **Content** | An editable name for the parameter storing the attachment |
| **Content-type** | The content type of parameter (read-only). |
| **Content-ID** | A unique content ID for the parameter (read-only). |

## Add Input Attachment Dialog Box

This page enables you to add input attachments to your Web requests.

| | |
|---|---|
| **To access** | Click Add Service Call and select the top node—the Operation name. Select **Add to Request (input)** in the **Attachments** section. |
| **Important information** | You must import a service before adding an attachment to a service call.<br>For background information, see "Web Service Call Attachments" on page 764. |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Take data from** | The location of the data.<br>• **File.** The file location:<br>  • Absolute Path: The full path of the file. Note that this file must be accessible from all machines running the script.<br>  • Relative Path: (recommended) A file name. Using this method, during replay, VuGen searches for the attachment file in the script's folder. To add it to the script's folder, select **File > Add Files to Script** and specify the file name.<br>• **Parameter.** The name of a parameter containing the data. |
| **Content-type** | The content type of the file containing the data. The **Detect Automatically** option instructs VuGen to automatically determine the content type. The **Value** box accepts manual entries and provides a drop-down list of common content types. |
| **Content-id** | A unique identifier for the attachment. By default, VuGen generates this automatically. Optionally, you can specify another ID in the **Value** box. |

## Add Array Elements Dialog Box

This page enables you to add elements to an argument array with an identical structure to the existing array. This is available for both Input and Output arguments.

For Input elements, you can base the new array's values on an existing element.

| | |
|---|---|
| **To access** | Click Add Service Call. Select an argument node that is an array. |
| **Important information** | You must have an array in your argument tree in order to view this dialog box. |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Name** | The name and index of the array's parent node. |
| **Start Index** | The index from which to add new array elements. |
| **Elements** | The number of identical array elements to add to your argument tree. |
| **Copy values from index** | Creates the new array elements with the values of a specific array element (only available for Input arguments). |

## Process Base64 Data - Simple Data Dialog Box

This dialog box enables you to set the encoding options for your simple base64 data.

| | |
|---|---|
| **To access** | For simple, non-complex Base64 values:<br>• Select an input argument in the Web Service Call Properties of **Base64** type.<br>• Select the **Value** option.<br>• Choose **Embed encoded text**.<br>• Click the Browse button. |
| **Important information** | For a complex array, use the "Process Base64 Data - Complex Data Dialog Box" below. |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Decode to File ... | Allows you to save the decoded text to a file. |
| **Text to encode** | For complex data, use the "Process Base64 Data - Complex Data Dialog Box" below. |
| **Encoding Options** | A list of encoding methods.<br>**Default:** Unicode (UTF-8) |
| **Encoded data** | The encoded version of the data from the **Text to encode** pane. |

## Process Base64 Data - Complex Data Dialog Box

This dialog box enables you to set the encoding options for your complex base64 data.

| | |
|---|---|
| **To access** | For complex Base64 values:<br>• Select a complex input argument in the Web Service Call Properties, of **Base64** type.<br>• Select the **Value** option click the Parameter icon.<br>• Replace the value with a parameter.<br>• Right-click the Parameter icon in the Value box and select **Parameter Properties**.<br>• Click the **Edit Data** button.<br>• In the desired values set column, click the B64 button. |
| **Important information** | For simple, non-complex data, use the "Process Base64 Data - Simple Data Dialog Box" on the previous page. |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Encode | Encodes the specified file. |
| Decode to File ... | Enables you to save decoded data to a file. This is usually data obtained during replay. |
| **File** | Encode the file by reference or its contents.<br>• **File path.** The file to encode.<br>• **Link to file.** References the file containing the values If cleared, it uses the content of the specified file. It copies the content to the script folder.<br>**Tip:** For text exceeding 10KB, enable **Link to file**. |
| **Text** | Encodes the specified text string.<br>• **Text to encode.** The Base64 text to encode. As you type the text, VuGen encodes it in the **Encoded data** pane.<br>• **Encoding Options.** A list of encoding methods. The default is Unicode (UTF-8). |

### Aspects List

The following table lists the available testing aspects:

| Aspect Name | Description |
|---|---|

| | |
|---|---|
| **Positive Testing** | Generates a full positive test that checks each operation of the service. |
| **Standard Compliance** | Checks the service's compliancy with industry standards such as WS-I and SOAP. |
| **Service Interoperability** | Tests the interoperability of the service's operations with all supported Web Services toolkits.<br>Contains the following sub-aspects:<br>• **.NET Framework.** Tests that the services are fully interoperable with .NET Framework WSE 2 Toolkit by calling all of its operations with default/expected values.<br>• **Axis/Java Based Web Services.** Tests that the services are fully interoperable with Axis 1.3 Web Services Framework by calling all of its operations with default/expected values. |
| **Security Testing** | Tests service security. Contains the following sub-aspects:<br>• **SQL Injection Vulnerability.** Checks if the service is vulnerable to SQL injections by injecting SQL statements and errors into relevant parameters.<br>• **Cross-site Scripting (XSS).** Attempts to hack the service by injecting code into a Web site that will disrupt its functionality. |
| **Boundary Testing** | Using the negative testing technique, creates tests to manipulate data, types, parameters, and the actual SOAP message to test the service to its limits.<br>Contains the following sub-aspects:<br>• **Extreme Values.** Provides invalid data types to the services and verifies they are not accepted.<br>• **Null Values.** Provides NULL parameters to the services to verify they are not accepted. |
| **Performance Testing** | Contains the following sub-aspects:<br>• **Stress Testing.** Tests the maximum load that can be placed on the application.<br>• **Overload Sustainability Testing.** Tests how well the hardware allocated for the application can support the number of anticipated users.<br>• **Volume Testing.** Tests that the system can handle a massive data entry.<br>• **Longevity Test.** Tests that the system can sustain a consistent number of concurrent Vusers executing transactions using near-peak capacity, over a minimum 24-hour period.<br>• **Scalability Testing.** Repeated stress, overload, volume, and longevity tests with different server or network hardware configurations. |

## Specify Services Screen

This wizard screen enables you to select Web services to associate with your traffic-based script.

| | |
|---|---|
| **To access** | **Analyze Traffic** button |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| Details... | Opens the **Manage Services** dialog box for providing further information about the service. For more information, see "Manage Services Dialog Box" on page 817. |
| Import | Opens the **Import Service** dialog box. For more information, see "Import Service Dialog Box" on page 822. |
| Delete | Removes the selected service from the list. |
| **<services list>** | A list of the available services:<br>• **Service Name.** The native name of the service.<br>• **WSDL Location.** The source of the WSDL. |

## Specify Traffic Information Screen

This wizard screen enables you to specify the capture file for the incoming or outgoing traffic.

| | |
|---|---|
| **To access** | **Analyze Traffic** button, **Next** |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| ... | **Browse.** Allows you to select a capture file to import. |
| Filter Options... | Opens the **Traffic Filters** node in the Recording Options dialog box. This allows you to specify which IP addresses to include or exclude from the script. For details, see "Recording Options" on page 164. |
| SSL Configuration... | Opens the "SSL Configuration Dialog Box" on the next page which allows you to add SSL certificates to analyze traffic from a secure server. |
| **Capture file** | The name of a capture file containing the server traffic, usually with a **cap** extension. |
| **Incoming Traffic** | The IP address and port of the server whose incoming traffic you want to examine. |

| | |
|---|---|
| **Outgoing Traffic** | The IP address of the server whose outgoing traffic you want to examine. |
| **Record into action** | The section into which you want to create the script: **vuser_init**, **Action**, or **vuser_end**. For actions you want to repeat, use the **Action** section. For initialization steps, use **vuser_init**. |

## SSL Configuration Dialog Box

This dialog box enables you to configure the SSL information for decrypting your traffic file.

| | |
|---|---|
| **To access** | **Analyze Traffic** button, **Next**, SSL Configuration... |
| **Relevant tasks** | "Add Content" on page 771 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | **Add Certificate.** Adds a new entry to the certificate list, containing information about an SSL server and its private key. |
| | **Delete**. Removes the selected private key. |
| **<certificate list>** | The properties of the SSL entry. Specify the following:<br>• **IP.** IP address of the server being analyzed.<br>• **Port.** Port of the server being analyzed.<br>• **File.** The path of the certificate file (with a **pem** extension) containing the private key. Use the **Browse** button to locate the file.<br>• **Password.** A password for decrypting the private key.<br>See **Certificate Utility** below for details on how to convert certificates. |
| | Opens the **Convert Certificate** tab of the **SSL Utility** that enables you to convert certificates from PKCS #12 and X.509 formats to PEM format.<br>For details on how to convert the certificate to PEM format, see the "SSL Utility" on page 1031, or see **web_set_certificate_ex** in the Function Reference (**Help > Function Reference**). |

# Web Services - Preparing Scripts for Replay

## Preparing for Replay Overview

After you create a script with Web Service calls, you prepare it for replay.

You can enhance it with custom error and log messages or with transactions. In addition, you can enhance your script with JMS functions, **jms_<suffix>** or XML functions, **lr_xml_<suffix>**. For more information, see the Function Reference (**Help > Function Reference**).

Runtime settings let you emulate real users more accurately, configure the runtime settings. These settings include general settings and Web Service specific settings. For details, see "Runtime Settings" on page 301.

In certain cases, you may need to use the result of one Web Service call as input for another. To do this, you save the result to a parameter and reference it later. For more information, see "Prepare Scripts for Replay" on page 801.

## Testing Web Service Transport Layers Overview

Web services can be sent over various transport layers. The transport layer is the protocol used to transport messages to and from the server.

VuGen allows you to configure the transport layer for your services. It fully supports HTTP/HTTPS and JMS (Java Message Service) transport layers.

With user handlers, you can process SOAP requests and responses and assign them a custom behavior. For more information, see " User Handlers" on page 797.

- "Sending Messages over HTTP/HTTPS" below

### Sending Messages over HTTP/HTTPS

HTTP is used for sending requests from a Web client, usually a browser, to a Web server. HTTP is also used to return the Web content from the server back to the client.

HTTPS handles secure communication between a client and server. Typically, it handles credit card transactions and other sensitive data.

The typical request and response mechanism is synchronous. In synchronous messaging, the replay engine blocks script execution until the server sends its response. In asynchronous mode, the replay engine executes the script without waiting for server's response for previous messages.

If you are working with HTTP or HTTPS transport, you can use asynchronous calls in conjunction with WS-Addressing. For details, see "WS-Addressing" on page 795.

## JMS Transport Overview

JMS is a J2EE standard for sending messages, either text or Java objects, between Java clients.

Communication scenarios:

- **Peer-to-Peer.** Also known as **Point-to-Point**. JMS implements point-to-point messaging by defining a message queue as the target for a message. Multiple senders send messages to a message queue, and the receiver gets the message from the queue.
- **Publish-Subscribe.** Each message is sent from one publisher to many subscribers through a designated topic. The subscribers only receive messages sent after they have subscribed.

VuGen supports point-to-point communication by allowing you to send and receive JMS messages to and from a queue.

Before you can send messages over JMS transport, you need to configure several items that describe the transport:

- **JNDI initial context factory.** The class name of the factory class that creates an initial context which will be used to locate the JMS resources such as JMS connection factory or JMS queue.
- **JNDI provider.** The URL of the service provider which will be used to locate the JMS resources such as JMS connection factory or JMS queue.
- **JMS connection factory.** The JNDI name of the JMS connection factory.

In addition, you can set a timeout for received messages and the number of JMS connections per process.

You configure these settings through the JMS runtime settings. For details, see the **JMS > Advanced** view.

### See also:

- "JMS Script Functions" below
- "JMS Message Structure" on the next page

#### JMS Script Functions

VuGen uses its API functions to implement the JMS transport. Each function begins with a **jms** prefix:

| Function Name | Description |
|---|---|
| **jms_publish_ message_topic** | Publishes messages to a specific topic |
| **jms_receive_ message_queue** | Receives a message from a queue |
| **jms_receive_ message_topic** | Receives published messages to a specific topic on a subscription. |
| **jms_send_message_ queue** | Sends a message to a queue. |

| | |
|---|---|
| **jms_send_receive_message_queue** | Sends a message to a specified queue and receives a message from a specified queue. |
| **jms_subscribe_topic** | Creates a subscription for a topic. |
| **jms_set_general_property** | Sets a general property in the user context. |
| **jms_set_message_property** | Sets a JMS header or property for the next message to be sent, or uses a JMS header or property to filter received messages. |

The JMS steps/functions are only available when manually creating scripts—you cannot record JMS messages sent between the client and server.

Unlike peer-to-peer communication that uses message queues, the publish-subscribe functions, **jms_publish_message_topic**, **jms_subscribe_topic**, and **jms_receive_message_topic**, are not supported for Web Service calls. To use these functions with Web Service calls, you must manually set up user handlers to generate the JMS message payload. For more information, see "Create a User Handler" on page 807.

For details about the JMS functions, see the Function Reference (**Help > Function Reference**) (or click **F1** on the function).

### JMS Message Structure

Each JMS message is composed of:

- **Header.** contains standard attributes (Correlation ID, Priority, Expiration date).
- **Properties.** custom attributes.
- **Body.** text or binary information.

JMS can be sent with several message body formats. Two common formats are **TextMessage** and **BytesMessage**.

To override the default behavior, use a **jms_set_general_property** function before sending the message. Set the JMS_MESSAGE_TYPE property to TextMessage, BytesMessage, or Default. For Example:

jms_set_general_property("step1","**JMS_MESSAGE_TYPE**","BytesMessage");

For more information, see the Function Reference (**Help > Function Reference**).

## Asynchronous Messages Overview

You can use VuGen to emulate both synchronous and asynchronous messaging.

In synchronous messaging, the replay engine blocks script execution until the server sends its response. In asynchronous mode, the replay engine executes the script without waiting for server's response for previous messages.

This section also includes:

- "Sending Asynchronous Calls with HTTP/HTTPS" below
- "WS-Addressing" below

## Sending Asynchronous Calls with HTTP/HTTPS

This following section describes how to use asynchronous calls in HTTP/HTTPS. You use a **Wait for Event** step to instruct Vusers to wait for the response of previous asynchronous requests before continuing. The listener blocks the execution of the service until the server responds.

When adding a Web Service Wait for Event step, you specify the following:

- **Quantifier.** The quantifier indicates whether the Vuser should wait for **ALL** events to receive a response or **ANY,** just one of them. **ANY** returns the name of the first event to receive a response. **ALL** returns one of the event names.
- **Timeout.** the timeout in milliseconds. If no events receive responses in the specified timeout, then **web_service_wait_for_event** returns a NULL.
- **Events.** a list all of the asynchronous events for which you want to wait.

When running a script with asynchronous messaging, the Replay log provides information about the events and the input and output arguments.

For task details, see "Send Messages over HTTP/S" on page 804.

When setting up an asynchronous message, you can set the location to which the service responds when it detects an event using WS-Addressing. For more information, see "WS-Addressing" below.

## WS-Addressing

WS-Addressing is a specification that allows Web Services to communicate addressing information. It does this by identifying Web service endpoints in order to secure end-to-end endpoint identification in messages. This allows you to transmit messages through networks that have additional processing nodes such as endpoint managers, firewalls, and gateways. WS-Addressing supports Web Services messages traveling over both synchronous or asynchronous transports.

The WS-Addressing specification requires a **WSAReplyTo** address—the location to which you want the service to reply.

An optional **WSAAction** argument allows you to define a SOAP action for instances where transport layers fails to send a message.

The following example illustrates a typical SOAP message using WS-Addressing, implemented in the background by VuGen.

```
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
                xmlns:wsa="http://www.w3.org/2004/12/addressing">
    <S:Header>
        <wsa:MessageID>
            http://example.com/SomeUniqueMessageIdString
```

```
        </wsa:MessageID>
        <wsa:ReplyTo>
            <wsa:Address>http://myClient.example/someClientUser</wsa:Address>
        </wsa:ReplyTo>
<wsa:Address>http://myserver.example/DemoErrorHandler</wsa:Address>
        </wsa:FaultTo>
        <wsa:To>http://myserver.example/DemoServiceURI</wsa:To>
        <wsa:Action>http://myserver.example/DoAction</wsa:Action>
    </S:Header>
    <S:Body>
         <!-- Body of SOAP request message -->
    </S:Body>
</S:Envelope>
```

In the following example, the server responds to the interface 212.199.95.138 when it detects Event_1.

```
web_service_call("StepName=Add_101",
        "SOAPMethod=Calc.CalcSoap.Add",
        "ResponseParam=response",
        "AsyncEvent=Event_1",          "WSAReplyTo=212.199.95.138",
        "WSDL=http://lab1/WebServices/CalcWS/Calc.asmx?wsdl",
        "UseWSDLCopy=1",
        "Snapshot=t1153825715.inf",
        BEGIN_ARGUMENTS,
            "first=1",
            "second=2",
        END_ARGUMENTS,
        BEGIN_RESULT,
            "AddResult=Param_AddResult1",
        END_RESULT,
        LAST);
```

You can issue WS-Addressing calls in both asynchronous and synchronous modes. To use WS-Addressing in synchronous mode, leave the **Async Event** box empty in the Transport Layer options. In Script view, remove the **AsyncEvent** argument. This instructs the replay engine to block script execution until the complete response is received from the server.

For task details, see "Send Messages over HTTP/S" on page 804.

## Customizing Overview

VuGen provides several advanced capabilities that allow you to customize the way your script behaves. These capabilities are user handlers and configuration files.

With user handlers, you can process SOAP requests and responses and assign them a custom behavior. For more information, see below.

Configuration files let you customize advanced settings such as security information and the WSE configuration.

- " User Handlers" below
- " Custom Configuration Files" on page 799

## User Handlers

User Handlers are open APIs through which you can perform the following operations:

- Get and set the request/response SOAP envelopes
- Override the transport layer
- Get and set the request/response content type
- Get and Set values for LoadRunner parameters
- Retrieve a configuration argument from the script
- Issue messages to the execution log
- Fail an execution

You can set up a user handler directly in a script, or implement it through a DLL. You can apply the handler locally or globally.

For task details, see "Create a User Handler" on page 807.

For sample user handlers, see "User Handler Examples" on page 799.

### Handler Function Definitions

For basic implementation of a user handler, you define a user handler function within your Vuser script with the following syntax:

```
int MyScriptFunction(const char* pArgs, int isRequest)
```

The **pArgs** argument contains the string that is specified in **UserHandlerArgs** argument of **web_service_call** function.

The **isRequest** argument indicates whether the function is being called during processing of a Request (1) or Response (0) SOAP envelope.

The content of SOAP envelope is passed to a parameter called **SoapEnvelopeParam** for both requests and responses. After the function processes the SOAP envelope, make sure to store it in the same parameter.

To call the handler function, use the function name as a value for the **UserHandlerFunction** argument in the relevant Web Service Call step. For more information, see the Function Reference (**Help > Function Reference**).

### Event Handler Return Codes

VuGen recognizes the following return codes for the handler function.

| Return Code | | Description |
|---|---|---|

| | | |
|---|---|---|
| LR_HANDLER_SUCCEEDED | 0 | The Handler succeeded, but the SOAP envelope did not change. |
| LR_HANDLER_FAILED | 1 | The Handler failed and further processing should be stopped. |
| LR_HANDLER_SUCCEEDED_ AND_MODIFIED | 2 | The Handler succeeded and the updated SOAP envelope is stored in **SoapEnvelopeParam**. |

In the following example, a script handler manipulates the outgoing envelope:

```
//This function processes the SOAP envelope before sending it to the server.
int MyScriptFunction(const char* pArgs, int isRequest)
{
    if (isRequest == 1) {
        //Get the request that is going to be sent
        char* str = lr_eval_string("{SoapEnvelopeParam}");
        //Manipulate the string...
        //Assign the new request content
        lr_save_string(str, "SoapEnvelopeParam");
        return LR_HANDLER_SUCCEEDED_AND_MODIFIED;
    }
    return LR_HANDLER_SUCCEEDED;
}
Action()
{
    //Instruct the web_service_call to use the handler
    web_service_call( "StepName=EchoAddr_102",
        "SOAPMethod=SpecialCases.SpecialCasesSoap.EchoAddr",
        "ResponseParam=response",
        "userHandlerFunction=MyScriptFunction",
        "Service=SpecialCases",
        "Snapshot=t1174304648.inf",
        BEGIN_ARGUMENTS,
        "xml:addr="
            "<addr>"
                "<name>abcde</name>"
                "<street>abcde</street>"
                "<city>abcde</city>"
                "<state>abcde</state>"
                "<zip>abcde</zip>"
            "</addr>",
        END_ARGUMENTS,
        BEGIN_RESULT,
        END_RESULT,
        LAST);
    return 0;
```

### Custom Configuration Files

Configuration files let you customize advanced settings such as security information and the WSE configuration. These files let you control the behavior of the test during runtime.

The standard .NET configuration file, **mmdrv.exe.config**, is located in the VuGen installation folder. Some applications have their own configuration file, **app.config**.

You can customize the test run further, by filtering out the input or output. In addition, you can configure security information. such as token information and whether or not to allow unsigned test certificates.

For task details, see "Customize Configuration Files" on page 810.

## User Handler Examples

This section illustrates several common uses for user handlers.

### .NET Filters

You can apply a .NET filter to your messages using the user handler mechanism.

If you are familiar with Microsoft's Web Service Enhancements (WSE) 2.0, you can create a .NET filter and register it for incoming or outgoing SOAP messages. A .NET filter is a class that is derived from Microsoft.Web.Services2.SoapInputFilter or Microsoft.Web.Services2.SoapOutputFilter. By overriding the **ProcessMessage** function of this class, you can examine and modify the envelope's body and header.

To define the filter globally for the entire script, add the following lines to the script's default.cfg file below.

```
[UserHandler]
Function=LrWsSoapFilterLoader
Args=<Filters InputFilterClass="class name" InputFilterLib="lib name"
OutputFilterClass="class name" OutputFilterLib="lib name" />
Order=BeforeSecurity/AfterSecurity/AfterAttachments
```

The **InputFilterClass** parameter indicates the name of your class, and **InputFilterLib** indicates the name of the assembly in which the class resides. For example:

```
    web_service_call(
        ...
        "UserHandlerName=LrWsSoapFilterLoader",
        "UserHandlerArgs=<Filters
InputFilterClass=\"MyFilterNamespace.MyFilterClassName\"
InputFilterLib=\"MyAssemblyName\" />",
        BEGIN_ARGUMENTS,
        ...
        END_ARGUMENTS,
```

```
        ...
        );
```

Use SoapOutputFilter to examine an outgoing **web_service_call** request, and SoapInputFilter to examine the response from the server. Use **InputFilterClass** and **InputFilterLib** if your filter is derived from SoapInputFilter, or **OutputFilterClass** and **OutputFilterLib** if your filter is derived from SoapOutputFilter.

To define the filter for a specific step, add the following arguments to the **web_service_call** function.

```
UserHandlerName= LrWsSoapFilterLoader
UserHandlerArgs=<Filters InputFilterClass=\"class name\" InputFilterLib=\"lib name\"
OutputFilterClass=\"class name\" OutputFilterLib=\"lib name\" />
UserHandlerOrder=BeforeSecurity/AfterSecurity/AfterAttachments
```

### Overriding the Transport Layer

The following example shows a user handler function overriding the transport layer. VuGen does not automatically send the SOAP request over HTTP transport—instead it follows the transport method indicated in the custom handler.

After you receive a response, set the response envelope with the command:

```
lr_save_string(someResponseEnvelopeStr, "SoapEnvelopeParam");
```

To apply an alternate transport layer, specify **ReplaceTransport** as a value for the **UserHandlerOrder** argument. Define the transport layer in the handler.

```
web_service_call(
...
"UserHandlerFunction=<Transport HandlerFunction>",
"UserHandlerArgs=<handler arguments>",
"UserHandlerOrder=ReplaceTransport"
...

LAST);
```

### Including MIME Attachments

When working with Web Service scripts based on the .NET toolkit, the infrastructure does not support MIME attachments. Using the handlers mechanism, you can add MIME attachment functionality to .NET scripts.

The following sections describe how to send and receive MIME attachments for the .NET toolkit. You can receive and send a MIME attachment in the same operation.

### Sending MIME Attachments

To send a MIME attachment, add the boldfaced code to the **web_service_call**:

```
web_service_call( "StepName=EchoComplex_101",
  "SOAPMethod=SimpleService|SimpleServiceSoap|EchoComplex",
  "ResponseParam=response",
  "Service=SimpleService",
 "UserHandlerName=LrWsAttachmentsHandler", "UserHandlerArgs=ATTACHMENT_ADD;
ATTACHMENTS_FORMAT_MIME;  ContentType=text/plain; FileName=C:\\temp\\results.discomap",
  "ExpectedResponse=SoapResult",
  "Snapshot=t1208947811.inf",
  BEGIN_ARGUMENTS,
  "xml:cls="
   "<cls>"
    "<i>123456789</i>"
    "<s>abcde</s>"
   "</cls>",
  END_ARGUMENTS,
  BEGIN_RESULT,
  END_RESULT,
   LAST);
```

Modify the **FileName** and **ContentType** parameters to indicate the actual path and content type.

### Receiving MIME Attachments

To receive a MIME attachment, add the following code to the **web_service_call**:

```
"UserHandlerName=LrWsAttachmentsHandler",
 "UserHandlerArgs=ATTACHMENT_SAVE_ALL;ParamNamePrefix=attach;"
```

### Sending and Receiving MIME Attachments

To send and receive a MIME attachment in the same **web_service_call**, modify the Web Service call as shown below:

```
"UserHandlerName=LrWsAttachmentsHandler",
 "UserHandlerArgs=ATTACHMENT_SAVE_ALL;ParamNamePrefix=attach; ATTACHMENT_ADD;
ATTACHMENTS_FORMAT_MIME; ContentType=text/plain;
FileName=C:\\temp\\results.discomap",
```

## Prepare Scripts for Replay

This task describes how to prepare the script for replay and run it. It describes how to use the output of one Web Service call as input for another.

### Assign Input Parameter Values

First save the output result to a parameter, and then reference that parameter in a later Web Service call.

1. **Save the output parameter**.
   a. In the **Step Navigator**, double-click the Web Service call whose output you want to use, to view its properties.
   b. In the left pane, select the output argument whose value you want to save to a parameter.
   c. In the right pane, select **Save returned value in parameter**. Specify a name in the **Parameter** box.
2. **Use the saved parameter for input.**
   a. In the **Step Navigator**, double-click the Web Service call whose input parameters you want to set.
   b. In the left pane, select the input argument for which to use the saved parameter.
   c. In the right pane, select **Value**, and click on the abc icon. The Select or Create Parameter box opens.
   d. Select the saved output parameter from the **Parameter name** list.
   e. To specify an input parameter in Script view, select the value you want to replace and select **Use Existing Parameters** from the shortcut menu. Select one of the available parameters.

### Set the Runtime Settings - Optional

Open the runtime settings (F4) to configure JMS and VM settings. Click the **JMS > Advanced** node. For details, see the **JMS > Advanced** view in the runtime settings.

### Configure XSDs With any type Elements - Optional

For Web Services that have an XSD schema with an **Any** type element, `<xsd:element name="<Any_element>" type="xsd:anyType" />`, check that the script conforms with the following model:

```
BEGIN_ARGUMENTS,
               "xml:Any_element="
               "<Any_element>"
               "<string>the string to send</string>"
               "</Any_element>",
END_ARGUMENTS,
```

The actual SOAP may differ slightly, but as long as your script conforms to the above model, it will run properly.

You can also send complex type elements for the **<any>** type. For example:

```
"xml:Any_element="
         "<Any_element>"
         "<myComplexTypeName>"
         "<property1>123</property1>"
         "<property2>456</property2>"
```

```
        "</myComplexTypeName>"
        "</Any_element>",
```

### Run the Script

Click **Replay > Run**. Observe the output log for relevant messages.

### Review the Test Results

Display the Replay Summary to review results of the test run. For details, see "Replay Summary Pane" on page 131.

## Send Messages over JMS

This task describes how to send messages using the JMS transport method.

1. **Open the step properties**

   In the **Step Navigator**, select the step whose transport you want to set, and then select **Show Arguments** from the shortcut menu.

2. **Select the JMS transport method**

   Select the **Transport Layer Configuration** node and choose **JMS Transport**.

   For UI details, see "New Web Service Call Dialog Box" on page 778.

3. **Set the runtime settings - optional**

   Configure the runtime settings as described in the **JMS > Advanced** view.

4. **Send synchronous JMS messages - optional**

   Once you create a Web Service call and designate the transport method as JMS, VuGen sends the JMS messages in a synchronous manner. If desired, specify the queue information.

5. **Send asynchronous JMS messages - optional**

   To implement asynchronous messages over JMS, you send the request or retrieve the response using JMS steps—not Web Service calls.

   a. Click within the script at the desired location. Select **Insert > New Step** from the right-click menu, and locate the **JMS Functions** in the Steps Toolbox.

   b. Select a JMS function: **JMS Send Message Queue** sends a message to a queue. **JMS Receive Message Queue** receives a message from the queue.

   c. Click **OK** to open the JMS function properties.

   d. Specify a queue name and click **OK** to generate the JMS functions.

   For additional information about these functions, see the Function Reference (**Help > Function Reference**) (or click **F1** on the function).

6. **Send messages over JMS using SOAP messages - optional**

To send messages over JMS, using the SOAP message and without a Web Service call:

a. Record SOAP messages using a standard Web protocol.

b. Click within the script at the desired location. Select **Insert > New Step** from the right-click menu, and locate the **JMS Functions** in the Steps Toolbox.

c. Select a JMS function: **Send Message Queue** or **JMS Receive Message Queue**.

d. Click **OK** to open the JMS function properties.

e. Specify a queue name and click **OK** to generate the JMS functions.

For details, see the Function Reference (**Help > Function Reference**) (or click **F1** on the function).

## Send Messages over HTTP/S

This task describes how to send messages using the HTTP transport method.

1. Open the step properties.

   In the **Step Navigator**, select the step whose transport you want to set, and then select **Show Arguments** from the shortcut menu.

2. Select the HTTP/S transport method.

   Select the **Transport Layer Configuration** node and choose **HTTP/S Transport**.

3. Send a HTTP synchronous message. (Optional)

   To send messages in synchronous mode over HTTP, create a standard Web Service call, and do not enable the **Async Support** option.

4. Send asynchronous HTTP messages. (Optional)

   a. Choose **HTTP/S Transport** and select the **Async Support** option.

   b. Type an event name in the **Async Event** box.

   c. Click **OK** to generate the Web Service call.

   d. Add a **Wait for Event** step. Select **Insert > New Step** from the right-click menu and choose **web_service_wait_for_event** from the SOAP functions in the Steps Toolbox.

   e. Specify a step name, a quantifier, and a timeout. Click **Add** and insert the name of the event that you defined in the previous step.

   In Script view, VuGen indicates asynchronous messaging with the added parameter, **AsyncEvent**.

```
web_service_call("StepName=EchoString_101",
        "SOAPMethod=EchoRpcEncoded.EchoSoap.EchoString",
        "ResponseParam=response1",
        "Service=ExtendedECHO_rpc_encoded",
        "AsyncEvent=Event_1",
        "Snapshot=t1157371707.inf",
        BEGIN_ARGUMENTS,
```

```
        "sec=7",
        "strString=mytext",
        END_ARGUMENTS,
        BEGIN_RESULT,
        "EchoStringResult=first_call",
        END_RESULT,
        LAST);
```

The **AsyncEvent** flag instructs the Vuser to wait for the response of previous asynchronous service requests.

5. Send an asynchronous message using WS-Addressing. (Optional)
   a. Select the **Async Support** option and provide an event name in the **Async Event** box. This can be an arbitrary name.
   b. Select **WSA Support**. In the **WS-A Reply to** box, enter an IP address or **autodetect** to use the current host. Autodetect is useful when running the same script on several different machines. The server will reply to the specified location when the event occurs.
   c. Click **OK** to save the settings.
   d. Instruct the Vuser to wait for an event. Select **Insert > New Step** from the right-click menu and choose **web_service_wait_for_event** from the SOAP functions in the Steps Toolbox.
   e. Specify a step name, quantifier, and timeout. To add an event name, click **Add**. The Web Service will wait for the specified event before responding.
   f. Use the **Edit**, **Move Up**, and **Move Down** buttons to manipulate the events.

## Define a Testing Method

This task describes how to select a testing method.

1. **Open the step properties**

   In the Step Navigator, right-click the step whose response you want to test, and select **Show Arguments**.

2. **Select an argument**

   Select the Output Argument node. For details, see "New Web Service Call Dialog Box" on page 778.

3. **Select a testing method and choose an expected response**

   - To perform negative testing only, select the **Negative Testing** check box and choose **SOAP Fault** as the **Expected Response**.
   - To accept any type of SOAP response, select the **Negative Testing** check box and choose **Any SOAP** as the **Expected Response**.
   - To perform positive testing only, clear the **Negative Testing** check box.

4. **Verify function in the script**

In Script view, VuGen indicates the testing method with the **ExpectedResponse** argument. In the following example, the script performs negative testing, indicated by the **SoapFault** value:

```
web_service_call("StepName=AddAddr_101",
        "SOAPMethod=AddrBook|AddrBookSoapPort|AddAddr",
        "ResponseParam=response",
        "Service=AddrBook",
        "ExpectedResponse=SoapFault",
        "Snapshot=t1189409011.inf",
        BEGIN_ARGUMENTS,
        END_ARGUMENTS,
        BEGIN_RESULT,
        END_RESULT,
        LAST);
```

5. **Evaluate the SOAP fault value**

   When you replay a script that results in a SOAP fault, VuGen saves the fault to a parameter called **response**. To check the returned value of the SOAP fault, evaluate the **response** output parameter using **lr_xml_find.**

   In the following example, **lr_xml_find** checks for a **VersionMismatch** SOAP fault and issues an output message.

```
lr_xml_find("XML={response}",
                "FastQuery=/Envelope/Body/Fault/faultString ",
                "Value=VersionMismatch",
                LAST);
if (soap_fault_cnt >0)
     lr_output_message{"A Version Mismatch SOAP Fault occurred")
```

   For more information about **lr_xml_find**, see the Function Reference (**Help > Function Reference**).

## Add a Database Connection

This task describes how to add a database connection step through Tree view.

1. **Open Solution Explorer**

   Select **View > Solution Explorer**.

2. **Select a section**

   Select the desired section: **vuser_init** or **Action**. To avoid repeating the connection sequence in every iteration, place it in the **vuser_init** section.

3. **Insert a database connection step**

   Select **Design > Insert in Script > New Step**. Choose the **lr_db_connect** step. The Database Connection dialog box opens. Specify a **Step Name**, **Connection Name**, and **Data Provider**, OLEDB

or SQL.

4. **Create a database connection string**
   a. Click **Connection String Generator** to generate a database connection string specific to your environment.
   b. Indicate the connection properties:
      - **Server Name**
      - **Database Name**
      - **Authentication** method: Windows Authentication or User/password.
      - **Username** and **Password**
   c. Click **Test Connection** to verify that the information you provided is correct.
   d. Select an **SQL Provider**, OLEDB or SQL, and click **Generate**.
5. **Verify function in the script**

   Check that an **lr_db_connect** function was written to the script.

## Create a User Handler

This task describes how to write a user handler for your script.

1. **Prerequisite: Create a Web Service call**

   Import a WSDL file and create a standard Web Service Call. For details, see "Adding Web Service Script Content - Overview" on page 762.
2. **Define a user handler function**

   Define a user handler before the Web Service call:

```
int MyScriptFunction(const char* pArgs, int isRequest)
{
...
}
```

3. **Call the user handler function**

   Call the handler function by specifying the function name as a value for the **UserHandlerFunction** argument. in the Web Service Call.

```
web_service_call(
...
"UserHandlerFunction=MyScriptFunction",
"UserHandlerArgs=<handler arguments>",
```

```
LAST);
```

4. **Evaluate the handler function**

   Evaluate the handler's return code to determine if it succeeded. Use the return codes as described in " User Handlers" on page 797.

```
//This function processes the SOAP envelope before sending it to the server.
int MyScriptFunction(const char* pArgs, int isRequest)
{
    if (isRequest == 1) {
        //Get the request that is going to be sent
        char* str = lr_eval_string("{SoapEnvelopeParam}");
        //Manipulate the string...
        //Assign the new request content
        lr_save_string(str, "SoapEnvelopeParam");
        return LR_HANDLER_SUCCEEDED_AND_MODIFIED;
    }
    return LR_HANDLER_SUCCEEDED;
}
```

5. **Create a DLL file. (Optional)**

   To define a user handler through a DLL, locate the API header file, **LrWsHandlerAPI.h** in the product's **include** folder.

   You can use a sample Visual Studio project located in the samples/WebServices/SampleWsHandler folder as a template for creating a handler. The sample retrieves the request and response envelope and saves it to a parameter. To use this sample, open it in Visual Studio and modify it as required. If you do not need to save the request/response to a parameter, you can remove that section of the sample.

   After editing the sample, save it and compile the DLL. When you compile the project, Visual Studio places the **<user_handler_name>.DLL** file in the **bin** folder. If you compile the project from another location, or if you want to copy the DLL from one machine to another, make sure to place it in the **bin** folder.

6. **Configure the user handler. (Optional)**

   Declare the DLL user handler globally or locally.

   To apply the user handler globally to all requests in the script, add the following section to the **default.cfg** file in the script's folder.

```
[UserHandler]
Function=<name>
Args=<arguments>
Order=<BeforeSecurity/AfterSecurity/AfterAttachments>
```

- **Name.** The name of the DLL.
- **Args.** A list of the configuration arguments for the handler. Use the **GetArguments** method to retrieve the arguments in your handler.
- **Order.** The order in which Vusers process the user handler in requests: **Before Security**, **After Security**, or **After Attachments**. You can also use this argument to override the transport layer, by entering the value **Replace Transport**.

> **Note:** Setting the **UserHandlerFunction** property of a **web_service_call** function, overrides the definitions in the .cfg file.

By default, user handlers are processed before the security. For request messages, Vusers process the attachments handler after the security handler. For responses, Vusers process the handlers in a reversed order. In typical cases the order does not matter, so any value is acceptable.

To override the Transport layer, specify **Order=Replace Transport** and specify the new transport handler. If you implement the transport handler as a separate DLL, the **HandleRequest** function is called, while the **HandleResponse** function is ignored.

To use the handler locally, for a specific request, add the following arguments to the **web_service_call** function:

```
UserHandlerName=<name1>
UserHandlerArgs=<args1>
UserHandlerOrder=<BeforeSecurity/AfterSecurity/AfterAttachments/Replace
                                  Transport>
```

7. **Copy the user handler to all required machines**

   Make sure that the user handler DLL is accessible to all Load Generator machines running scripts that call it. You may, for example, copy it to the product's **/bin** folder.

   If you copy the script to another machine, it retains the handler information, since it is defined in script's folder.

8. **Implement the user handler. (Optional)**

   To implement a user handler, you use the entry functions **HandleRequest** or **HandleResponse**. Both functions have a single parameter, **context**, whose properties you can set in your handler. Use the Get functions to retrieve properties, and Set functions to pass information from the replay framework to the handlers or between the handlers.

   - **GetEnvelope.** Gets the envelope content. For example:
     ```
     const char * pEnvelope = context->GetEnvelope();
     ```
   - **GetEnvelopeLength.** Gets the envelope length
   - **SetEnvelope.** Sets the envelope content and length. For example:
     ```
     string str("MySoapEnvelope...");
     context->SetEnvelope(str.c_str(), str.length());
     ```

- **SetContentType.** Sets a new value for HTTP header content type
- **LogMessage.** Issues a message to the replay log
- **GetArguments.** Gets the configuration arguments defined for the current handler in order to pass it to the DLL
- **GetProperty.** Gets a custom property value
- **SetProperty.** Sets a custom property value

For more information, see the comments in the **LrWsHandlerAPI.h** file located in the product's **include** folder.

## Customize Configuration Files

The following steps describe how to modify configuration files. For details, see " Custom Configuration Files" on page 799.

### Locate the configuration file

Determine the location of the configuration file. The standard .NET configuration file, **mmdrv.exe.config** is located in the product's **bin** folder. Some applications have their own file, **app.config**.

### Save the application's configuration file

If your application has its own app.config file:

- To apply the configuration information globally to all scripts, save the **app.config** file as **mmdrv.exe.config** in the **bin** folder, overwriting the existing file.
- To apply the configuration information locally, specifically for this script, copy the **app.config** file to the script's folder. This overrides the **mmdrv.exe.config** file, and remains associated with this script even when you copy it to other machines.

### Set the security - optional

By default, VuGen allows unsigned certificates to facilitate testing. To disallow unsigned certificates, modify the **allowTestRoot** flag in the <security> section to **false**.

```
<security>
   <x509 storeLocation="currentuser" alllowTestRoot="false"
```

## Web Services Snapshots - Overview

Vuser scripts based on the Web Services protocol utilize VuGen's Snapshot pane.

The Snapshot pane enables you to view snapshots of Web service calls. When you display the Snapshot pane for a Web Services script, the left side of the Snapshot pane displays a tree view of the snapshot data; the right side of the Snapshot pane displays a text view of the snapshot data.

The tree view on the left of the Snapshot pane is composed of a number of nodes. An icon to the left of each node indicates the type of the node:

- **Element:** Indicates that the node represents an element in the XML file.
- **Attribute:** Indicates that the node represents an attribute in the XML file.
- **Value:** Indicates that the node represents a value in the XML file.

In addition to the basic Snapshot pane functionality, the Snapshot pane for Web Services scripts includes additional functionality. The UI for this additional functionality is described below.

| | |
|---|---|
| **To access** | Select **View > Snapshot**, or click the **Show Snapshot Pane** button on the VuGen toolbar. |
| **Relevant tasks** | "Prepare Scripts for Replay" on page 801 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | Displays a snapshot of the SOAP response returned by the server. |
| | Displays a snapshot of the SOAP request sent to the server by the Web Service call. |
| | Opens the XPath Search dialog box which enables you to perform an XPath search of the snapshot. |
| | Displays the previous or next results of the XPath search. |
| | Displays or hides the XML **attribute** nodes in the tree view of the snapshot. |
| | Displays or hides the XML **value** nodes in the tree view of the snapshot. |

| | |
|---|---|
| **<shortcut menu>** | • **Copy Selection.** Copies the text that is selected in the text view to the clipboard.<br>• **Search Community.** Performs a community search using the text that is selected in the text view as the search string. For details about performing a community search, see "Editor Pane" on page 77.<br>• **Copy XPath.** In the tree view, copies the XPath of the selected node to the clipboard. In the text view, copies the XPath of the XML element in which the cursor is located to the clipboard.<br>• **Copy full value.** In the tree view, copies the full XML code of the selected node to the clipboard. In the text view, copies the full XML code of the XML element in which the cursor is located.<br>• **Insert XML Check.** Opens the Insert XML Check dialog box that enables you to insert an **XML Find** step into the Vuser script.<br>**Note:**<br>• This option is available in the Response view only.<br>• This option is available for **attribute** and **value** nodes only.<br>• **Save value in parameter.** Opens the Save Value as Parameter dialog box that enables you to save the selected value to a simple parameter.<br>**Note:**<br>• This option is available in the Response view only.<br>• This option is available for attribute and value nodes only.<br>• **Save XML in parameter.** Opens the Save Value as Parameter dialog box that enables you to save the selected value to an XML parameter.<br>This option is available in the Response view only.<br>• **Create Correlation.** Opens the Correlation tab in the Design Studio. The text selected in the Snapshot pane appears as a manual correlation entry in the Design Studio. For details, see "Manually Correlate Scripts" on page 260.<br>**Note:**<br>• This option is available in the Response view only.<br>• This option is available for attribute and value nodes in the tree view, and when text is selected in the text view.<br>• **Create Correlation Rule.** Opens the Add as Rule dialog box that enables you to add the selected text as part of a correlation rule. For details, see "Correlating Values" on page 254. |

| | |
|---|---|
| | **Note:**<br>• This option is available in the Response view only.<br>• This option is available for attribute and value nodes in the tree view, and when text is selected in the text view. |

### See also:

- "Work with Snapshots" on page 297
- "Snapshot Pane" on page 85 for details on the standard Snapshot pane UI

## Database Connection Dialog Box

This dialog box helps you create a string to connect to your database.

| | |
|---|---|
| **To access** | Click **Connection String Generator** in the Database Connection dialog box. |
| **Relevant tasks** | "Send Messages over JMS" on page 803 |
| **See also** | "Connection String Generator Dialog Box" below |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Connection String Generator | Opens the Connection String Generator. For details, see "Connection String Generator Dialog Box" below. |
| **Step Name** | The name or IP address of the database server. |
| **Connection String** | The string by which to connect to the database. Use the **Connection String Generator**. |
| **Data Provider** | The SQL provider: **OLEDB** or **SQL**. |

## Connection String Generator Dialog Box

This dialog box helps you create a string to connect to your database.

| | |
|---|---|
| **To access** | Click **Connection String Generator** in the Database Connection dialog box. |
| **Relevant tasks** | "Send Messages over JMS" on page 803 |
| **See also** | "Database Connection Dialog Box" above |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Test Connection... | Tests the connection to the database. |
| Generate | Generates the database connection string and writes it in the **Connection String** field in the Database Connection dialog box. |
| **Server Name** | The name or IP address of the database server. |
| **DB Name** | The name of the database. |
| **Authentication** | The authentication method for the database: **Windows Authentication** or **User/password**.<br>• **User Name, Password**. The credentials for the database. |
| **SQL Provider** | The SQL provider: **OLEDB** or **SQL**. |

## Web Services - Managing Services

Import, manage, and compare Web services in VuGen.

### Importing Services

VuGen lets you import services for the purpose of creating a high-level tests with Web Service Call steps. Typically, you begin creating a script by importing a WSDL file.

The Import mechanism requires the following information:

- **Source.** The source of the WSDL: URL, File, UDDI, or Application Lifecycle Management. UDDI is a universal repository for services (Universal Description, Discovery, and Integration). Service brokers register and categorize published Web Services and provide search capabilities. The UDDI business registry is an example of a service broker for WSDL-described Web Services.
- **Location.** the path or URL of the WSDL, entered manually or by browsing.
- **Toolkit.** The toolkit to permanently associate with all services in the script for all subsequent imports and replays (only available for the first service added to the script). The toolkit setting instructs VuGen to send real client traffic using an actual toolkit—not an emulation.

  VuGen supports the .NET Framework with WSE 2 version SP3 and Axis/Java based Web Services Framework toolkits. VuGen imports, records, and replays the script using the actual .NET or Axis toolkit. By default, VuGen uses automatic detection to determine the most appropriate toolkit.
- **Connection Settings.** Authentication or proxy server information. This setting is useful for WSDLs residing on secure servers, or WSDLs that must be accessed via a proxy server.

If VuGen detects a problem with your WSDL when attempting to do an import, it issues an alert and prompts you to open the report. The report lists the errors and provides details about them.

For task details, see "Add and Manage Services" on page 816.

## Comparison Reports

VuGen lists the differences between the files in a Comparison report.

You can configure the comparison settings, indicating which items to ignore during the comparison. For more information, see the "XML/WSDL Comparison Dialog Box" on page 823.

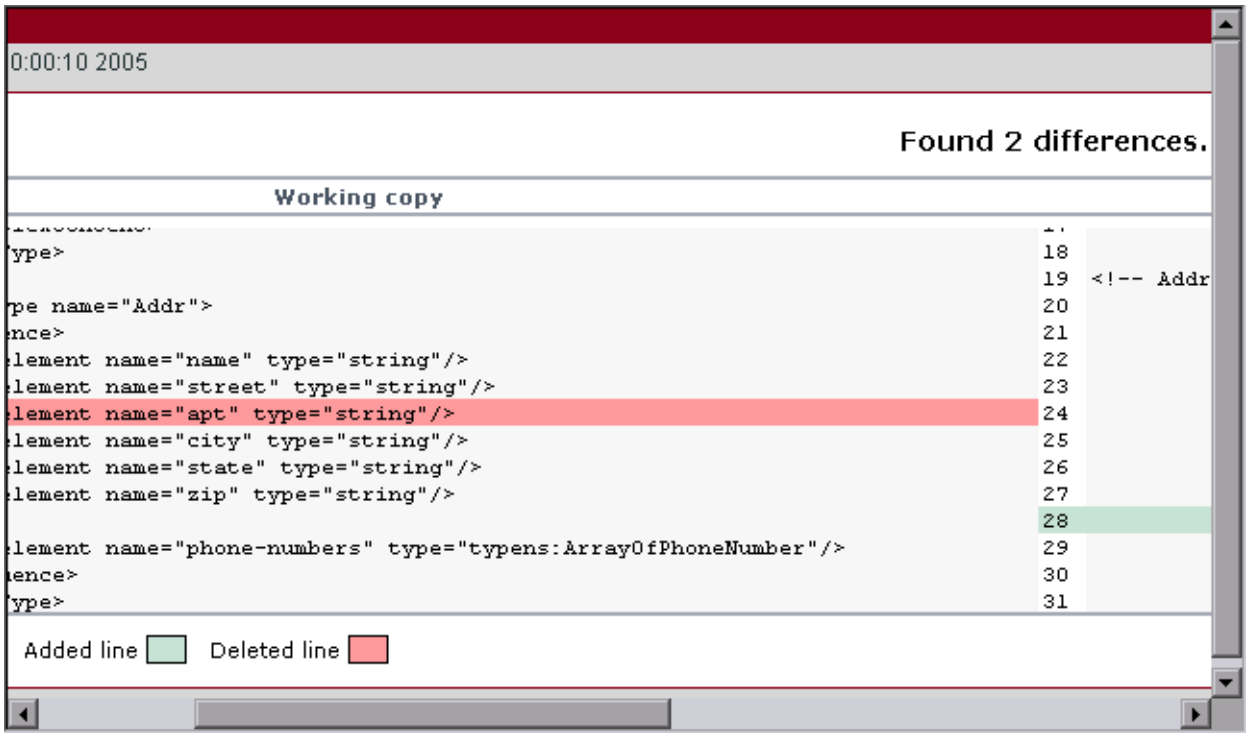
In WSDL Comparison reports, there are two columns— **Working Copy** and **Original File.** The Working Copy is the WSDL stored with the script, while the Original File is the WSDL at its original location—a network file path or a URL.

In XML Comparison reports, each column displays the path of an XML file.

The Comparison report uses the following legend to mark the differences between the two files:

- **Yellow.** Changes to an existing element (shown in both versions).
- **Green.** A new element added (shown in the original file copy).
- **Pink.**A deleted element (shown in the working copy).

In the following example, line 24 was deleted from the original copy and line 28 was added.




## Web Reference Analyzer

Many WSDL files reference other files such as XSD and other XML files. Before running a script, you may want to determine what these files are and if they are available.

VuGen's WSDL Reference Analyzer checks the WSDL for dependencies, and lists them in the WSDL Reference Analyzer window and in a log file.

The Analyzer places the WSDL and its dependent files in a zip archive file. It saves the dependency information to a log file, listing its path in the Analyzer window.

For user interface details, see "WSDL Reference Analyzer Dialog Box" on page 824.

For task details, see "Analyze WSDL Dependencies" on the next page.

## Add and Manage Services

This task describes how to create a list of services that you can call from your test. Using the Manage Services window, you import services and configure their settings.

### Open the Manage Services Dialog Box

Select **SOA Tools > Manage Services** or click the toolbar button to open the Manage Services dialog box.

### Import a Service

Click **Import**. In the Import Service dialog box, select a WSDL source and browse to the location.

For **URL** type imports, the Browse button opens a new browser. Navigate to the WSDL and then close the browser. This action places the URL in the location box. For details, see the "Import Service Dialog Box" on page 822.

If your service requires authentication or uses a proxy, configure these settings before importing the WSDL. Expand the Import Services dialog box and click **Configure**. For details, see the "Connection Settings Dialog Box" on page 821.

Repeat this step for all the services you want to include in your test.

### Get to Know the WSDL

Familiarize yourself with the WSDL. View its details as described in the "Manage Services Dialog Box" on the next page.

Click **View WSDL** to open the locally saved WSDL file in Internet Explorer and study its structure.

### Check for WSDL Updates - Optional

Use the Comparison tool to check that the WSDL did not change since your last import or update.

First, set the comparison options. Click **SOA Tools > SOA Settings > XML/WSDL Comparison**. Specify what differences to ignore. For details, see "XML/WSDL Comparison Dialog Box" on page 823.

In the Manage Services window, click **Compare** to open a report comparing the working copy of the WSDL with the one at the original location.

If you discover changes in the Comparison report, click **Update Now** to retrieve the latest version of the WSDL from its source.

### Override the Service Address- Optional

View the address in the **Service Address** box. This is the default endpoint address as retrieved from the WSDL. If you want to override it, select **Override address** and type in an alternate endpoint address for the service requests.

To return to the default address, clear the **Override address** option. For details, see the "Manage Services Dialog Box" below.

### Set a Security Scenario - Optional

Click the **Protocol and Security** tab to use WS-Security or another type of a security scenario.

For more information, see "Web Services - Security" on page 824.

## Analyze WSDL Dependencies

This task describes how to use the Reference Analyzer to determine WSDL dependencies. For user interface details, see "WSDL Reference Analyzer Dialog Box" on page 824.

1. Open the Reference Analyzer

   Select **SOA Tools > WSDL Reference Analyzer**.
2. Select a source and target

   In the **Select WSDL file** box, indicate the location of the WSDL you want to analyze.

   In the **Output file path** box, indicate a location for the zip file.
3. Begin the analysis

   Click **Start Analyzing**. The Analyzer lists all of the dependencies in the output window along with their paths.
4. View the log

   View the results in the log window. To clear the results and perform another analysis, click **Clear Log**.

## Manage Services Dialog Box

This dialog box enables you to manage your WSDLs, provide authentication information, and set a security scenario.

| | |
|---|---|
| **To access** | Use one of the following:<br>• Click the **Manage Services** button on the VuGen toolbar.<br>• **SOA Tools > Manage Services** |
| **Relevant tasks** | "Add and Manage Services" on the previous page |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | Opens the Import Service dialog box.<br><br>**Select WSDL from.** Enter the location of the WSDL. Browse for the information or enter it manually:<br><br>• **URL.** Complete URL. Make sure to insert a complete URL—not a shortened version.<br>• **File.** Full path and file name.<br>• **UDDI.** UDDI registry ID. The Browse button opens the Search for Service in UDDI Dialog Box.<br><br>**Connection Settings**. Click to configure the authentication and proxy settings of the server hosting the WSDL. For details, see "Connection Settings Dialog Box" on page 821.<br><br>**Advanced Settings**. Select a toolkit for the test.<br><br>**Note:** The **Automatic** setting uses an algorithm to determine the most suitable toolkit. |
| | Removes the selected service from the list. |
| | Opens the WSDL Comparison Report showing the Working copy and Original copy of the WSDL side-by-side.<br>To set the comparison settings, see "XML/WSDL Comparison Dialog Box" on page 823. |
| | Displays the WSDL in a browser. |
| **<WSDL list>** | A list of the imported WSDLs. |
| **Description tab** | Provides information about the WSDL, its endpoint address, toolkit, and update information. |
| **Operations tab** | Lists the operations of the service: **Operation Name**, **PortName**, and **Used in Script** (Yes or No). |

| | |
|---|---|
| **Connection Settings tab** | In some cases WSDLs reside on secure sites requiring authentication. In certain instances, the WSDL is accessed through a proxy server.<br>VuGen supports the importing of WSDLs using security and WSDLs accessed through proxy servers. The following security and authentication methods are supported:<br>• SSL<br>• Basic and NTLM authentication<br>• Kerberos for the .NET toolkit<br>For more information about setting the connection information while importing the WSDL, see "Add and Manage Services" on page 816.<br>**Note:** This only applies to URL and UDDI type imports. |
| **UDDI Data tab** | The UDDI server, UDDI version, and service key.<br>You can view the details of the UDDI server for each service that you imported from a UDDI registry.<br>The read-only information indicates the URL of the UDDI server, the UDDI version, and the Service key. |
| **Protocol and Security tab** | Allows you to view and set a security scenario for your Web Service calls and shows the details of the security scenario applied to the script.<br>If you did not choose a scenario, it uses the default **<no scenario>**. For more information, see "Add and Manage Services" on page 816. |

### Description Tab

Displays information about the service.

| UI Element | Description |
|---|---|
| Update Now | Loads the latest version of the WSDL from its original location. |
| **Created By** | The name with which you logged in. You can edit this field and specify a different name. This is useful for sorting the services in reports (read-only). |
| **Description** | An editable field into which you can type information about the service. |
| **Last update from original** | The last date and time the WSDL was updated (read-only).<br>You can update the version manually or retrieve it automatically each time you reopen the test. |

| | |
|---|---|
| **Original Location** | The original location from where the WSDL was imported (read-only). |
| **Override address** | Enables you to enter an alternate endpoint for the service in the **Service Address** box. |
| **Service Address** | The endpoint of the service to which service requests are sent, retrieved from the WSDL file (read only). To override the default address, select **Override address**. |
| **Service Name** | The native service name in the WSDL file that is displayed by default when importing the service (read-only). |
| **Toolkit** | The toolkit associated with the service. You set this when you import the service (read-only). |
| **Update when opening script** | Updates the WSDL from its source each time you open the script. |

### Connection Settings Tab

The following elements are included:

| UI Element | Description |
|---|---|
| **Authentication** | **Use Authentication Setting:** Enables you to enter credentials for authentication.<br>• **Username, Password.** The user name and password to use for retrieving the WSDL.<br>**Tip:** For users not in the default domain, type the domain name before the user name. For example, `domain1/alex_qc`. |
| **Proxy** | **Use Proxy Setting**. Enables you to enter proxy details and credentials.<br>• **Server.** Name or IP address of proxy server.<br>• **Port.** Port through which to access the WSDL.<br>• **Username, Password.** the user name and password to be used for authentication. For users not in the default domain, type the domain name before the user name. For example, `domain1/alex_qc`. |

### UDDI Data Tab

The following elements are included:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Service Key** | A unique identifier of the service on the UDDI server, used to locate the service definition when updating the service. |
| **UDDI Server** | The URL address and version of the UDDI server from which the service definition is imported. |
| **UDDI Version** | The version of the UDDI registry: 2 or 3. |

## Connection Settings Dialog Box

Enables you to provide authentication credentials and proxy server details for the machine hosting the WSDL file.

| | |
|---|---|
| **To access** | • For a new service: Select **SOA Tools > Manage Services.** Click the **Import** button. In the Import Services dialog box, click **Connection Settings**.<br>• For existing services: Select a service in the Mange Services dialog box, and click the **Connection Settings** tab. |
| **Important information** | Only available for services imported through a URL and UDDI. |
| **Relevant tasks** | "Add and Manage Services" on page 816 |

The following elements are included:

| UI Element | Description |
|---|---|
| **Authentication** | **Use Authentication Setting:** Enables you to enter credentials for authentication.<br>• **Username, Password.** the user name and password to use for retrieving the WSDL.<br>**Tip:** For users not in the default domain, type the domain name before the user name. For example, `domain1/alex_qc`. |
| **Proxy** | **Use Proxy Setting**. Enables you to enter proxy details and credentials.<br>• **Server.** Name or IP address of proxy server.<br>• **Port.** Port through which to access the WSDL.<br>• **Username, Password.** the user name and password to be used for authentication. For users not in the default domain, type the domain name before the user name. For example, `domain1/alex_qc`. |

## Import Service Dialog Box

Enables you to import WSDLs from a file system, a URL, Application Lifecycle Management, a UDDI, or Systinet.

| | |
|---|---|
| **To access** | Use one of the following:<br>• Select **Services > New > Import Services**<br>• Select **New > Import Services** from the shortcut menu |
| **Relevant tasks** | "Add and Manage Services" on page 816 |

The following elements are included:

| UI Element | Description |
|---|---|
| ... | **Browse.** Enables you to locate a service on the file system, through a browser, UDDI registry, or Application Lifecycle Management repository depending on your **Import WSDL from** selection. |
| Connection Settings ... | Opens the Connections Settings dialog box for configuring the authentication and proxy settings of the server hosting the WSDL. For details, see "Connection Settings Dialog Box" on the previous page. |
| Advanced Settings... | Allows you to select a toolkit for the test. Choose **Automatic**, **.NET**, or **Axis**. The **Automatic** setting uses an algorithm to determine the most suitable toolkit. |
| Import | Begins the import process. |
| **Select WSDL from** | Location of WSDL. Browse for the information or enter it manually:<br>• **URL:** Complete URL. Make sure to insert a complete URL—not a shortened version.<br>• **File:** Full path and file name.<br>• **UDDI:** UDDI registry ID. The Browse button opens the "Search for Service in UDDI Dialog Box" below. |

## Search for Service in UDDI Dialog Box

This dialog box enables you to locate a specific service from a UDDI registry.

| | |
|---|---|
| **To access** | 1. In the Manage Services window, click **Import**.<br>2. In the Import Service dialog box, select **UDDI** in the **Select WSDL from** section.<br>3. Click ... . |
| **Relevant tasks** | "Add and Manage Services" on page 816 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| Search | Begins the search for a service based on the text in the **All or part of the service name** box. |
| **<service list>** | An alphabetical list of all the services that match the string. The grid shows the following columns: **Service Name**, **Service Key**, **Service Description**, **Service WSDL**. |
| **All or part of the service name** | A string including the desired service name or part of the name. You do not need to use wildcard expressions.<br>The following options narrow the search:<br>• **Exact Match.** The service name must exactly match your text.<br>• **Case Sensitive.** The case of service name must match the case of the specified text. |
| **UDDI server inquiry address** | The complete path for the inquiry on the UDDI server. |
| **UDDI Version 2/3** | The UDDI version of the services to display in the list. |

## XML/WSDL Comparison Dialog Box

This dialog box enables you to configure the settings for comparing different versions of a WSDL You can instruct the comparison tool to ignore specific differences such as case, comments, and so forth.

| | |
|---|---|
| **To access** | **SOA Tools > SOA Settings > XML/WSDL Comparison.** |
| **Relevant tasks** | "Add and Manage Services" on page 816 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Show only differences** | Show only differences in the report—do not display the matching text. |
| **Ignore case** | Do not show case mismatches as differences. |
| **Ignore comments** | Do not mark mismatches in the comment as differences. |
| **Ignore processing instructions** | Do not mark mismatches in the processing instructions as differences. |
| **Ignore namespaces** | Do not mark mismatches in namespaces as differences. |

### WSDL Reference Analyzer Dialog Box

This dialog box enables you to determine the dependencies of a WSDL file.

| | |
|---|---|
| **To access** | **SOA Tools > WSDL Reference Analyzer** |
| **Relevant tasks** | "Analyze WSDL Dependencies" on page 817 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| Start Analyzing | Begins the analysis, showing all the results in the Log window. |
| Clear Log | Clears the log window and log file. |
| Open Containing Folder | Opens the folder containing the output file. |
| **<log window>** | A running log of the reference analysis. |
| **Select WSDL file** | The local path or URL of the WSDL file to analyze. |
| **Output file path** | A location for the output zip file. |

## Web Services - Security

### Setting Security Overview

When building Web Service applications, there is a challenge in building scalable applications that are secure. You can secure Web Services by having the message sent over a secure transport, such as Secure Sockets Layer (SSL), but this is limited to point-to-point communication.

To allow you to send your messages securely, VuGen supports several security mechanisms, Security Tokens (WS-Security), and SAML.

The following table lists the considerations for using each of the models.

| Legacy Model | Scenario Based Model |
|---|---|
| You are working with a script that already uses the legacy model | You are testing a WCF Service. |
| You are testing a service written in frameworks such as .NET 2.0, Axis, or other older toolkits | You are testing a service written in a new framework, such as Axis2 or Metro (WSIT). |

| | |
|---|---|
| You require a low-level control over WS-Security tokens | Your service uses advanced specifications such as WS-SecureConversation or WS-Trust. |
| You are having trouble using the new model or find the capabilities of the legacy more adequate for your needs | You are having trouble using the legacy model or you find the capabilities of the new model more adequate. |

**Note:** If your WSDL is located in a secure location, you must provide the security information through the Manage Services dialog box. For more information, see the "Connection Settings Dialog Box" on page 821.

## See also:

- "Security Tokens and Encryption" below
- "SAML Security Options" on page 827

### Security Tokens and Encryption

The WS-Security specification lets you place security credentials in the actual SOAP message. You accomplish this by instructing a client to obtain security credentials from a source that is trusted by both the sender and receiver. When a SOAP message sender sends a request, those security credentials, known as security **tokens**, are placed in the SOAP message. When the Web server receives the SOAP request, it does not need to send additional requests to verify the integrity of the sender. The server verifies that the credentials are authentic before letting the Web Service execute the application. By not having to go back to the source of the credentials, this significantly improves the application's scalability.

To further secure Web Services, it is common to use digital signatures or encryption for the SOAP messages. Digitally signing a SOAP message verifies that the message has not been altered during transmission. Encrypting a SOAP message helps secure a Web Service by making it difficult for anyone other than the intended recipient to read the contents of the message.

The Web Services security mechanism associates security tokens with messages. This mechanism supports several security token formats to accommodate a variety of authentication requirements. For example, a client might need to provide a proof of identity or a security certificate.

To support WS-Security, VuGen allows you to create security tokens for your script. You can create multiple tokens and set their properties. After creating a token, you use it to sign or encrypt a SOAP message.

In certain instances, you do not send the token explicitly—you use the token for the purpose of signatures or encryption, without including the actual token in the SOAP envelope header. Using the **Security Token** option, you can indicate whether to send the actual token explicitly.

## Available Security Tokens

The available tokens are **Username and Password**, **X.509 Certificate**, **Kerberos**, **Kerberos2**, and **PFX File**. The information you need to provide differs for each token.

- **UserName Token.** The **UserName**token contains user identification information for the purpose of authentication: **User Name** and **Password**.

  You can also specify Password Options, indicating how to send the password to the server for authentication: **SendPlainText**, **SendNone**, or **SendHashed**.

- **X.509 Certificate Token.** This security token is a token based on an X.509 certificate. To obtain a certificate, you can either purchase it from a certificate authority, such as VeriSign, Inc. or set up your own certificate service to issue a certificate. Most Windows servers support the public key infrastructure (PKI) which enable you to create certificates. You can then have it signed by a certificate authority or use an unsigned certificate.

  When you add an X.509 token to the Vuser script, you specify the **Token Name**, **Certificate**, and **Reference type**. The Browse button opens the "Select Certificate Dialog Box" on page 853 which allows you to find a certificate from a Windows store.

- **Kerberos / Kerberos2 Tokens.** (for Windows 2003 or XP SP1 and later) The Kerberos protocol is used to mutually authenticate users and services on an open and unsecured network. Using shared secret keys, it encrypts and signs user credentials. A third party, known as a KDC (Kerberos Key Distribution Center), authenticates the credentials. After authentication, the user may request a service ticket to access one or more services on the network. The ticket includes the encrypted, authenticated identity of the user. The tickets are obtained using the current user's credentials.

  VuGen supports tokens based on both Kerberos and Kerberos2 security tokens. The primary difference between the Kerberos and Kerberos2 tokens is that Kerberos2 uses the Security Support Provider Interface (SSPI), so it does not require elevated privileges to impersonate the client's identity. In addition, the Kerberos2 security token can be used to secure SOAP messages sent to a Web Service running in a Web farm.

  When you add a Kerberos token to the Vuser script, you specify a logical **Token Name** for the token along with the **Host** and **Domain** names of the Web Services machine.

- **PFX File Token.** These Personal Information Exchange token files (with **.pfx** or **.p12** extensions) contain all of the token information, including the server certificate, intermediate certificates, and the private key in a single file. This file uses the PKCS#12 (Personal Information Exchange Syntax) standard. Click the Browse button to locate the file.

> **Tip:** If you have a JKS keystore (**.jks** file), VuGen automatically converts it to a **.p12** file.

For details about the token attribute in the script, see the Function Reference (**Help > Function Reference**).

## Adding the Security Policy

To add a security policy to a section of your script, you enclose the relevant steps with **Web Service Set Security** and **Web Service Cancel Security** steps.

When you add a **Web Services Set Security** step to your script, VuGen adds a **web_service_set_security** function that contains arguments with the tokens, message signatures, and encryption that you defined.

```
web_service_set_security(
   SECURITY_TOKEN, "Type=USERNAME", "TokenName=mytoken1",    "UserName=bob",
"Password=123", "PasswordOptions=SendNone",    "Add=True", LAST);
```

Parameterization is not supported for the following arguments: **Token Type, Logical Name, Base Token, Issuer Token** or **Derive From** arguments.

## Working with Message Signatures and Encrypted Data

When you add a security token to a SOAP message, it is added to the SOAP message in the form of an XML element in the WS-Security SOAP header.

The message, however, is exposed and therefore requires additional security. This is especially true when the credentials, including the password, are sent in plain text as it is with role-based security.

The two methods used to secure the data are digital signatures and encryption.

- **Digital Signatures.** Digital Signatures are used by message recipients to verify that messages were not altered since their signing. The digital signature is usually in the form of XML within the SOAP message. The recipient checks the signature to make sure it is valid. Certain environments, such as WSE, automatically verify the signature on the SOAP recipient's computer.
- **Encryption.** Although the XML digital signature offers a mechanism for verifying that the message has not been altered since it was signed, it does not encrypt the SOAP message—the message is still plain text in XML format. To secure the message in order that it should not be exposed, you encrypt it, making it difficult for an intruder to view and obtain a user's password.

Parameterization is not supported for message signatures and encryption arguments. For details on adding message signatures and encryption to your script, see "Add Security to a Web Service Script" on page 838.

### SAML Security Options

VuGen supports SAML (Security Assertion Markup Language) for Web Services. SAML is an XML standard for exchanging security-related information, called **assertions**, between business partners over the Internet. The assertions can include attribute statements, authentication, decision statements, and authorization decision statements.

SAML uses brokered authentication with a security token issued by STS (Security Token Service). The STS is trusted by the client and the Web Service to provide interoperable security tokens. SAML tokens are important for Web Service security because they provide cross-platform interoperability and a

means of exchanging information between clients and services that do not reside within a single security domain.

You can set the SAML settings for an entire script or part of the script. For details, see "Add SAML Security" on page 841.

> **Note:** You cannot apply SAML security and the standard Web Service (a **Web Service Set Security** step) security to the same step. To cancel Web Service security, insert a **Web Service Cancel Security** step.

### Signing SAML Assertions

VuGen provides a method for signing an unsigned SAML assertion. As input, you provide the unsigned assertion, a certificate file, and the optional password. As output, VuGen provides the signed SAML assertion. For task details, see "Add SAML Security" on page 841.

### Policy Files

SAML policy files follow the WSE 3.0 standard and define the attribute values for the SAML security. By default, VuGen uses the **samlPolicy.config** file located in the installation's **dat** folder.

When entering SAML security information, you can enter it manually in the properties dialog box, or you can refer to a policy file containing all of the security information. You can create your own policy file based on samlPolicy.config.

You can modify the policy file to include values for the security parameters, such as username and certificate information. When adding a SAML security step to your script, if you explicitly specify values for the security arguments, they override the values in the policy file.

If you make changes to the default policy file, we recommend that you copy the new policy file to your script's folder. Make sure to save custom policy files with a **.config** extension to insure that they remain with the script, even when running it on other machines or calling it from the LoadRunner Controller.

To learn more about the SAML policy files, see the SAML STS example on the MSDN Web site. If you want to emulate SAML Federation behavior, copy the **samlFederationPolicy.config** file from the data folder to your script's folder, and specify it as the policy file.

## Security Scenarios Overview

VuGen allows you to test Web Services that utilize advanced security and WS-Specifications. Such services can be written in various platforms such as WCF (Windows Communication Foundation), Metro (WSIT), and Axis2. For WCF services, VuGen also supports proprietary standards and transports.

You enable this support by setting up a security scenario. Each scenario represents a typical environment used in conjunction with Web Service calls. VuGen provides several built-in security scenarios that are commonly used. It applies the scenario's settings individually to each service.

For the built-in scenarios, the user interface lets you provide identity information where required. You can customize security, transport, proxy, and other advanced settings.

If you cannot find a scenario that corresponds to your environment, you can use the generic custom scenario.

### See also:

- "Security Scenario Editor Dialog Box" on page 848

## Choosing a Security Model

VuGen supports two models for configuring security for your Web Service calls: *Legacy* (no scenario) and *Scenario*. This chapter describes the Scenario security models. The Legacy model refers to the manual addition of Web Service Set Security steps, or the **web_service_set_security** function.

The following table lists the considerations for using each of the models.

| Legacy Model | Scenario Based Model |
|---|---|
| You are working with a script that already uses the legacy model | You are testing a WCF Service |
| You are testing a service written in frameworks such as .NET 2.0, Axis, or other older toolkits | You are testing a service written in a new framework such as Axis2 or Metro (WSIT). |
| You require a low-level control over WS-Security tokens | Your service uses advanced specifications such as WS-SecureConversation or WS-Trust |
| You are having trouble using the new model or find the capabilities of the legacy functions adequate | You are having trouble using the legacy model or you find the capabilities of the new model more adequate |

## Private, Imported, and Shared Scenarios

To assign a security scenario to a specific service, use the Manage Services window. The **Protocol and Security** tab contains the interface to create and view security scenarios for individual services.

You can select a scenario in three ways:

- **Private scenario.** Create a new scenario by selecting one of the built-in ones and customizing it for your Web Service.
- **Imported scenario.** Use a scenario created at an earlier time. The scenario will be editable, and if someone modifies the original scenario, it will not affect you.
- **Shared scenario.** Load a security scenario already configured by another user from a remote location or the file system. You cannot edit this scenario's settings from the Manage Services window. If someone edits the scenario, it will affect your environment. You usually use this option after working with the product for some time and saving the scenario files.

## Scenario Categories

The scenario describes the configuration of your Web Service. It contains information such as security, encoding, proxy, and so forth. VuGen provides a Security Scenario editor that allows you to configure the settings for each scenario.

To determine the scenario that best fits your service, refer to the sections below. If you are unsure which scenario to choose, we recommend to use the **Custom Binding** scenario. For more information, see "The Custom Binding Scenarios" on page 834.

Use the default **<no scenario>** for:

- simple Web Services where no advanced standards are required.
- scripts that use the legacy security model
- Web Services that require a specific security setting, not available in any of the existing scenarios.

If you select a built-in scenario and experience problems in replay, it is possible that no scenario was required and the problem is elsewhere. Reset the value to **<no scenario>.**

The built-in security scenarios are divided into the following categories:

### Core Scenarios

The following table describes the built-in Core scenario.

| Scenario Name | When to use |
|---|---|
| Plain SOAP | • Web services which do not require advanced standards<br>• Web services which may require you to specify the WS-Addressing version |

For this type of scenario, if your service uses WS-Addressing, specify the version.

### Security Scenarios

The following table describes the built-in Security scenario.

| Scenario Name | When to use |
|---|---|
| Username Authentication | • Client is authenticated with a username and password on the message level |

For this type of scenario, specify the username/password, and if your service uses WS-Addressing, specify the version.

### WCF Scenarios

The following table shows the scenarios for Web Services that utilize WCF. The WSHttpBinding-based scenarios are divided according to the way the client authenticates itself to the server. For example, if your client presents a user name and a password to the server, choose the **Username (message**

**protection)** scenario. The user interface lets you provide the identity information in the form of a user name or a certificate as required.

| WCF Scenario Name | When to use |
|---|---|
| WSHttpBinding - No Authentication | • Client uses the server's X.509 certificate for encryption<br>• Client is not authenticated<br>• Communication may utilize advanced standards such as secure conversation and MTOM |
| WSHttpBinding - Windows authentication | • Client and server use Windows authentication<br>• Security is based on Kerberos or SPNEGO negotiations<br>• Communication may utilize advanced standards such as secure conversation and MTOM |
| wsHttpBInding - Certificate authentication | • Client uses the server's X.509 certificate for encryption<br>• Client uses its own X.509 certificate for signature<br>• Communication may utilize advanced standards such as secure conversation and MTOM |
| WSHttpBinding - username (message protection) authentication | • Client uses the server's X.509 certificate for encryption<br>• Client is authenticated with a username and password<br>• Communication may utilize advanced standards such as secure conversation and MTOM |
| WSHttpBinding - username (transport protection) authentication | • SSL is enabled<br>• Client is authenticated with a username and password<br>• Communication may utilize advanced standards such as secure conversation and MTOM |
| WSFederationHttpBinding | • Client authenticates against the STS using a predefined scenario<br>• Client uses the token given from the STS to authenticate against the server |
| Custom Binding | • Web Service that uses WS-* standards<br>• WCF services of any configuration |

### Optimization Scenarios

The following table describes the built-in Optimization scenario.

| Scenario Name | When to use |
|---|---|
| MTOM | • MTOM enabled Web services<br>• Web Services which may require you to specify the WS-Addressing version |

For MTOM type scenarios, if your service uses WS-Addressing, specify the version.

## WCF Scenario Settings

This section describes the values required for the WCF security scenarios:

### The WsHttpBinding Scenario

#### No Authentication (Anonymous)

In this scenario, the client uses the server's certificate to encrypt a message; there is no client authentication.

You specify only one of the following settings:

- **Negotiate service credentials.** Negotiate the Web Service's certificate with the server.
- **Specify service certificate.** Browse for a service certificate. For more information, see "Select Certificate Dialog Box" on page 853. If you select this option, the **Negotiate service credentials** option is not available.

Provide the DNS information.

- **Expected server DNS.** The expected identity of the server in terms of its DNS. This can be **localhost**, an IP address, or a server name. It can also be the common name by which the certificate was issued.

#### Windows Authentication

This WCF scenario uses Windows Authentication.

You declare the expected identity of the server in terms of its **SPN** or **UPN** identities. If you are testing a WCF service that has not been customized and uses the default configuration, use this type of scenario.

#### Certificate Authentication

In this WCF WSHttpBinding scenario, the client uses the server's X.509 certificate to encrypt the message and its own certificate for a signature.

Specify only one of the following settings:

- **Negotiate service credentials.** Negotiate the Web Service's certificate with the server.
- **Specify service certificate.** Browse for a service certificate. For details, see "Select Certificate Dialog Box" on page 853. If you select this option, the **Negotiate service credentials** option is not available.

Provide the DNS information:

- **Expected server DNS.** The expected identity of the server in terms of its DNS. This can be **localhost**, an IP address, or a server name. It can also be the common name by which the certificate was issued.

### Username Authentication (Message Protection)

In this WCF WSHttpBinding scenario, the client uses the server's X.509 certificate to encrypt the message, and sends a user name and password to authenticate itself.

Specify the following settings:

- **Username. Password.** The client's user name and password credentials.

Specify only one of the following settings:

- **Negotiate service credentials.** Negotiate the Web Service's certificate with the server.
- **Specify service certificate.** Browse for a service certificate. For details, see "Select Certificate Dialog Box" on page 853. If you select this option, the **Negotiate service credentials** option is not available.

Provide the DNS information:

- **Expected server DNS.** The expected identity of the server in terms of its DNS. This can be **localhost**, an IP address, or a server name. It can also be the common name by which the certificate was issued.

### Username (Transport Protection) Authentication

This WCF WSHttpBinding scenario enables SSL and authenticates the client with a user name and password on the message level.

Specify the following settings:

- **Username. Password.** The client's user name and password credentials.

#### The Federation Scenario

In the **WSFederationHttpBinding** scenario, the client authenticates against the STS (Security Token Service) to obtain a token. The client uses the token to authenticate against the application server.

Therefore, two bindings are needed, one against the STS and another against the application server.

First, use the Security Scenario editor to define an STS binding. For more information, see "Create and Manage Security Scenarios" on page 842. When setting the binding against the application server, specify this file in the **Referenced file** box.

For the Federation scenario, specify the following server information:

- **Transport.** HTTP or HTTPS
- **Encoding.** Text or MTOM

For the Federation scenario, specify the following security information:

- **Authentication mode.** IssuedToken, IssuedTokenForCertificate, IssuedTokenForSslNegotiated, IssuedTokenOverTransport, or SecureConversation

- **Bootstrap policy.** IssuedToken, IssuedTokenForCertificate, IssuedTokenForSslNegotiated, or IssuedTokenOverTransport

For the Federation scenario, specify the following identity information:

- **Server certificate.** Browse for a server certificate. For more information, see the "Select Certificate Dialog Box" on page 853.
- **Expected server DNS.** the expected identity of the server in terms of its DNS. This can be **localhost** or an IP address or server name.

For the Federation scenario, specify the following STS (Security Token Service) information:

- **Issuer address.** The address of the issuer of the STS. This can be **localhost,** an IP address, or a server name.
- **Referenced binding.** The file that references the binding that contacts the STS (Security Token Service)

### The Custom Binding Scenarios

The **Custom Binding** scenario enables the highest degree of customization. Since it is based upon WCF **customBinding**, it allows you to test most WCF services, along with services on other platforms such as Java that use
WS - <*spec_name*> specifications.

Use the **Custom Binding** scenario to configure a custom scenario that does not comply with any of the predefined security scenarios.

For the Custom Binding scenario, specify the following server information:

- **Transport.** HTTP, HTTPS, TCP, or NamedPipe
- **Encoding.** Text, MTOM, or WCF Binary

Specify the following security information:

- **Authentication mode.** None, AnonymousForCertificate, AnonymousForSslNegotiated, CertificateOverTransport, Kerberos, KerberosOverTransport, MutualCertificate, MutualSslNegotiated, SecureConversation, SspiNegotiated, UserNameForCertificate, UserNameForSslNegotiated, UserNameOverTransport, or SspiNegotiatedOverTransport
- **Bootstrap policy.** For SecureConversation type authentication, specify a bootstrap policy: AnonymousForCertificate, AnonymousForSslNegotiated, CertificateOverTransport, Kerberos, KerberosOverTransport, MutualCertificate, MutualSslNegotiated, SspiNegotiated, UserNameForCertificate, UserNameForSslNegotiated, UserNameOverTransport, or SspiNegotiatedOverTransport
- **Net security.** the network security. Select None, Windows stream security, or SSL stream security. For services with HTTP transport, leave the default value, **None**. To enable SSL for HTTP, choose the HTTPS transport.

If your Web Service uses **Reliable messaging**, enable the option, and select **Ordered** or **Not Ordered**.

### Identities

Your security settings may require you to provide identity details for either the client and server, or both of them.

An example of identity details for the client, are user name/password or an **X.509** certificate.

For identity information, provide one or more authentication details as required by the service:

**Username**, **Password**, **Server certificate**, **Client certificate,** or a custom Windows identity. For details about choosing a certificate, see "Select Certificate Dialog Box" on page 853.

Some scenarios require you to declare the expected identity of the server in terms of its DNS, SPN, or UPN identity.

- **DNS.** Provide the name of a server or use localhost.
- **SPN.** Provide the SPN identity in the `domain\machine` format.
- **UPN.** Provide the UPN identity in the `user@domain` format.

After setting the basic values, you can set advanced attributes as described in "Advanced Settings Dialog Box" on page 849.

#### WCF Extensibility

You can implement your own binding, behavior, or channel when using customBinding by defining the `assemblyPath` and `typeName` by modifying the configuration file `<script directory>/WSDL/@config/[your config].stss`.

The `assemblyPath` attribute should have a value of either the full path of the dll or its relative path to script directory.

The `typeName` attribute should have the full type name: `ns.typeName`.

## Binding

Name the scenario attribute in the protocols element and provide the `assemblyPath` and `typeName` attributes.

The class you use for binding is inherited from `System.ServiceModel.Channels.Binding`.

## Channel

Add a new element under the customization node. You can specify any name for the element, however the element must contain the two attributes:`assemblyPath` and`typeName`.

The class to use for binding is inherited from `System.ServiceModel.Channel.BindingElement`.

> **Note:** This will work with customBinding scenarios only.

## Behavior

Add a new element under the behaviors element (which is under `endpointBehavior`) and add the two attributes `assemblyPath` and `typeName`.

To bind the new element, implement the `System.ServiceModel.Description.IEndpointBehavior` class.

> **Note:** If you inherit from `System.ServiceModel.Description.ClientCredentials`, the client credentials from this class will be used.

## Examples of Channel and Behavior

```
<protocols scenario="customBinding" uiType="customBinding"
xmlns="http://mf/ServiceTest/config">
<mode>Private</mode>
<customization>
<textMessageEncoding />
<preferlrhttpTransport />
<myChannel assemblyPath="CustomChannel.dll" typeName="CustomChannel.WCFChannel" />
</customization>
<behaviors>
<endpointBehaviors>
<behavior>
<clientVia viaUri="qwqwq" />
<myBehavior assemblyPath="CustomBehavior.dll" typeName="CustomBehavior.WCFbehavior" />
</behavior>
</endpointBehaviors>
</behaviors>
</protocols>
```

An example of overriding the whole binding (the configuration may contain just one line):

```
<protocols scenario="userBinding" assemblyPath="WCFBinding.dll" typeName="
WCFBinding.Binding"/>
```

## Preparing Security Scenarios for Running

### Parameterizing Security Elements

You can parameterize the security elements in a script independently. For example, in a username-based security scenario, you might want each Vuser or iteration to use a different user name.

### Protecting Custom Headers

When an operation uses SOAP headers, VuGen does not automatically sign or encrypt them. To incorporate a protection scheme such as a signature or encryption, you must manually add the following information to the scenario's configuration file (.stss) in the **behavior** element:

- soapAction of the relevant operation
- The header XML name and namespace
- The protection level

The following example shows an outgoing message with the soapAction, **http://mySoapAction**. The XML element **header1** from namespace **http://myServiceNamespace** is encrypted and signed. The **header2** element from the same namespace is only signed.

```
<protocols ...>
              ...
              <behaviors>
                    <contractBehaviors>
                        <behavior>
                            <channelProtectionBehavior>
                                <protectionRequirements action="http://mySoapAction">
                                    <incomingEncryptionParts>
                                        <header localName="header1"
namespace="http://myServiceNamespace" />
                                    </incomingEncryptionParts>

                                    <incomingSignatureParts>
                                        <header localName="header1" namespace="
http://myServiceNamespace " />
                                        <header localName="header2" namespace="
http://myServiceNamespace " />
                                    </incomingSignatureParts>

                                </protectionRequirements>

                           </channelProtectionBehavior>
                        </behavior>
                    </contractBehaviors>
              </behaviors>
            </protocols>
```

### Emulating Users with Iterations

Many of the security scenarios establish a session with the server. For example, every scenario that uses **WS-SecureConversation** establishes a server session. This session is established when the first operation is executed and ends when the script is finished. By default, VuGen closes all sessions after each iteration and opens them again when the next iteration begins. This implies that every iteration simulates a new session and Vuser.

When working with multiple iterations, this may not be the desired effect—you may prefer to keep the original session active and not open a new session for each iteration. This applies when load testing through the LoadRunner Controller or when setting multiple iterations in the runtime settings.

You can override this behavior so that only the first iteration will establish a new session, while all subsequent ones will continue to use the open session. This simulates a user who repeatedly performs an action using the same session.

To determine which simulation mode to use, choose the one which is most appropriate to what you are simulating. For example, if you are simulating a load test where most of the actions are performed repeatedly by the same user in a single session, use the above configuration. If you are unsure, leave the default settings.

## Add Security to a Web Service Script

This task describes how to add set the security for your Web Service calls. For details about Web Services security, see "Setting Security Overview" on page 824.

1. **Insert a new Web Services Security Step**
   a. Place the cursor at the point at which you want to add the security settings. In most cases, it is best to place it in **vuser_init** so that the security scope will be applied to the whole script. To apply the security for specific calls, place it at the desired location.
   b. Right-click and select **Insert > New Step**. In the Steps Toolbox, under the **Web Services** functions, double-click**web_service_set_security**. The Set Security Properties box opens.
2. **Add a security token - optional**
   a. Click **Security Tokens** and select a token type. For details, see "Security Tokens and Encryption" on page 825.
   b. In the Token Name box, assign a name for the token to be used by VuGen in identifying the token.

      Add any relevant information, such as **User Name** and **Password** for the UserName token type.
3. **Add a message signature or encryption - optional**
   a. On the Set Security Properties box toolbar, click **Add Message Signature** or **Add Message Encryption**.
   b. In the **Signing/Encryption token** box, select a token to use with the message signature or encryption. Both signatures and encryptions require you to specify a token previously defined as the signing/encrypting token. See "Add a security token - optional" above above.

c. Under **What to sign**, specify a target token, or leave the field blank to apply the signature or encryption to the whole message body. For details, see "Security Tokens and Encryption" on page 825.

4. **Cancel the security settings - optional**

   To cancel the security settings at a specific point within the script, add a **web_service_cancel_security** step at the desired point.

## See also:

- "Set Security Properties Dialog Box" on page 845
- "Customize the Security" below
- "Add SAML Security" on page 841

## Customize the Security

This task describes how to how to configure special cases common to Web Service security.

### Reference a token with a SubjectKeyIdentifier - optional

By default, Service Test adds all of the defined X.509 tokens to the SOAP envelope and references them as binary tokens. It is also possible to exclude the tokens from the message and reference them with a SKI (Subject Key Identifier). This is common with tokens that are used for encryption.

1. Add a token as described in the "Add Security to a Web Service Script" on the previous page.
2. In the script, change the value for **Add** to **false**:

```
SECURITY_TOKEN, "Type=X509","LogicalName=myToken", "StoreName=My",
"IDType=SubjectName", "IDValue=CN=myCert", "StoreLocation=CurrentUser",
"Add=False",
```

3. If necessary, set the **useRFC3280** settings as described in "useRFC3280" on page 841 below.

### You can customize the UserName token with a nonce and timestamp.

1. Locate the **web_service_set_security** function in the script.
2. Add the attributes and their values according to this chart:

| Name | Meaning | Possible values |
|---|---|---|
| **IsNonceIncluded** | Include a nonce with the token. | **True** (default) or **False** |

| | | |
|---|---|---|
| **TimestampFormat** | The timestamp format to use with the token. | • **None**. no timestamp<br>• **Full**. a <timestamp> element with <created> and <expired> inner elements<br>• **Created**. (default) only a <created> element |

For example:

```
web_service_set_security(
        SECURITY_TOKEN, "Type=USERNAME","LogicalName=myToken", "UserName=John",
"Password=1234",  "PasswordOptions=SendPlainText", "IsNonceIncluded=true",
"TimestampFormat=Full", "Add=True",
        LAST);
```

### Customize the encryption - optional

You customize encryption by indicating whether to encrypt the whole element or only its content. This is common when encrypting tokens such as a user name. By default, only the content is encrypted. The following steps describe how to encrypt the entire token.

1. Locate the **web_service_set_security** function in the script.
2. Add the **EncryptionType** attribute with the value **Element**.

```
web_service_set_security(
...
ENCRYPTED_DATA, "UseToken=myToken", "TargetToken=myOtherToken",
"EncryptionType=Element",
LAST);
```

3. To return to the default, remove the **EncryptionType** attribute or set it to **Content.**

### Customize WS-Security - optional

To change the algorithm Service Test uses for encryption or to modify some other low-level security details.

1. To change either of these items, open the **%Service Test%/bin/mmdrv.exe.config** file in a text editor.
2. If this file does not contain the **<microsoft.web.services2>** element, add it as shown below.

```
<configuration>
...
  <microsoft.web.services2>
    <security>
      <x509 storeLocation="CurrentUser" allowTestRoot="true" useRFC3280="true" />
```

```
      <binarySecurityTokenManager valueType="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3">
        <sessionKeyAlgorithm name="TripleDES" />
        <keyAlgorithm name="RSA15" />
      </binarySecurityTokenManager>
      </security>
  </microsoft.web.services2>
...
<configuration>
```

3. Set the element values as required:

| Name | Meaning | Possible values |
|---|---|---|
| **verifyTrusy** | Check sent/received x.509 certificate's validity. | • True (default)<br>• False |
| **sessionKeyAlgorithm** | The algorithm the session symmetric key should use to encrypt the message. | • AES128<br>• AES192<br>• AES256<br>• TripleDES |
| **keyAlgorithm** | The algorithm to use by the public key to encrypt the session key. | • RSA15<br>• RSAOAEP |
| **useRFC3280** | Generate subject key identifiers that are interoperable and not Windows specific. | • True<br>• False (default) |

## Add SAML Security

This task describes how to add SAML security for your Web Service calls.

1. Insert a new Web Services Security step.
   a. Place the cursor where you want to add the security settings.
   b. Right-click and select **Insert > New Step**.
   c. In Steps Toolbox > SOAP functions, select **web_service_set_security_saml**.
2. Insert a SAML assertion.

a. Right-click and select **Insert > New Step**.

b. In Steps Toolbox > SOAP functions, select **ws_sign_saml_assertion**.

c. Provide the unsigned assertion, a certificate file, and a password. (Optional)

3. Set the security policy. (Optional)

   Specify a policy file, or leave blank to use the default. Manually enter values override any values in the policy file. Make sure to provide an Issuer URL, also known as the **STS URL**.

4. Cancel the SAML settings. (Optional)

   At the required location in the script, insert a **web_service_cancel_security_saml** step.

### See also:

- "SAML Security Options" on page 827
- For syntax information, see the Function Reference (**Help > Function Reference**)

## Create and Manage Security Scenarios

The following steps describes how to create and customize a security scenario for a specific service.

1. Open the Security Scenario Data dialog box.

   a. Click **Manage Services**. In the left pane, select the service for which you want to set the security scenario. If necessary, import a service, as described in "Import Service Dialog Box" on page 822.

   b. Select the **Protocol and Security** tab and click the **Edit Data** button. The Security Scenario Data dialog box opens.

2. Create or open a security scenario.

   **Create a security scenario**

   a. Select **Private scenario** and select a built-in security scenario for the current service.

   b. In the **Scenario type** box, select a scenario. For details, see "Choosing a Security Model" on page 829.

   c. Specify the required values for your scenario. For details, see "WCF Scenario Settings" on page 832.

   d. To specify a certificate (only applicable to some of the scenarios), click the Browse button adjacent to the **Client certificate** or **Specify service certificate** box to open the Select Certificate dialog box. For details, see the "Select Certificate Dialog Box" on page 853.

   - **To retrieve a certificate from a file:** Select **File** and locate its path.
   - **To retrieve a certificate from the Windows store:** Select **Windows Store**. Then select a store location and name, and specify a search string:
     - To search for all certificates, leave the **Search text** box empty.
     - To search for a specific certificate, specify a substring of the certificate name. If

required, specify a password for the private key.

Click **Find** to generate the list of certificates found in the store.

**Open a security scenario**

- **To use an editable scenario:** Select **Private scenario** and click **Import**. In the Shared Scenario dialog box, select a stored scenario. If required, modify the settings as described in "WCF Scenario Settings" on page 832.
- **To use a non-editable scenario:** Select **Shared Scenario**. Then browse to and select a stored scenario.

> **Note:** If someone modifies a shared scenario file at its source, it will affect your script.

3. Modify the security scenario, as described below. (Optional)

| To: | Do this: |
|---|---|
| Configure advanced settings | For most scenarios, the default proxy, encoding, and other settings are ideal, but you can modify these settings if needed.<br>a. Click **Advanced** to configure the settings. For details, see "Advanced Settings Dialog Box" on page 849.<br>b. Click **OK** to save the security scenario. |
| Modify an existing security scenario | To create and modify security scenarios that will be available globally for all scripts—not just this specific service, use the Security Scenario Editor. You can also use the editor to save the scenario.<br>a. Select **SOA Tools** > **Security Scenario Editor**.<br>b. Click the **Load** button and browse for an existing **stss** scenario file.<br>c. Modify the scenario settings as required<br>d. Click **Save** or **Save as**. |
| Protect SOAP headers | Manually modify the **behavior** element in the scenario's configuration file:<br>a. In VuGen, open the Script view. Choose choose **View > Script View**.<br>b. Click in the script editor and select **Open Script Directory** from the shortcut menu.<br>c. Locate the security scenario's configuration file <*service_name*>**.stss** in **WSDL/@config** folder.<br>d. Modify the behavior section of the file. For details, see "Protecting Custom Headers" on page 837. |

| To: | Do this: |
|---|---|
| Set the iteration mode | Configure your environment to use the same session for all iterations:<br>a. Open the script root folder: In Script view, click inside the script and choose **Open Script Directory** from the shortcut menu.<br>b. Open **default.cfg** file in a text editor.<br>c. In the **[WebServices]** section, add in a row under the toolkit. If you are using the Axis toolkit or if you configured other settings, the file contents may differ.<br>`[WebServices]`<br>`Toolkit=.NET`<br>`SimulateNewUserInNewIteration=0`<br>d. Save and close the file.<br>For details, see "Emulating Users with Iterations" on page 838. |

## Parameterize Security Elements

This task describes how to independently parameterize the security elements in a script.

1. **Open the Security Scenario Editor**

   Select **SOA Tools > Security Scenario Editor**.

2. **Set up a scenario for each Vuser**

   Set up a scenario for each Vuser as described in "Create and Manage Security Scenarios" on page 842. We recommend you use the names **user1**, **user2**, and so forth, and save them in a new folder, **%script root%/WSDL/referencedConfig**.

3. **Open the Parameter List window and create a parameter**

   Select **Vuser > Parameters List**. Create a new parameter, **<ServiceName>_shared_config**. Replace the <ServiceName> with the case-sensitive name of the service you are testing. To determine the exact name of the service, click **Manage Services** to see the list of services.

4. **Add parameter values**

   In the values table, in each row add the file names of the security scenarios with their .stss extensions. You can use a relative path, relative to the script folder. Click **Add Row** to add multiple values. Close the Parameter List dialog box.

5. **Call the parameter**

   a. Click **Manage Services** and select the **Protocol and Security** tab. Click **Edit Data**.

   b. Select **Shared Scenario**. Click the Browse button and enter the parameter name, **<ServiceName>_shared_config**, in the test box.

## Set Security Properties Dialog Box

This dialog box enables you to set the security properties for your Web Service calls.

| | |
|---|---|
| **To access** | In the **Steps Toolbox** pane, double-click the **web_service_set_security** step. |
| **Relevant tasks** | "Add Security to a Web Service Script" on page 838 |
| **Important Information** | If you have edited key algorithm or session algorithm values in the mmdrv.config file for an existing script, these values are replaced with the system default values. |

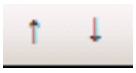
### WS—Security Tab

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| <**Token Grid**> | Displays a unique number, type and name of all tokens that have been added. |

| | |
|---|---|
| **Add Security Token** | Enables you to select a token type:<br>**UserName Token**<br>• **Token Name.** A meaningful name for the token.<br>• **Include nonce.** If selected, an arbitrary number is used once to sign communication.<br>• **Username/Password.** Specify the username/password.<br>• **Password type.** Text, Hash, None<br>• **Timestamp format.** Full, Created, None<br>**X.509 Certificate Token**<br>• **Token Name.** A meaningful name for the token.<br>• **Certificate.** If selected, an arbitrary number is used once to sign communication.<br>• **Reference Type.** BinarySecurityToken, Reference<br>**Kerberos/Kerberos2 Token**<br>• **Token Name.** A meaningful name for the token.<br>• **Host.** The host name of the server against which you want to authenticate. In most cases, it is the host portion of the service URL.<br>• **Domain.** The Windows domain of the server against which you want to authenticate.<br>**PFX File Token**<br>• **Token Name.** A meaningful name for the token.<br>• **PFX File.** Contains all of the token information, including the server certificate, intermediate certificates, and the private key in a single file.<br>**Tip:** If you have a **.jks** file, use it here. VuGen will automatically convert it to a **.p12** file.<br>• **PFX File Password.** The password for your PFX file.<br>**Note:** If you are using a **.jks** file, use your .jks file password. |

| | |
|---|---|
| **Add Message Signature** | • **Signing token.** The token to use for signing, usually an X.509 type. Select from the list of all added tokens.<br>• **Canonicalization algorithm.** A URL for the algorithm to use for canonicalization. A drop-down list provides common algorithms. If you are unsure which value to use, keep the default.<br>• **Transform Algorithm.** A URL for the Transform algorithm to apply to the message signature. A dropdown list provides common algorithms. If you are unsure which value to use, keep the default.<br>• **Inclusive namespace list.** A list of comma-separated prefixes to be treated as inclusive (optional).<br>• **What to sign.** The SOAP elements to sign: SOAP Body, Timestamp, and WS-Addressing.<br>  • **Xpath (optional).** An XPath that specifies which parts in the message to sign. If left blank, the elements selected in the Signature options field are signed. For example, `//*[local-name(.)='Body']`.<br>  • **Token (optional).** The target token you want to sign. Select from the drop-down list of all added tokens. With most services, this field should be left empty. |
| **Add Message Encryption** | • **Encrypting Token.** The token to use for encryption, usually an X.509 type. You can select from a list of all previously created tokens.<br>• **Encrypting Type.** Indicates whether to encrypt the whole destination Element or only its Content.<br>• **Key algorithm.** The algorithm to use for the encryption of the session key: RSA15 or RSAOAEP.<br>  If you have edited the mmdrv.config file with a custom key algorithm value for an existing script, this value is replaced with the system default value of RSA15.<br>• **Session algorithm.** The algorithm to use for the encryption of the SOAP message. You can select from a list of common values.<br>  If you have edited the mmdrv.config file with a custom session algorithm value for an existing script, this value is replaced with the system default of AES128.<br>• **What to encrypt.**<br>  • **Xpath (optional).** An XPath that indicates the parts of the message to encrypt. If left blank, only the SOAP body is encrypted.<br>  • **Token (optional).** The name of the encrypted token. A drop-down box provides a list of all added tokens. With most services, this field should be left empty. |
| | Delete a token definition from the grid. |

| | |
|---|---|
|  | **Up/Down**. Positioning tools that allow you to set the priority of the security elements.<br>**Note:** Make sure the security elements are positioned in order of their priority. |
| **Exclude Timestamp** | Removes the timestamp from the SOAP header before sending the security element to the server. |

### WS Addressing

The WS-Addressing tab indicates whether WS-Addressing is used by the service, and if so, its version number. You can also specify the IP address of the server to which you want the response to be sent.

### See also:

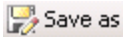
- "Security Tokens and Encryption" on page 825
- "Add SAML Security" on page 841

## Security Scenario Editor Dialog Box

This dialog box enables you to define security scenarios for your script.

| | |
|---|---|
| **To access** | **SOA Tools > Security Scenario Editor** |
| **Important information** | You can also define scenarios for a specific service. For details, see "Create and Manage Security Scenarios" on page 842. |

User interface elements are described below:

| UI Element | Description |
|---|---|
|  | **New.** Resets the editor for defining a new security scenario. If you made changes to the current scenario, it prompts you to save them. |
| | **Load.** Opens an existing shared scenario from a URL or file. |
| | **Save.** Saves the scenario file. If you have not saved the file at least once, it prompts you for a name. |
|  | **Save as.** Saves the scenario file at a new location. |
| | **Help.** Opens the Online help for security scenarios. |
| | **Close.** Closes the dialog box. |

| | |
|---|---|
| Advanced... | Opens the Advanced Setting dialog box for setting the encoding, reliable messaging, secure session information, and proxy configuration.<br>For details, see "Advanced Settings Dialog Box" below. |
| **Scenario type** | The security scenario type: No scenario or a sub-type of Core, Security, WCF, or Optimization scenarios. |

## Advanced Settings Dialog Box

This dialog box lets you configure advanced settings for security scenario in the areas of Encoding, Advanced Standards, Security, or HTTP and Proxy. You access these setting via the "Security Scenario Editor Dialog Box" on the previous page.

Not all settings are relevant for all scenarios—some of them might be disabled or hidden depending on the scenario type.

### Encoding

The Encoding tab lets you indicate the type of encoding to use for the messages: **Text**, **MTOM**, or **Binary**. The default is **Text** encoding.

For each of these encoding methods, you can choose a version of WS-Addressing:

- None
- WSA 1.0
- WSA 04/08

> **Tip:** To instruct a SOAP request to leave out WS-Addressing, add a **web_service_set_option** function with the **ExcludeWseHeaders** flag set to *true* before the **web_service_call** function. For details, see the Function Reference (**Help > Function Reference**).

### Advanced Standards

This tab lets you configure advanced WS- standards, such as Reliable Messaging and the Via address option.

If your service implements the **WS-ReliableMessaging** specification, enable the **Reliable Messaging** option and set the following options:

- **Reliable messaging ordered.** indicates whether the reliable session should be ordered
- **Reliable messaging version.** WSReliableMessagingFebruary2005 or WSReliableMessaging11

### Via Address

In certain instances, you may need to send a message to an intermediate service that submits it to the actual server. This may also apply when you send the message to a debugging proxy. This corresponds to the WCF **clientVia** behavior.

In such cases it may be useful to separate the physical address to which the message is actually sent, from the logical address for which the message is intended. The logical address may be the physical address of the final server or any name. It appears in the SOAP message as follows:

<wsa:Action>http://myLogicalAddress<wsa:Action>

The logical address is retrieved from the user interface. By default, it is the address specified in the WSDL. You can override this address from the Manage Services dialog box.

### Security

The Advanced security settings correspond to the **WS-Security** specifications.

For security scenarios that are based upon WCF WSHttpBinding, you can indicate the following settings:

- **Enable secure session.** Establish a security context using the WS-SecureConversation standard.
- **Negotiate service credentials.** Allow WCF proprietary negotiations to negotiate the service's security.

For **WSHttpBinding, Custom Binding**, or **WSFederationHttpBinding** WCF type scenarios, you can set the default algorithm suite and protection level:

| Attribute | Meaning | Possible Values |
|---|---|---|
| **Default Algorithm Suite** | The algorithm to use for symmetric/ asymmetric encryption.<br>These are the values from the SecurityAlgorithmSuite configuration in WCF: | • Basic128<br>• Basic128Rsa15<br>• Basic128Sha256<br>• Basic128Sha256Rsa15<br>• Basic192<br>• Basic192Rsa15<br>• Basic192Sha256<br>• Basic192Sha256Rsa15<br>• Basic256<br>• Basic256Rsa15<br>• Basic256Sha256<br>• Basic256Sha256Rsa15<br>• TripleDes<br>• TripleDesRsa15<br>• TripleDesSha256<br>• TripleDesSha256Rsa15 |
| **Protection Level** | Should the SOAP Body be encrypted/signed | None, Sign, and EncryptAndSign (default) |

For **Custom Binding** or **WSFederationHttpBinding** WCF type scenarios, you can customize the security settings in greater detail. The following table describes the options and their values:

| Attribute | Meaning | Possible Values |
|---|---|---|
| **Message Protection Order** | The order for signing and encrypting | • SignBeforeEncrypt<br>• SignBeforeEncrypt-AndEncryptSignature<br>• EncryptBeforeSign |
| **Message Security Version** | The WS-Security security version | A list of the current versions |
| **Security Header Layout** | The layout for the message header | • Strict<br>• Lax<br>• LaxTimeStampFirst<br>• LaxTimeStampLast |
| **Key Entropy Mode** | The entropy mode for the security key. | • Client Entropy<br>• Security Entropy<br>• Combined Entropy |

You can enable or disable the following options:

- **Require derived keys.** Indicates whether or not to require derived keys.
- **Require security context cancellation.** Disabling this option implies that stateful security tokens will be used in the **WS-SecureConversation** session (if enabled).
- **Include timestamp.** Includes a timestamp in the header.
- **Allow serialized token on reply.** Enables the reply to send a serialized token.
- **Require signature confirmation.** Instructs the server to send a signature confirmation in the response.

For X.509 certificates, you can specify values for the following items:

| Attribute | Meaning | Possible Values |
|---|---|---|
| **X509 Inclusion Mode** | When to include the X509 certificate | • Always to Recipient<br>• Never<br>• Once<br>• AlwaysToInitiator |
| **X509 Reference Style** | How to reference the certificate | • Internal<br>• External |

| | | |
|---|---|---|
| **X509 require derived keys** | Should X509 certificates require derived keys | • Enable - Yes<br>• Disable - No |
| **X509 key identifier clause type** | The type of clause used to identify the X509 key. | • Any<br>• Thumbprint<br>• IssuerSerial<br>• SubjectKeyIdentifier<br>• RawDataKeyIdentifier |

### HTTP and Proxy

This tab lets you set the HTTP and Proxy information for your test.

#### HTTP(S) Transport

The following table describes the HTTP(S) Transport options:

| Option | Meaning | Possible Values |
|---|---|---|
| **Transfer mode** | The transfer method for requests/responses | Buffered, Streamed, StreamedRequest, StreamedResponse |
| **Max response size (KB)** | The maximum size of the response before being concatenated | Default 65 KB |
| **Allow cookies** | Enable cookies | Enabled/Disabled |
| **Keep-Alive Enabled** | Enable keep-alive connections | Enabled/Disabled |
| **Authentication scheme** | HTTP authentication method | None, Digest, Negotiate, NTLM, IntegratedWindows Authentication, Basic, Anonymous |
| **Realm** | The realm of the authentication scheme | Any URL |
| **Require client certificate** | For SSL transport, require a certificate | Enabled/Disabled |

### Proxy Information

If the Web service's transport uses a proxy server, you can specify its details in the **Security** tab. The following table describes the proxy options:

| Option | Meaning | Possible Values |
|---|---|---|
| **Use default web proxy** | Use machine's default proxy settings | Enabled/Disabled |
| **Bypass proxy on local** | Ignore proxy when the service is on the local machine | Enabled/Disabled |
| **Proxy address** | the proxy server | Any URL |
| **Proxy authentication scheme** | HTTP authentication method on Proxy | None, Digest, Negotiate, NTLM, IntegratedWindows Authentication, Basic, Anonymous |

## Select Certificate Dialog Box

This dialog box enables you to search and locate a certificate from a file or Windows store.

| | |
|---|---|
| **To access** | Select a scenario that uses a certificate in one of the following ways:<br>• Open the Security Scenario Editor: Choose **SOA Tools > Security Scenario Editor**.<br>• In the Manage Services dialog box, select the **Protocol and Security** tab and click the **Edit Data** button.<br>Select a WCF scenario that uses a client or service certificate, such as **WsHttpBinding** or **WSFederationHttpBinding**. In the **Certificate** field, click the Browse button. |
| **Important information** | This only applies to security scenarios that allow you to specify client, server, or service certificates. |
| **Relevant tasks** | • "Add Security to a Web Service Script" on page 838<br>• "Create and Manage Security Scenarios" on page 842. |

### Select Certificate from File

When you choose **File**, the dialog box shows the user interface elements described below:

| UI Element | Description |
|---|---|
| | **Browse.** Allows you to locate the certificate file with a .pem, .arm, .der, or .pfx extension. |
| **File** | The complete path of the certificate file. |

| | |
|---|---|
| **Password (optional)** | The password required to access the certificate. |

### Select Certificate from Windows Store

When you choose **Windows Store**, the dialog box shows the user interface elements described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| Find | Begins the search for the certificate. |
| **Import from** | The location of the certificate:<br>• **Windows store**<br>• **File** |
| **Store location** | The store location, for example **Current User**. |
| **Store name** | The store name, for example, **AuthRoot**. |
| **Search text** | The text to match in the certificate name.<br>By default, the search mechanism searches for the text by the certificate's *subject name*.<br>**Tip:** To search for an X.509 certificate by its *serial number* or *issuer name* , drag a **web_service_set_security** function from the Steps Toolbox, into your script. From the Security Tokens drop-down, select **X509 Certificate Token**. Click the Browse button adjacent to the **Certificate** field. In the Select Certificate dialog box, select *Issuer Name* or *Serial Number* from the **Search field** drop-down list. |
| **Password (optional)** | The password required to access the certificate. |
| **<certificate list>** | A list of the certificates in the Windows store sorted by Subject, Issuer, Private, Store Location, and Store Name. |

## Web Services Security Examples

This section illustrates several common security scenarios.

### Authenticating with a Username Token

The following example illustrates the sending of a message level username/password token (a username token), where the user name is John and the password is 1234.

```
    web_service_set_security(
        SECURITY_TOKEN, "Type=USERNAME","LogicalName=myToken", "UserName=John",
"Password=1234", "PasswordOptions=SendPlainText", "Add=True",
        LAST);
```

### Signing a Specific Element with an X.509 Certificate

It is possible to sign only a specific element in a message. The following example signs a specific element using an XPATH expression:

```
    web_service_set_security(
        SECURITY_TOKEN, "Type=X509","LogicalName=myCert", "StoreName=My",
"IDType=SubjectName", "IDValue=CN=myCert", "StoreLocation=CurrentUser", "Add=True",
        MESSAGE_SIGNATURE, "UseToken=myCert", "TargetPath=//*[local-name
(.)='someElement' and namespace-uri(.)='http://myNamespace']",
        LAST);
```

### Signing with an X.509 Certificate

The following example shows a script using an X.509 certificate for a digital signature.

```
    web_service_set_security(
        SECURITY_TOKEN, "Type=X509","LogicalName=myCert", "StoreName=My",
"IDType=SubjectName", "IDValue=CN=myCert", "StoreLocation=CurrentUser", "Add=True",
        MESSAGE_SIGNATURE, "UseToken=myCert",
        LAST);
```

> **Note:** The certificate needs to be installed in the Windows certificate store. In the example above, you need to set the actual store name, store location, and subject name of your certificate.

### Encrypting with a Certificate

The following sample encrypts a message with the service's X.509 certificate.

```
    web_service_set_security(
        SECURITY_TOKEN, "Type=X509","LogicalName=serviceCert", "StoreName=My",
"IDType=SubjectName", "IDValue=CN=serviceCert", "StoreLocation=CurrentUser",
"Add=False",
        ENCRYPTED_DATA, "UseToken=serviceCert",
        LAST);
```

After you specify the details of your X.509 certificate, you can encrypt a specific XPATH in the message.

Since we want to generate a Subject Key Identifier, we set the Add value to **False**.

### Authenticating with a Username Token and Encrypting with an X.509 Certificate

The following example sends a username token to the service and encrypts it with the server's X.509 certificate:

```
web_service_set_security(
        SECURITY_TOKEN, "Type=X509","LogicalName=serviceCert", "StoreName=My",
"IDType=SubjectName", "IDValue=CN=serviceCert", "StoreLocation=CurrentUser",
"Add=True",
        SECURITY_TOKEN, "Type=USERNAME","LogicalName=myUser", "UserName=John",
"Password=1234", "PasswordOptions=SendPlainText", "Add=True",
        ENCRYPTED_DATA, "UseToken=serviceCert", "TargetToken=myUser",
        LAST);
```

The **UseToken** and **TargetToken** properties indicate which token to use and which to encrypt. Their values reference the **LogicalName** property of the tokens.

### Encrypting and Signing a Message

This example shows how to sign a message using a private key and then encrypt it using the service's public key.

```
web_service_set_security(
        SECURITY_TOKEN, "Type=X509","LogicalName=myCert", "StoreName=My",
"IDType=SubjectName", "IDValue=CN=myCert", "StoreLocation=CurrentUser", "Add=True",
        SECURITY_TOKEN, "Type=X509","LogicalName=serverToken", "StoreName=My",
"IDType=SubjectName", "IDValue=CN=serverCert", "StoreLocation=CurrentUser",
"Add=False",
        MESSAGE_SIGNATURE, "UseToken=myCert",
        ENCRYPTED_DATA, "UseToken=serverCert",
    LAST);
```

### Referencing an X.509 Certificate Using a Hash

In certain cases, you may be unable to reference a certificate with a subject name. This example shows how to reference the certificate using its unique hash.

```
web_service_set_security(
   SECURITY_TOKEN, "Type=X509","LogicalName=serviceCert", "StoreName=My",
   "IDType=Base64KeyID", "IDValue=pOl0+1iuotKLlO91nhjDg5reEw0=",
   "StoreLocation=CurrentUser", "Add=False",
   ENCRYPTED_DATA, "UseToken=serviceCert",
 LAST);
```

# Troubleshooting and Limitations for Web Services

This section describes troubleshooting and limitations for the Web Services protocol.

> **Tip:** For general VuGen troubleshooting and limitations, see "Troubleshooting and Limitations for VuGen" on page 914.

- **Issue:** When you run a **web_service_call** function, the function may fail because a security token is included in the response.

  **Workaround:** Use the **web_service_set_option** function to turn on the **DoNotValidateSecurity** option, using the syntax below:

  ```
  web_service_set_option("DoNotValidateSecurity", "true");
  ```

- If your script contains one of the following API functions: **web_service_call**, **web_service_set_security**, or **web_service_set_security_saml**, you will encounter an error if WSE 2.0 SP3 and WSE 3.0 are not installed.
  **Solution:**
  - First, activate .NET 3.5. For Windows 8.1/2012 R2, turn on the feature as described in the MSDN. For Windows 7/2008 R2, download and install .NET Framework 3.5.
  - Next, install the WSE components from your LoadRunner installation package folders, **lrunner\Common\wse20sp3** and **lrunner\Common\wse30**, or download them from the Internet.
- The **Record default web browser** option in the Recording Wizard, is only supported for Internet Explorer.
- For large SOAP envelopes, Record and Replay snapshots are disabled.
- The Import SOAP feature is not supported for envelopes containing a single element larger than 500KB.
- Recording requests with attachments or security is not supported.
- For Axis toolkit, Web service calls that include both attachments and security are not supported.
- For .NET toolkit, SOAP version 1.2 is not supported for asynchronous calls.
- You can enter text strings up to 10 KB to encode to base 64. If your string is larger, use the **Get from file** option.
- VuGen supports Web Service messages over JMS message Queue, but does not support JMS Topics.
- If the response is MIME format, replay may fail for Web services imported through the .NET toolkit.
  **Workaround:** Insert a **web_service_set_option** function before the **web_service_call**. Enable the **HandleMIMEResponse** attribute:
  `web_service_set_option("HandleMIMEResponse", "true")`
- JMS Bindings Extensions are not supported.

- All services in your script should have the same security scenario. This can be configured via the Protocols and Security tab.
- Asynchronous Web Service calls and custom user handlers are not supported for WCF.
- LoadRunner cannot replay scripts containing the **soa_xml_validate** function.
- When using **Update service**, steps that are already in the script will not display the updated properties (in the step argument view) until you close and reopen the application. After you reopen the app, step arguments are updated. If the script is open when performing "update service", then on the script view arguments, the application throws an exception.
  **Workaround:** Close the script file while running "update service", or reopen the test after running "update service".
- A Web Service script might not open when you import the WDSL with the Axis toolkit.**Workaround:** Import the WDSL with the .NET toolkit.

  If there is a problem recreating the scripts, do the following:

  a. Create a new test.
  b. Import the WSDL using .NET toolkit.
  c. Go to the directory of the new script.
  d. Copy the folder "WSDL" and paste it on the directory of the old script.
  e. In the directory of the old script open the **default.cfg** file.
  f. Under [WebServices] header, change **"Toolkit=Axis"** to **"Toolkit=.NET"** .

# Windows Sockets Protocol

## Recording Windows Sockets - Overview

The Windows Sockets protocol supports applications which communicate over the TCP/IP protocol using a Microsoft WinSock DLL. The WinSock protocol allows you to see the actual data sent and received by the buffers.

The WinSock protocol records functions that relate to the sockets, data buffers, and the Windows Sockets environment. Using VuGen, you record your application's API calls to the Winsock.dll or Wsock32.dll. For example, you could create a script by recording the actions of a telnet application. After creating a Winsock Vuser script, you can view the recorded buffers as raw data or as a snapshot.

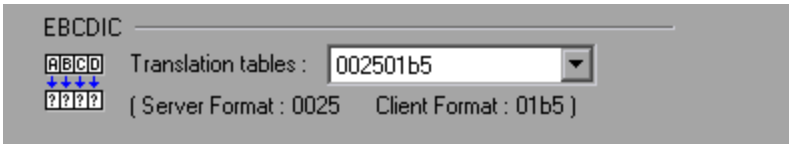
### See also:

- "Windows Sockets Data" on the next page
- "Windows Sockets Snapshots - Overview" on page 860

## Translation Tables

You can display Windows Sockets data in EBCDIC format through a translation table.

A translation table allows you to specify the format for recording when using the WinSock single protocol, and for code generation when using a WinSock multi protocol. This applies to users running on mainframe machines or AS/400 servers. Both the server and client machines determine the format of the data from translation tables installed on your system. If your data is in ASCII format, it does not require translation.




The first four digits of the listbox item represent the server format. The last four digits represent the client format. In the above example, the selected translation table is `002501b5`. The server format is `0025` and the client format is `01b5` indicating a transfer from the server to the client. In a transmission from the client to the server, you would select the item that reverses the formats—`01b50025` indicating that the client's `01b5` format needs to be translated to the server's `0025` format.

The translation tables are located in the **ebcdic** folder under the VuGen's installation folder. If your system uses different translation tables, copy them to the **ebcdic** folder.

For details on selecting a translation table in the recording options, see the "WinSock Recording Options" on page 221.

## Windows Sockets Data

When you use VuGen to create a Windows Sockets Vuser script, your actions are recorded into the three sections of the script: **vuser_init**, **Action**, and **vuser_end**. In addition to the Vuser script, VuGen also creates data files:

- **snapshotdata.ws** contains the data that was transmitted or received during the recording session. VuGen's Snapshot pane displays the contents of the data file. Do not modify the contents of the **snapshotdata.ws** file.
- **data.ws** contains the data that is transmitted during the replay sessions, and is expected to be received. You can right-click any step in the Editor and then select **Show Arguments** to show the buffer content that is stored in **data.ws** for the selected step. Using the **Text View** tab of the dialog box that opens, you can edit the data that is stored for any data buffer.

Several LRS functions, such as **lrs_receive** and **lrs_send**, handle the actual data that is transferred between servers and clients. The data that is received or transmitted is stored in data buffers, which can be very large. In order to simplify the appearance of the Vuser script, the actual data is stored in external files—not in the C file. When a data transfer occurs, the data is copied from the external file into a temporary buffer.

The external file, **data.ws**, contains the contents of all the temporary buffers. The buffers' contents are stored as sequential records. The records are marked by identifiers indicating whether the data was sent or received, and the buffer descriptor. The LRS functions use the buffer descriptors to access the data.

The descriptors have one of the following formats:

```
recv buf index   number of bytes received
send buf index   number of bytes sent
```

The buffer index begins with 0 (zero), and all subsequent buffers are numbered sequentially (1,2,3...) regardless of whether they are send or receive buffers.

In the following example, an **lrs_receive** function was recorded during a Vuser session:

```
lrs_receive("socket1", "buf4", LrsLastArg)
```

In this example, **lrs_receive** handled data that was received on socket1. The data was stored in the fifth receive record(buf4)—note that the index number is zero-based. The corresponding section of the **data.ws** file shows the buffer and its contents.

```
recv buf4 39
    "\xff\xfb\x01\xff\xfb\x03\xff\xfd\x01"
    "\r\n"
    "\r\n"
    "SunOS UNIX (sunny)\r\n"
    "\r"
    "\x0"
    "\r\n"
    "\r"
    "\x0"
```

For task details, see "View and Modify Windows Sockets Buffers" on page 864.

## Windows Sockets Snapshots - Overview

Vuser scripts based on the Windows Sockets Vuser protocol utilize VuGen's Snapshot pane.

For Windows Sockets Vuser scripts, the Snapshot pane displays snapshots of the recorded data buffers. The following sections describe how to view and use the data.

### Displaying Buffer Data

To display a specific buffer in the Snapshot pane:

- In the Editor, select the step that contains a reference to the required buffer.
- In the Step Navigator, double-click the step that contains a reference to the required buffer.

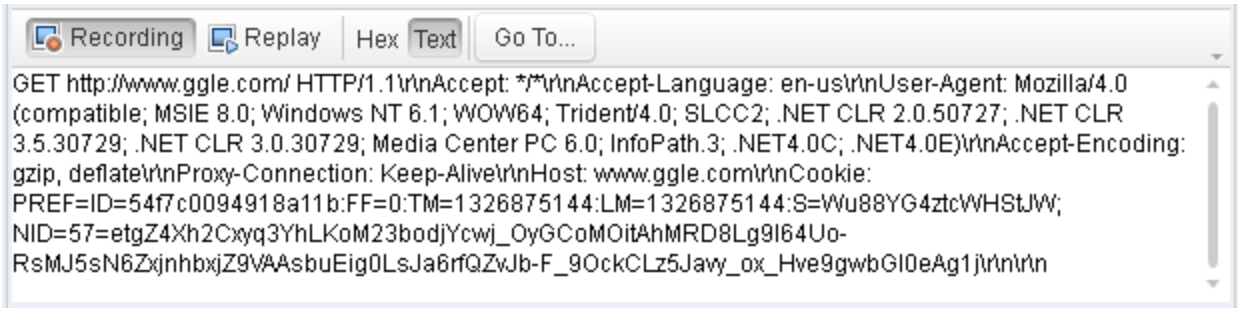
### Buffer Views

You can view the buffer snapshots in either **Text** view or **Hex** view.

> **Note:** VuGen stores the snapshot data as read-only data. You cannot modify the contents of the snapshots. However, you can modify the buffer data that is associated with any of the steps in a Vuser script. To modify the buffer data, right-click the required step in the Editor and then select

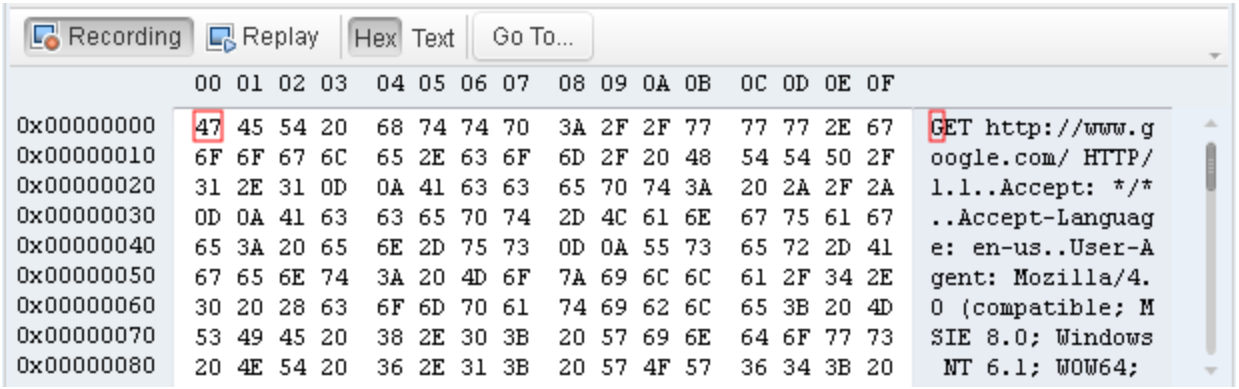
**Show Arguments**. The Text View tab of the dialog box that opens lets you modify the buffer data. For details, see "View and Modify Windows Sockets Buffers" on page 864.

The **Text** view shows the buffer data as text.




The **Hex** view shows the buffer data in hexadecimal representation. The data is displayed in three columns:

- The left column shows the offset of the first character in each row.
- The middle column shows the hexadecimal value of the data.
- The right column shows the data in ASCII format.




## Status Bar

The status bar below the buffer snapshot displays information about the buffer and the data selected in the buffer:

- **Buffer number.** The buffer number of the displayed buffer.
- **Buffer size.** The total number of bytes in the buffer.
- **Buffer type.** The type of buffer—received or sent.
- **Data.** The value of the data that is selected in the buffer, in decimal and hexadecimal formats. Both big endian and little endian sequences are displayed.
- **Offset.** The offset of the selected data from the beginning of the buffer. If you select multiple bytes, the offset displays the range of the selection.

| | | | |
|---|---|---|---|
| buf5: 587 bytes(s) received | BE: 1129324658 (0x43502072) | LE: 1914720323 (0x72205043) | Selection: from 235 (0xEB) to 239 (0xEF) |

### Navigating within the Buffer Data

- To go to a specific offset within the buffer (absolute), click **Go To**. In the Go To Offset dialog box enter an offset value, and then click **Apply**.
- To jump to a location relative to the selected entry, click **Go To**. In the Go To Offset dialog box, click **Advance by** , specify the number of bytes to advance, and then click **Apply**.

### Creating Correlations from the Buffer Data

#### Creating Correlations from the Buffer Data

Using the shortcut menu, you can create correlations directly from the **Recording** tab in the Snapshot pane.

- To create a regular correlation, select the data and choose **Create Correlation** from the shortcut menu. VuGen adds an **lrs_save_param** function to the script.
- To create a boundary bounded correlation, select the data and choose **Create Boundary Correlation** from the shortcut menu. VuGen adds an **lrs_save_searched_string** function to the script.

## Data Navigation Tools

VuGen provides you with **Go To** functionality to help you to navigate through the data in the Snapshot pane. This helps you to identify and analyze a specific values in the snapshot. You can move around within the data buffer by specifying an offset. You can indicate the absolute location of the data, or a location relative to the current position of the cursor within the buffer. You can also select a range of data, by specifying the starting and end offsets. For details on the dialog box options, see "Go To Offset Dialog Box" on page 867.

## Buffer Data Editing

You can perform all of the standard edit operations on buffer data: copy, paste, and delete. To perform edit operations on buffer data, right-click the required step in the Editor and select **Show Arguments**. You can then perform the edit operation in the **Text View** tab of the dialog box that opens. You cannot perform any edit operations in the Binary View tab.

> **Note:** You perform edit operations on buffer data only, not on Snapshot data - which is read-only.

When you copy data from a buffer, VuGen allows you to copy the data either in hexadecimal format or in decimal format. When you insert data into a buffer, VuGen allows you to specify the format of the data—single byte, 2-byte, or 4-byte.

# Record a Windows Sockets Script

This task describes optional steps that you can perform when recording using Windows Sockets.

For details on starting a recording session, see "Record a Vuser Script" on page 158.

## Before you start recording:

1. **Open the recording options.**

   After creating a WinSock script, select **Record > Recording Options** and click the **WinSock** node.

2. **Select a translation table.**

   In the **EBCDIC section**, select a translation table. If your data is in ASCII format, select the **None** option. Otherwise VuGen will convert the ASCII data. For details, see "Translation Tables" on page 858.

3. **Exclude any non-relevant sockets.**

   In the **Exclude Settings** section, add any non-relevant sockets to the list. You should exclude hosts and ports that do not influence the server load under test, similar to the local host and the DNS port (53), which are excluded by default.

   To exclude the entries from the recording, but include them in the log, clear the **Do not include excluded sockets in log** option.

   For user interface details, see the "WinSock Recording Options" on page 221.

4. **Set a think time threshold.**

   Indicate a think time threshold. If VuGen detects a pause in action less than the threshold time, it will not generate a **Think Time** step/ **lr_think_time** function. For details, see the "WinSock Recording Options" on page 221.

## After recording and saving the script:

1. **Parameterize the script.**

   Replace recorded values with parameters using the shortcut menu. For more information, see "Parameterization" on page 357.

2. **Regenerate the script.**

   If you need to regenerate the script, for example if you want to include an excluded host:port, or if the translation was not correct:

   - Select **Record > Regenerate Script**.
   - Click the **Options** link.

- Under **General**, select **Protocols**, and then under **Active Protocols**, ensure that the **Windows Sockets** check box is selected.
- Under **Sockets**, select **Winsock**, and then modify the settings.

> **Note:** Options for script regeneration are available for multi-protocol scripts only.

## View and Modify Windows Sockets Buffers

The following steps describe how to view, modify, and navigate through WinSock buffer data.

### Modifying buffer data

You can modify buffer data in the Show Arguments dialog box for specific **lrs** steps in a Vuser script. You can use the Show Arguments dialog box to modify buffer data for the following steps:

- lrs_length_receive
- lrs_length_send
- lrs_receive
- lrs_receive_ex
- lrs_send

For further details on these steps, see the Function Reference (**Help > Function Reference**).

To display the Show Arguments dialog box for any of the above steps, right-click a step in the Editor and select **Show Arguments**. A dialog box opens and displays the buffer data in the **Buffer Content** section of the dialog box.

For further details about editing buffer data in Windows Sockets steps, see "Buffer Data Editing" on page 862.

> **Note:** You cannot modify any data in the Snapshot pane

### View and modify the data in the data.ws file

In the Solution Explorer, double-click the **data.ws** file. The contents of the data.ws file appear in the VuGen Editor. Modify the data directly in the Editor. For details, see "Windows Sockets Data" on page 859.

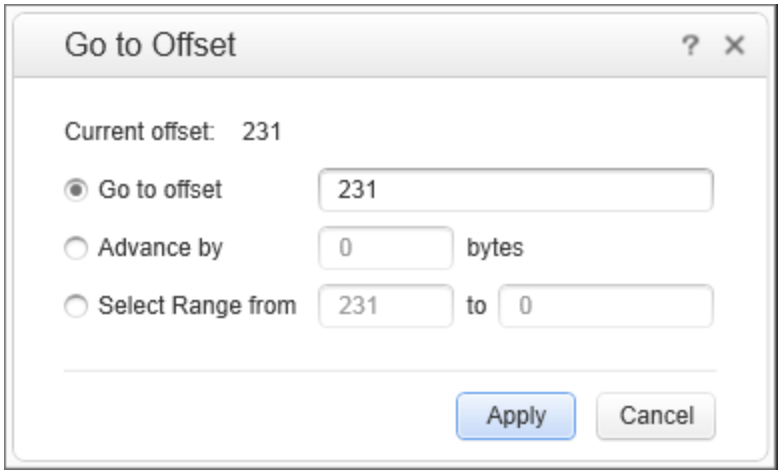
> **Note:** Although it is possible to modify data.ws files, it is recommended that you do not modify these files.

## View the data in the Snapshot pane

1. Ensure that the Snapshot pane is displayed.
2. Open the Vuser script in the Editor and select a step, or double-click an entry in the Step Navigator. The associated snapshot is displayed in the Snapshot pane. You cannot edit the snapshot data.

## Navigate within the snapshot data

To navigate within the buffer data, display in the Snapshot pane, and then click **Go to**. The Go to Offset dialog box opens.




- To go to a specific offset within the buffer (absolute), select **Go to offset** and specify an offset value. Click **Apply**.
- To jump to a location relative to the cursor, click **Advance by** and specify the number of bytes to advance. To advance ahead, enter a positive value. To move backwards within the buffer, use a negative value. Click **Apply**.
- To select a range of data within the buffer, click **Select range from** and specify the beginning and end offsets. Click **Apply**.

## Insert data into a buffer

You can insert a numerical value into a data buffer. You can insert the data as a single, double-byte, or 4-byte value. The following steps describe how to insert a number into a data buffer.

1. Copy the numerical data to be inserted to the clipboard.
2. Right-click a step in the Editor and select **Show Arguments**.
3. In the dialog box that opens, ensure that the **Text View** tab is displayed.
4. Under **Buffer Content**, right-click at the location in the buffer where you want to insert the data, and then select **Advanced > Paste Byte** or **Advanced > Paste Short (2 bytes)** or **Advanced > Paste Long (4 bytes)**.
5. Click **OK**. VuGen inserts the data into the buffer.

### Copy and paste blocks of data

You can modify the buffer data as characters, decimal numbers, or hexadecimal numbers. For details, see "Buffer Data Editing" on page 862.

> **Note:** You can edit buffer data only when you view the step arguments. You cannot edit buffer data in the Snapshot pane.

1. Right-click a step in the Editor and select **Show Arguments**. A dialog box opens and displays the buffer data in the **Buffer Content** section of the dialog box.
2. To copy buffer data:
   - As characters, select one or more bytes and press Ctrl+c.
   - As a decimal number, select one or more bytes, right-click in the selection and select **Advanced > Copy As Decimal Number**.
   - As a hexadecimal number, select one or more bytes, right-click in the selection and select **Advanced > Copy As Hexadecimal Number**.
3. To paste the data:
   - As characters, press Ctrl+V.
   - As a single byte (assuming the size of the data on the clipboard is a single byte), right-click at the desired location in the buffer and click **Advanced > Paste Byte**.
   - In short format (2-byte), right-click at the desired location in the buffer and click **Advanced > Paste Short (2 bytes)**.
   - In long format (4-byte), right-click at the desired location in the buffer and click **Advanced > Paste Long (4 bytes)**.
4. To delete data, select the data in the Text view, right-click inside the selection and select **Delete**.

## Data Buffers

When you use VuGen to create a Windows Sockets Vuser script, VuGen creates the **data.ws** data file. This file contains the data that is transmitted, and is expected to be received, during the replay sessions. You can right-click any step in the Editor and then select **Show Arguments** to show the buffer content that is stored in **data.ws** for the selected step. Using the **Text View** tab of the dialog box that opens, you can edit the data that is stored for any data buffer.

The **data.ws** data file has the following format:

- File header
- A list of buffers and their contents

The file header includes an internal version number of the data file format. The current version is 2. If you try to access data from a data file with format version 1, VuGen issues an error.

```
;WSRData 2 1
```

An identifier precedes each record, indicating whether the data was received or sent, followed by the buffer descriptor, and the number of bytes received (for **lrs_receive** only). The buffer descriptor contains a number identifying the buffer.

For example,

```
recv buf5   25
```

indicates that the buffer contains data that was received. The record number is 5, indicating that this receive operation was the sixth data transfer (the index is zero based), and twenty-five bytes of data were received.

If your data is in ASCII format, the descriptor is followed by the actual ASCII data that was transferred by the sockets.

If your data is in EBCDIC format, it must be translated through a look-up table. For information on setting the translation table, see the "WinSock Recording Options" on page 221. The EBCDIC whose ASCII value (after translation) is printable, is displayed as an ASCII character. If the ASCII value corresponds to a non-printable character, then VuGen displays the original EBCDIC value.

```
recv buf6 39
    "\xff\xfb\x01\xff\xfb\x03\xff\xfd\x01"
    "\r\n"
    "SunOS UNIX (sunny)\r\n"
```

The following segment shows the header, descriptors, and data in a typical data file:

**Example:**

```
;WSRData 2 1
send buf0 48
    "\xff\xfd\x01\xff\xfd\x03\xff\xfb\x03\xff\xfb\x18"
recv buf1 15
    "\xff\xfd\x18\xff\xfd\x1f\xff\xfd"
    "#"
    "\xff\xfd"
    "'"
    "\xff\xfd"
    "$"
send buf2 24
    "\xff\xfb\x18"
```

## Go To Offset Dialog Box

This dialog box allows you to go to a specific location within the recorded data.

| | |
|---|---|
| **To access** | On the Snapshot pane toolbar, click **Go to**. |
| **Relevant tasks** | "View and Modify Windows Sockets Buffers" on page 864 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Current offset** | The current offset of the cursor (read only). |
| **Go to offset** | Goes to a specific, absolute offset within the data. |
| **Advance by...bytes** | Jumps to a location relative to the cursor, by a number of bytes. Positive values indicate a forward direction. Negative values indicate a reverse direction. |
| **Select range from...to...** | Selects a range of data within the buffer. |
| **Apply** | Moves the cursor to the specified offset. |

# Customize Your Scripts

This section provides you with tools to customize your scripts, such as how to manually program scripts and how to replay them on Linux machines.

### What do you want to do?

- "Create a PCAP File" below
- "Manually Program a Script using the VuGen Editor" on page 871
- "Create Scripts in External IDEs" on page 877
- "Use DLLs and Customize VuGen" on page 882
- "Create and Run Scripts in Linux" on page 886
- "Program with the XML API" on page 889
- "Share Software Content Resources" on page 906

### See also:

- "Troubleshooting and Limitations for VuGen" on page 914
- "Non-English Language Support" on page 899
- "Additional Components" on page 1566

# Create a PCAP File

PCAP (Packet Capture) files consist of network packet data, created by capturing live network activity through capture tools such as Wireshark. The generated .pcap file can be used for packet sniffing and analyzing network activity. VuGen can parse **.pcap** files and convert them to a Vuser script.

The primary uses for the .pcap file are:

- **Web Services scripts.** For details, see "Create a Script by Analyzing Traffic (Web Services)" on page 775.
- **Mobile Applications scripts.** For details, see "Create a Vuser Script by Analyzing a Captured Traffic File" on page 723.

This task describes how to create a **.pcap** capture file of network or application traffic to use in the preparation of a Vuser script.

## Create a capture file

#### Windows

Create a capture file containing a log of all TCP traffic over the network on a Windows platform. Use a downloadable capture tool such as *Wireshark*. Make sure to save the Wireshark capture file in the *tcpdump* format, as this is the format supported by VuGen.

> **Note:** For PCAP analysis, VuGen supports Wireshark versions Wireshark 2.2.x (up to and including version 2.2.7).

### Android

Create a capture file using *tPacketCapture* on Android devices, or a similar application.

### Linux Redhat

1. If **tcpdump** is not installed, install it using the Linux package installation command.
2. To save captured traffic in a file, use **tcpdump -w xxxx.pcap**. For details about using tcpdump, such as interface selection, and file size settings, see the tcpdump man page.

## Tips for creating .pcap files

- To generate a smaller, more manageable script, try to capture the network traffic only for the time that you perform actions in your application.
- When using external tools, make sure that all packet data is being captured and none of it is being truncated.
- For command line capture utilities, make sure to provide all of the required arguments.

# Troubleshooting missing packets

**Issue:** Your script is missing steps you recorded into a capture file.

You encounter the following warning in the **Output Pane> Code generation** tab:

```
Warning: One or more responses are missing or have missing packets. Therefore, a step
may appear to be missing in the script.
This issue can be caused if the recording was stopped before all the responses were
received.
If the script is generated from a .pcap file, check if the file has missing packets.
```

This error may be caused by unnecessary network activity on the recorded machine, which can cause the capturing application to drop packets.

**Steps to Resolve:** Ensure that the capturing machine has no unnecessary network traffic in the background.

> **Workaround for Mobile Applications - HTTP/HTML scripts:** You can circumvent this issue using the Recording options. Select **Recording Options > HTTP Properties > Advanced > Generate steps with missing responses** to generate steps for HTTP requests that are missing server responses.

# Manually Program a Script using the VuGen Editor

## Manually Programming Scripts - Overview

VuGen allows you to program your own functions into the script, instead of recording an actual session. You can use the Vuser API or standard programming functions. Vuser API functions allow you to gather information about Vusers. For example, you can use Vuser functions to measure server performance, control server load, add debugging code, or retrieve runtime information about the Vusers participating in the test or monitoring.

This chapter describes how to program a Vuser script from within the VuGen editor, incorporating your application's libraries or classes.

You can also develop a Vuser script through programming within the Visual C and Visual Basic environments. In these environments, you develop your Vuser script within your development application, while importing the Vuser API function libraries. For more information, see "Create Scripts in External IDEs" on page 877.

To create a customized script, you first create a skeleton script. The skeleton script contains the three primary sections of a script: **init**, **actions**, and **end**. These sections are empty and you manually insert functions into them.

You can create empty scripts for the C and Java programming languages.

> **Tip:** Make sure to define all variables at the beginning of the action, before all other API functions. This will prevent compilation errors.

## Programming Vuser Actions

The Vuser script files, *test*.c, *test*.usr, and *test*.cfg, can be customized for your Vuser.

You program the actual Vuser actions into the *test.c* file. This file has the required structure for a programmed Vuser script. The Vuser script contains three sections: vuser_init, Actions, and vuser_end.

Note that the template defines extern C for users of C++. This definition is required for all C++ users, to make sure that none of the exported functions are modified inadvertently.

**Example:**

```
#include "lrun.h"
#if defined(__cplusplus) || defined(cplusplus) extern "C"
{
#endif
int LR_FUNC vuser_init(LR_PARAM p)
{
lr_message("vuser_init done\n");
```

```
return 0;
}
 int Actions(LR_PARAM p)
{
lr_message("Actions done\n");
return 0;
}
 int vuser_end(LR_PARAM p)
{
lr_message("vuser_end done\n");
return 0;
}
#if defined(__cplusplus) || defined(cplusplus)}
#endif
```

You program Vuser actions directly into the empty script, before the **lr_message** function of each section.

The *vuser_init* section is executed first, during initialization. In this section, include the connection information and the logon procedure. The *vuser_init* section is only performed once each time you run the script.

The *Actions* section is executed after the initialization. In this section, include the actual operations performed by the Vuser. You can set up the Vuser to repeat the Actions section (in the *test*.cfg file).

The *vuser_end* section is executed last, after the all of the Vuser's actions. In this section, include the clean-up and logoff procedures. The *vuser_end* section is only performed once each time you run the script.

**Note:** Load Generators control Vusers by sending SIGHUP, SIGUSR1, and SIGUSR2 Linux signals. Do not use these signals in your Vuser scripts.

## Create a Template

VuGen includes a utility that copies a template into your working folder. The utility is called mkdbtest, and is located in $M_LROOT/bin. You run the utility by typing:

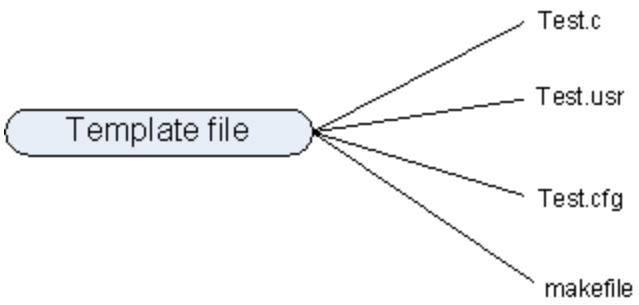
```
mkdbtest  name
```

When you run mkdbtest, it creates a folder called name, which contains the template file, name.c. For example, if you type:

```
mkdbtest test1
```

*mkdbtest* creates a folder called *test1*, which contains the template script, *test1.c*.

When you run the *mkdbtest* utility, a folder is created containing four files test.c, *test.usr*, *test.cfg* and *Makefile*, where *test* is the test name you specified for `mkdbtest`.




## Define Transaction and Insert Rendezvous Points Manually

When programming a Vuser script without VuGen, you must manually configure the Vuser file in order to enable transactions and rendezvous. The configuration settings are listed in the test.usr file.

```
[General]
Type=any
DefaultCfg=Test.cfg
BinVuser=libtest.libsuffix
RunType=Binary
[Actions]
vuser_init=
Actions=
vuser_end=
[Transactions]
transaction1=
[Rendezvous]
Meeting=
```

Each transaction and rendezvous must be defined in the *usr* file. Add the transaction name to the Transactions section (followed by an "="). Add each rendezvous name to the Rendezvous section (followed by an "="). If the sections are not present, add them to the *usr* file as shown above.

## C Vuser Scripts

In a C Vuser script, you can use any C code that conforms to the standard ANSI conventions. To create an empty C Vuser script, select **C Vuser** in the Create a New Script dialog box. VuGen creates an empty C Vuser script:

```
Action1()
{
    return 0;
}
```

You can use C Vuser functions in all of Vuser script types that use C functions.

See the Function Reference (**Help > Function Reference**) for a C reference that includes syntax and examples of commonly used C functions.

### Guidelines for Using C Functions

All standard ANSI-C conventions apply to C Vuser scripts, including control flow and syntax. You can add comments and conditional statements to the script just as you do in other C programs. You declare and define variables using ANSI C conventions.

The C interpreter that is used to run Vuser scripts accepts the standard ANSI C language. It does not support any Microsoft extensions to ANSI C.

Before you add any C functions to a Vuser script, note the following limitations:

- A Vuser script cannot pass the address of one of its functions as a callback to a library function.
- The **stdargs**, **longjmp**, and **alloca** functions are not supported in Vuser scripts.
- Vuser scripts do not support structure or union arguments or return types. Pointers to structures are supported.
- In Vuser scripts, string literals are read-only. Any attempt to write to a string literal generates an access violation.
- C Functions that do not return int, must be casted. For example,
  ```
  extern char * strtok();
  ```

### Calling libc Functions

In a Vuser script, you can call **libc** functions. However, since the interpreter that is used to run Vuser scripts does not support any Microsoft extensions to ANSI C, you cannot use Microsoft's include files. You can either write your own prototypes, or ask Customer Support to send you ANSI-compatible include files containing prototypes for **libc** functions.

### Linking Mode

The C interpreter that is used to run Vuser scripts uses a "lazy" linking mode in the sense that a function need not be defined at the start of a run, as long as the function is defined before it is used. For example:

```
lr_load_dll("mydll.dll");
    myfun();  /* defined in mydll.dll -- can be called directly,
                 immediately after mydll.dll is loaded. */
```

## Java Vusers

In Java Vuser scripts, you can place any standard Java code. To create an empty Java Vuser script, select **Java Vuser** in the New Virtual User dialog box. VuGen creates an empty Java script:

```
import lrapi.lr;
public class Actions
{
    public int init() {
```

```
        return 0;
    }
    public int action() {
        return 0;
    }
    public int end() {
        return 0;
    }
}
```

> **Note:** For Java type Vusers, you can only edit the **Actions** class. Within the Actions class, there are three methods: **init**, **action**, and **end**. Place initialization code in the **init** method, business processes in the **action** method, and cleanup code in the **end** method.

## .NET Vusers

You can create an empty .NET Vuser script, in which to place .NETcode. This script type lets you incorporate your existing .NET application into VuGen. To create an empty .NET Vuser script, select **.NET** in the Create a New Script dialog box.

In a .NET Vuser script the default language is C#. If your script is generated from a recorded session, VuGen enables you to change the script language from C# to VB.NET by selecting Visual Basic .NET Language from **Record > Recording Options > General > Script** and regenerating the script.

> **Tip:** You can edit the script in Visual Studio by clicking the button.

The following example shows the Action section of an empty .NET script:

```
namespace Script
 {
  public partial class VuserClass
  {
    public int Action()
  {
  // Add your code here
  return 0;
  }
 }
}
```

> **Note:** Enter the business process code in the **Action** method. Add initialization code to the **vuser_init** method, and cleanup code to the **vuser_end** method.

# Troubleshooting and Limitations - Programming

## Framework 4.5 for .NET scripts

**Issue:** Running a .NET script recorded in VuGen, fails if you run it in Visual Studio—cannot find associated DLL.

**Steps to Resolve:** If the script was recorded with a .NET 4.5 AUT, rebuild the script with Framework version 4.5 in Visual Studio.

## Framework 3.5 for .NET scripts

**Issue:** .NET DLLs created in Visual Studio using Framework 3.5 may not run.

**Steps to Resolve:** Add the following to the **<app>.config file**: (if there is no such file, create one)

```
<configuration>
<startup>
    <supportedRuntime version="v4.0" />
   </startup>
</configuration>
```

## Missing references for scripts created in Visual Studio

**Issue:** A compilation error occurs (such as error CS0246 or warning MSB3644) when compiling scripts created in Visual Studio, possibly because the referenced assemblies were not added correctly to Script.csproj.

**Steps to Resolve:**

- If Visual Studio is installed:
  a. Open the script solution file in Visual Studio.
  b. In the Solution Explorer, click **References** and select **Add Reference** from the right-click menu.
  c. Save the solution, close Visual Studio, and reopen the script in VuGen.

- If Visual Studio is not installed:
  a. Open the script.csproj file in a text editor.
  b. In the **ItemGroup** element, add an Include statement for your reference as follows:

```
<Reference Include="ICSharpCode.SharpZipLib">
<HintPath>C:\xxxx\ICSharpCode.SharpZipLib.dll</HintPath>
</Reference>
```

  c. Save the file, close the editor, and reopen the script in VuGen.

# Create Scripts in External IDEs

## Creating Vuser Scripts or Unit Tests in Visual Studio or Eclipse

The following sections describe how to develop a Vuser script or unit test through programming within the Visual Studio or Eclipse environments.

LoadRunner provides add-ins that allow you to develop scripts with supported versions of Visual Studio or Eclipse.

> **Note:** For details on supported Eclipse versions, see the System Requirements.

There are two types of add-ins:

- Visual Studio/Eclipse IDE add-in
- Visual Studio/Eclipse IDE add-in for Developers

The basic **IDE add-in** allows you to create a Vuser script within the Visual Studio or Eclipse environment. You program a standard Vuser script using standard C# functions or protocol-specific functions from the Function Reference (**Help > Function Reference**), within your native environment. You can then use the script in your testing environment, for example, in a LoadRunner scenario.

The **IDE add-in for Developers** allows you to create *and run* a unit test directly from within Visual Studio or Eclipse. The run mechanism is VuGen's **mdrv** process. This add-in adds a **Devops** menu to the Visual Studio or Eclipse interface. You can configure runtime settings directly from your development environment and run the test. The saved tests, NUnit (Visual Studio) or JUnit (Eclipse), can be referenced directly from your testing environment, for example, a LoadRunner scenario.

These add-ins are provided in **Additional Components** folder of your LoadRunner installation package. Be sure to select the correct Visual Studio add-in for your version of Visual Studio.

Once you install the basic **IDE add-in**, you can create a new VuGen script within Visual Studio. Alternatively, you can begin developing your script in VuGen. If, while developing a script in VuGen, you realize that you need to the capabilities of your native environment, the **Open in Visual Studio** or **Open in Eclipse** button opens the script in the respective application. This requires you to have first installed the basic **Visual Studio IDE add-in**. (For Eclipse, VuGen automatically installs the add-in the first time you choose **Open in Eclipse**). For details, see "Debugging .NET Vuser Scripts" on page 618 or "Edit and Run Scripts in Eclipse" on page 564.

When working in Visual Studio or Eclipse, the complete VuGen API is available from the Object browser. For information about each of the Vuser functions that you can use when programming your script, see the Function Reference (**Help > Function Reference**).

# Create a Vuser Script in Visual Studio

LoadRunner's basic IDE add-ins for Visual Studio let you create a Vuser script in Visual Studio in VB, C++, or C#.

Although .NET-based and Java protocols support creating threads, we recommend that you do not use background threads in real load testing scenarios because:

- Threads can degrade tests scalability.
- Threads can affect performance measurements.
- The utility functions' behavior is undetermined if called from any thread except the Vuser main thread which runs the vuser_init, Action, and vuser_end actions. This applies to all functions named lr*.

**To create a Vuser script in Visual Studio:**

1. Install the IDE add-in for your version of Microsoft Visual Studio from your LoadRunner installation package's **Additional Components** folder. For example, **Additional Components\IDE Add-Ins\LRVS<*version*>IDEAddInSetup.exe**.
2. In Visual Studio, select the appropriate template from the Installed Templates **LoadRunner VB|C++|C# .NET Vuser**. Visual Studio creates a new project with one class and a template for a Vuser, and the script file, **<name>.usr**. The template contains three sections, **Initialize**, **Actions**, and **Terminate**.

   The following example shows a Visual C# template:

```
public int Initialize()
      {
          // TO DO: Add virtual user's initialization routines
            return lr.PASS;
      }
      public int Actions()
      {
          // TO DO: Add virtual user's business process actions
          return lr.PASS;
      }
      public int Terminate()
      {
          // TO DO: Add virtual user's termination routines
          return lr.PASS;
      }
```

3. Add code to the template, in the TODO sections.
4. Open the Object Browser (**View** menu). Expand the LoadRunner node (for example **Interop.LoadRunner**) to see the LoadRunner elements. Add the desired elements to your script, such as transactions, rendezvous points, and messages.
5. Expand the Toolbar menu, **Vuser**, and enhance your script with runtime settings and parameters.

For more information, see the runtime settings **General > Run Logic** or the "Parameter List Dialog Box" on page 389.

6. Use the Vuser menu to replay the script and test its functionality.
7. Select **Vuser > Create Load Scenario**, to create a LoadRunner scenario using this .usr file.
8. You can also build the LoadRunner project as a DLL file, which will be saved in the same folder as the project. You can reference this DLL directly from a LoadRunner scenario.

## Create a Vuser Script in Eclipse

LoadRunner's basic IDE add-in for Eclipse, lets you create a Java Vuser script in Eclipse. You can begin in VuGen and then open Eclipse. Alternatively, you can work from start to finish in Eclipse.

> **Note:** For details on supported Eclipse versions, see the System Requirements.

### Prerequisite

Make sure you have JDK 1.7 (JRE 7) on your machine. Go to **java.com** to check your version or download the required version.

After you install it:

1. Open Eclipse and select **Window > Preferences**.
2. Navigate to the **Java > Installed JREs** node. If **jre7** is not in the **Installed JREs** list, click **Add** and use the wizard to add its folder (for example c:\Program Files\Java\jre7).
3. In the Installed JREs list, click the check box by **jre7** to instruct Eclipse to use this version.
4. Close Eclipse.

### Method 1: Create a script in VuGen, and develop it further in Eclipse

1. Open VuGen and create a new Java type script.
2. Record or add steps as you normally would.
3. Click the Open in Eclipse button on the toolbar. If this is the first time you are opening Eclipse from VuGen, a message box prompts you to enter the Eclipse location. VuGen automatically copies the required files in Eclipse's **dropins** folder. Eclipse opens with your current script in the **Package Explorer** pane.

### Method 2: Create the script in Eclipse

1. Manually copy the **hp.lr.vugeneclipse42addin.jar** file from your LoadRunner installation package's **Additional Components\IDE Add-Ins\EclipseAddin** folder into the Eclipse **dropins** folder. Extract the files from the jar file.
2. Open Eclipse. Select **File > New > Project** and expand the **LoadRunner Script** node. You can create any of the Java protocol scripts: Java over HTTP, Java Record Replay, or Java Vuser.

> **Tip:** To verify that the add-in installed correctly, open the Eclipse Installation Details dialog box (Help > About > Installation Details.)

### Develop the script in Eclipse

1. Expand the script's node and select the **default package > Action.java** node. Code the script as you normally would in the Eclipse editor, in the appropriate sections.
2. In the script's node in the Package Explorer, expand **Referenced Libraries > classes > lrapi >** to access the desired LoadRunner elements, such as transactions, rendezvous points, and messages.
3. Expand the **Vuser** menu (this may require you to select the parent script name in the Package Explorer) and enhance your script with runtime settings and parameters. For more information, see the runtime setting **General > Run Logic** or "Parameter List Dialog Box" on page 389.
4. Save and run the project. Select **Vuser > Run Vuser** to test the script. Then select **Vuser > Create Load Scenario** to run it from the Controller. Note that you will not be able to open this script in VuGen once you edit it in Eclipse.

## Develop a Unit Test Using Visual Studio (NUnit test)

The LoadRunner Add-in for Developers lets you create an NUnit test in Visual Studio for use with LoadRunner.

> **Note:** To run your NUnit test from Controller, make sure that the NUnit framework is installed on the Controller and Load Generator machines.

**To create an NUnit test in Visual Studio:**

1. Install the IDE for Dev add-in for your Microsoft Visual Studio version from your LoadRunner installation package's **..\Additional Components\IDE Add-ins Dev** folder. For example, **..\Additional Components\IDE Add-Ins Dev\LRVS<*version*>IDEAddInDevSetup.exe**.
2. In Visual Studio, open your unit test. This test should comply with the following guidelines:
   - It is a class library
   - There is a reference to the NUnit library, nunit.framework.dll, **using nunit.framework**. This must have the same NUnit framework version as the one installed on the Controller and Load Generator machines.
   - At least one of the classes in the project should be a TextFixture (using the [TextFixture] annotation)
3. In the code, instantiate the LoadRunner API function.

   > **Example:** private LoadRunner.LrApi lr = new LoadRunner.LrApi();

4. Select **DevOps Vuser > Add LoadRunner API Reference** to add protocol-specific or general API

functions to your test. Alternatively, select **Add LoadRunner API Reference** from the context menu. Add LoadRunner functionality, such as transactions, think time, messaging, and so forth.

5. Build the LoadRunner project as a DLL file, which will be saved in the same folder as the project.
6. Select **DevOps Vuser > Run Vuser** to run the test with the LoadRunner engine. In the Visual Studio Output window, select **Show output from: LoadRunner Information** to view the runtime data.
7. (Optional) Add the DLL as a unit test to an existing or new LoadRunner scenario. For details, see "New Scenario Dialog Box" on page 969.

## Develop a Unit Test Using Eclipse (JUnit or Selenium test)

The LoadRunner Eclipse Add-in for Developers lets you create a JUnit test in using Eclipse.

> **Note:** For supported Eclipse versions, see System Requirements.

**To create a unit test in Eclipse:**

1. Make sure you have JDK 1.7 (JRE 7) on your machine. Go to **java.com** to check your version or download the required version. After you install it, open Eclipse and select **Window > Preferences**. Navigate to the **Java > Installed JREs** node. If **jre7** is not in the **Installed JREs** list, click **Add** and use the wizard to add its folder (for example c:\Program Files\Java\jre7). In the Installed JREs list, click the check box by **jre7** to instruct Eclipse to use this version.
2. Run the Eclipse Dev add-in, LREclipseIDEAddInDevSetup.exe, from your LoadRunner installation package's **Additional Components\IDE Add-Ins Dev** folder. After installing the Eclipse add-in, rebuild the plugin cache by running the following command line string: **Eclipse.exe -clean**.
3. In Eclipse, open your Selenium or JUnit test.
4. Code the test as you normally would in Eclipse.
5. Build your java classes.
6. Select **Devops Vuser > Add LoadRunner API Reference** to add the desired LoadRunner functions to your script as well as transactions, rendezvous points, and messages.
7. Expand the **Devops Vuser** menu and enhance the test with runtime settings and parameters. For more information, see the runtime setting **General > Run Logic** or the "Parameter List Dialog Box" on page 389.
8. Select **Devops Vuser > Run Vuser** to run the test from within Eclipse to verify its functionality.
9. Use the **Devops Vuser** menu to launch the LoadRunner Controller, or add the test to a Controller scenario that is already open.
10. Add the class file at any time as a unit test, to a LoadRunner scenario. For details, see "New Scenario Dialog Box" on page 969.

> **Note:** If you are running multiple instances of Eclipse and you want to use add-in for each instance, you must manually install the Eclipse Add-in for Developers for each instance. Locate the **hp.lr.continuousdelivery.eclipse42addin.jar** file in the <LoadRunner root>\bin

folder, and copy it to the **dropins** folder for each Eclipse instance.

# Use DLLs and Customize VuGen

## Calling Functions from External DLLs

You can call functions that are defined in external DLLs. By calling external functions from your script, you can reduce the memory footprint of your script and the overall runtime.

To call the external function, you load the DLL in which the function is defined.

You can load a DLL in one of the following ways:

- Locally (for one script) by using the **lr_load_dll** function. For task details, see "Load a DLL Locally" below.
- Globally (for all scripts) by adding statements to the **vugen.dat** file. For task details, see "Load a DLL Globally" on the next page.

## Load a DLL Locally

This task describes how to use the **lr_load_dll** function to load a DLL into your Vuser script. Once the DLL is loaded, you can call any function defined within the DLL without having to declare it in your script.

**To load a DLL locally:**

1. In a C Vuser script, add an **lr_load_dll** function to load the DLL at the beginning of your script. Place the statement at the beginning of the *vuser_init* section. **lr_load_dll** replaces the **ci_load_dll** function.

   Use the following syntax:

   ```
   lr_load_dll( library_name);
   ```

   Note that for Linux platforms, DLLs are known as shared libraries. The extension of the libraries is platform dependent.

2. Call the function defined in the DLL in the appropriate place within your script.

   In the following example, the **insert_vals** function, defined in **orac1.dll**, is called, after the creation of the **Test_1** table.

   **Example:** int LR_FUNC Actions(LR_PARAM p) { lr_load_dll("*orac1*.dll");

```
lrd_stmt(Csr1, "create table Test_1 (name char(15), id integer)\n", -1,
        1 /*Deferred*/, 1 /*Dflt Ora Ver*/, 0); lrd_exec(Csr1, 0, 0, 0, 0, 0);


/* Call the insert_vals function to insert values into the table. */
insert_vals();
 lrd_stmt(Csr1, "select * from Test_1\n", -1, 1 /*Deferred*/, 1 /*Dflt Ora Ver*/, 0); lrd_bind_col
(Csr1, 1, =;NAME_D11, 0, 0); lrd_bind_col(Csr1, 2, =;ID_D12, 0, 0); lrd_exec(Csr1, 0, 0, 0, 0,
0); lrd_fetch(Csr1, -4, 15, 0, PrintRow14, 0); ...
```

**Note:** You can specify a full path for the DLL. If you do not specify a path, **lr_load_library** searches for the DLL using the standard sequence used by the C++ function, LoadLibrary on Windows platforms. On Linux platforms you can set the **LD_LIBRARY_PATH** environment variable (or the platform equivalent). The **lr_load_dll** function uses the same search rules as **dlopen**. For more information, see the Unix help 'man' page for **dlopen**.

## Load a DLL Globally

This task describes how to load a DLL globally, to make its functions available to all your Vuser scripts. Once the DLL is loaded, you can call any function defined within the DLL, without having to declare it in your script.

**To globally load DLLs:**

1. Add a list of the DLLs you want to load to the appropriate section of the *mdrv.dat* file, located in your application's *dat* folder.

   Use the following syntax:

   ```
   PLATFORM_DLLS=my_dll1.dll, my_dll2.dll, ...
   ```

   replacing the word *PLATFORM* with your specific platform. For a list of platforms, see the beginning section of the *mdrv.dat* file.

   For example, to load DLLs for Winsock Vusers on an NT platform, include the following section in the mdrv.dat file:

   **Example:**
   ```
   [WinSock]

   ExtPriorityType=protocol

   WINNT_EXT_LIBS=wsrun32.dll
   ```

```
WIN95_EXT_LIBS=wsrun32.dll

LINUX_EXT_LIBS=liblrs.so

SOLARIS_EXT_LIBS=liblrs.so

HPUX_EXT_LIBS=liblrs.sl

AIX_EXT_LIBS=liblrs.so

LibCfgFunc=winsock_exten_conf

UtilityExt=lrun_api ExtMessageQueue=0

ExtCmdLineOverwrite=-WinInet No

ExtCmdLineConc=-UsingWinInet No

WINNT_DLLS=user_dll1 .dll, user_dll2 .dll, ...
```

2. Call the function defined in the DLL in the appropriate place within your script.

## VuGen File and Library Locations

The VuGen .dat files contain the location information of the script's files, as well as the library files for specific protocols.

There are two .dat files, residing in the **M_LROOT\dat** folder used by VuGen: **mdrv.dat** and **vugen.dat**.

### mdrv.dat

The mdrv.dat file contains a separate section for each protocol defining the location of the library files and driver executables.

For information about how to add a custom protocol, see "Protocol SDK" on page 912.

### vugen.dat

The vugen.dat file contains general information about VuGen, used by VuGen and the Controller.

```
[Templates]
RelativeDirectory=template
```

The Templates section indicates where the templates are for the VuGen protocols. The default entry indicates that they are in the relative *template* folder. Each protocol has a subfolder under *template*, which contains the template files for that protocol.

The next section is the **GlobalFiles** section.

```
[GlobalFiles]
main.c=main.c
@@TestName@@.usr=test.usr
default.cfg=test.cfg
default.usp=test.usp
```

The GlobalFiles section contains a list of files that VuGen copies to the test folder whenever you create a new test. For example, if you have a test called "user1", then VuGen will copy main.c, user1.usr and user1.cfg to the test folder.

The ActionFiles section contains the name of the file containing the Actions to be performed by the Vuser and upon which to perform iterations.

```
[ActionFiles]
@@actionFile@@=action.c
```

In addition to the settings shown above, *vugen.dat* contains settings that indicate the operating system and other compilation related settings.

## Storing Runtime Settings in External Files

Vuser behavior refers to the items that you can set in the runtime settings, such as wait times, pacing times, looping iterations, and logging.

Since VuGen creates the Vuser script and the Vuser behavior as two independent sources, you can configure user behavior without directly referencing the Vuser script. This feature lets you make configuration changes to a Vuser and store several profiles for the same Vuser script.

VuGen stores the behavior settings in the default **Vuser.cfg** file. You can save several versions of this file for different user behavior and then run the Vuser script referencing the relevant *.cfg* file.

By default, you cannot control the behavior file from VuGen. VuGen automatically uses the *.cfg* file with the same name as the script.

To call a specific configuration file, run the Vuser from the command line and add the following string:

```
-cfg c:\tmp\<MyCustomConfigFile>.cfg
```

For information on command line parameters, see "Command Line Parameters" below.

> **Note:** The Linux utility, *run_db_vuser*, does not support this option.

## Command Line Parameters

The Vusers can accept command line parameters when invoked.

To send command line parameters to a Vuser from within VuGen, add the attributes and their values in the runtime settings. For details, see the **General > Additional Attributes** runtime settings view.

To control a Vuser from the Windows command line, type **mdrv** at a command prompt from the installation's **bin** folder, with the desired commands. You can also add custom user parameters, after all the other driver parameters. For example:

```
mdrv.exe -usr c:\tmp\Vuser\Vuser.usr    -out c:\tmp\vuser     command_line_params
```

There are several Vuser API functions available to reference them (such as **lr_get_attrib_double**, and so forth). For details, see the Function Reference (**Help > Function Reference**).

> **Note:** The Linux utility, *run_db_vuser*, does not support some of the standard Windows command line options. For details, see "Run a Vuser Script from a Linux Command Line" on the next page.

# Create and Run Scripts in Linux

## Creating and Running Scripts in Linux - Overview

You can use VuGen on a Linux environment in the following ways:

- You can use VuGen to create Vuser scripts that run on Linux platforms. You record your application in a Windows environment and run it in Linux—recording is not supported on Linux.

> **Note:** VuGen provides a tool to check the compatibility of your script to run on Linux-based load generators. For details, see "Check Linux Compatibility" on page 297.

- Users working in Linux-only environments can program Vuser scripts. Scripts can be programmed in C or C++ and they must be compiled into a dynamic library.

To create a script through programming, you can use a Vuser template as a basis for a larger Vuser scripts. The template provides:

- Correct program structure
- Vuser API calls
- Source code and makefiles for creating a dynamic library

## Compile Scripts Manually on Linux

After you modify the template, you compile it with the appropriate *Makefile* in the script's folder. The compiler creates a dynamic library called **libtest.so**.

You can modify the *Makefile* and assign additional compiler flags and libraries by modifying the appropriate sections.

If you are working with a general template, you must include your application's libraries and header files. For example, if your application uses a library called *testlib*, include it in the LIBS section.

```
LIBS         = \
    -testlib \
    -lLrun50 \
    -lm
```

After you modify the *Makefile*, type `make` from the command line in the working folder to create the dynamic library files for the Vuser script.

After you create a script, you check its functionality from the command line. Check that your script communicates with the server and performs all the required tasks. For details, see "Run a Vuser Script from a Linux Command Line" below.

## Run a Vuser Script from a Linux Command Line

When using VuGen to develop Linux-based Vusers, you must check that the recorded script runs on the Linux platform. This task describes how to perform this check and run a Vuser script from a Linux command.

1. **Verify that the script replays in VuGen and is compatible for Linux**

   It is recommended to check the script in VuGen, before attempting to run it in Linux, because it is easier to edit and debug the script in VuGen.

   - Replay the script in VuGen to verify that the script works in Windows. For details, see "Replay a Vuser Script" on page 296.
   - Run the Linux compatibility tool to check the compatibility of your script to run on Linux-based load generators. For details, see "Check Linux Compatibility" on page 297.

2. **Copy the script files to the Linux server**

   Transfer the script files to the Linux server.

3. **Check the Vuser setup on the Linux machine by using verify_generator.**

   If you intend to run all of the Vusers on one host, type:

   ```
   verify_generator
   ```

   The verify_generator either returns **OK** when the setting is correct, or **Failed** and a suggestion on how to correct the setup.

   For detailed information about the verify checks type:

   ```
   verify_generator [-v]
   ```

   The verify utility checks the local host for its communication parameters and its compatibility with all types of Vusers. It checks the following items in the Vuser environment:

   - at least 128 file descriptors

- proper **.rhost** permissions: **-rw-r--r--**
- the host can be contacted using rsh to the host. If not, checks for the host name in **.rhosts**
- **M_LROOT** is defined
- .**cshrc** defines the correct **M_LROOT**
- .**cshrc** exists in the home directory
- the current user is the owner of the **.cshrc**
- a VuGen installation exists in **$M_LROOT**
- the executables have executable permissions
- **PATH** contains **$M_LROOT/bin**, and **/usr/bin**

4. **Run the script**

Run the script in standalone mode from the Vuser script folder, using the **run_db_vuser** shell script:

```
run_db_vuser.sh <commands> script_name.usr
```

The *run_db_vuser* shell script has the following command line options:

| Command | Description |
|---|---|
| **--help** | Display the available options. (This option must be preceded by two dashes.) |
| **-cpp_only** | Run cpp only (pre-processing) on the script. |
| **-cci_only** | Run cci only (pre-compiling) on the script to create a file with a .ci extension. You can run cci only after a successful cpp. |
| **-driver *driver_path*** | Use a specific external driver. Each database has its own driver located in the /bin folder. |
| **-exec_only** | Execute the Vuser .ci file. This option is available only when a valid .ci file exists. |
| **-ci *ci_file_name*** | Execute a specific .ci file. |
| **-out *output_path*** | Place the results in a specific folder. |

By default, *run_db_vuser.sh* runs **cpp**, **cci**, and **execute** in verbose mode. It uses the driver in the VuGen installation\bin folder, and saves the results to an output file in the Vuser script folder. You must always specify a *.usr* file. If you are not in the script folder, specify the full path of the *.usr* file.

For example, the following command line executes a Vuser script called test1, and places the output file in a folder called results1. The results folder must be an existing folder—it will not be created automatically:

```
run_db_vuser.sh-out /u/joe/results1  test1.usr
```

# Program with the XML API

## Programming with the XML API Overview

VuGen's support for XML allows you to dynamically work with XML code and retrieve the values during test execution. Follow these steps in creating an effective XML script:

- Record a script in the desired protocol, usually Web, Web Services, or Wireless.
- Copy the XML structures into your script.
- Add XML functions from the LR API in order to retrieve dynamic data and the XML element values.

The LR API uses XPath, the XML Path language to manipulate the text in an XML document.

You can instruct VuGen to display the output values of XML elements in the Execution log window using the runtime settings. VuGen displays the line numbers, the number of matches, and the value. To allow the displaying of values, you need to enable parameter substitution. In the runtime settings, open the **General:Log** node, select **Extended log**, and select **Parameter Substitution**. For more information, see "Runtime Settings" on page 301.

All Vuser API XML functions return the number of matches found.

## Using XML Functions

This section provides examples of how to work with data in an XML tree. Certain functions allow you to retrieve information, and others let you write information to an XML tree. The examples use the following XML tree containing the names and extensions of several employees in the Acme organization.

```
<acme_org>
     <accounting_dept>
          <employee type='PT'>
               <name>John Smith</name>
               <extension>2145</extension>
          </employee>
     </accounting_dept>
     <engineering_dept>
          <employee type='PT'>
               <name>Sue Jones</name>
               <extension>2375</extension>
          </employee>
     </engineering_dept>
</acme_org>
```

## Reading Information from an XML Tree

The following functions read information from an XML tree:

| | |
|---|---|
| **lr_xml_extract** | Extracts XML string fragments from an XML string. |
| **lr_xml_find** | Performs a query on an XML string. |
| **lr_xml_get_values** | Retrieves values of XML elements found by a query. |

To retrieve a specific value through a query, you specify the tags of the parent and child nodes in a path format. For information about how to retrieve multiple values, see "Multiple Query Matching " on page 892

For example, to retrieve an employee name in the Accounting department, use the following string:

```
lr_xml_get_values("XML={XML_Input_Param}",
"ValueParam=OutputParam",
"Query=/acme_org/accounting_dept/employee/name",
LAST);
```

The Execution log window (with Extended logging enabled) shows the output of this function:

```
Output:
Action.c(20): "lr_xml_get_values" was successful, 1 match processed
Action.c(25): Query result = John Smith
```

## Writing to an XML Tree

The following functions write values to an XML tree:

| | |
|---|---|
| **lr_xml_delete** | Deletes fragments from an XML string. |
| **lr_xml_insert** | Inserts a new XML fragment into an XML string. |
| **lr_xml_replace** | Replaces fragments of an XML string. |
| **lr_xml_set_values** | Sets the values of XML elements found by a query. |
| **lr_xml_transform** | Applies Extensible Stylesheet Language (XSL) transformation to XML data. |

The most common *writing* function is **lr_xml_set_values** which sets the values of specified elements in an XML string. The following example uses **lr_xml_set_values** to change the phone extensions of two *employee* elements in an XML string.

First, we save the XML string to a parameter called *XML_Input_Param*. We want two values to be matched and substituted, so we prepare two new parameters, *ExtensionParam_1* and *ExtensionParam_2*, and set their values to two new phone extensions, 1111 and 2222.

**lr_xml_set_values** contains the argument "ValueName=ExtensionParam", which picks up the values of ExtensionParam_1 and ExtensionParam_2. The current extensions of the two employees are substituted with the values of these parameters, 1111 and 2222. The value of OutputParam is then evaluated proving that the new phone extensions were in fact substituted.

```
Action() {

     int i, NumOfValues;
     char buf[64];

     lr_save_string(xml_input, "XML_Input_Param"); // Save input as parameter
     lr_save_string("1111", "ExtensionParam_1");
     lr_save_string("2222", "ExtensionParam_2");

     lr_xml_set_values("XML={XML_Input_Param}",
          "ResultParam=NewXmlParam", "ValueParam=ExtensionParam",
          "SelectAll=yes", "Query=//extension", LAST);

     NumOfValues= lr_xml_get_values("XML={NewXmlParam}",
          "ValueParam=OutputParam", "Query=//extension",
          "SelectAll=yes", LAST);

     for (i = 0; i < NumOfValues; i++) {/* Print the multiple values of MultiParam */

          sprintf(buf, "Retrieved value %d : {OutputParam_%d}", i+1, i+1);
          lr_output_message(lr_eval_string(buf));
     }

     return 0;
}
Output:
Action.c(40): Retrieved value 1: 1111
Action.c(40): Retrieved value 2: 2222
```

## Specifying XML Function Parameters

Most XML API functions require that you specify the **XML element** and a **query**. You can also indicate if you want to retrieve all results or a single one.

## Defining the XML Element

For defining the XML element to query, you can specify a literal string of the XML element, or a parameter that contains the XML. The following example shows the XML input string defined as a literal string,

"XML=<employee>JohnSmith</employee>"

Alternatively, the **XML** string can be a parameter containing the XML data. For example:

```
"XML={EmployeeNameParam}"
```

## Querying an XML Tree

Suppose you want to find a value within an XML tag, for example, an employee's extension. You formulate a query for the desired value. The query indicates the location of the element and which element you want to retrieve or set. The path that you specify limits the scope of the search to a specific tag. You can also search for all elements of a specific type under all nodes below the root.

For a specific path, use `"Query=/`*full_xml_path_name*`/`*element_name*`"`

For the same element name under all nodes, use `"Query=//`*element_name*`"`

In the VuGen implementation of XML functions, the scope of a query is the entire XML tree. The tree information is sent to the Vuser API functions as the value of the *xml* argument.

## Multiple Query Matching

When you perform a query on an XML element, by default VuGen returns only the first match. To retrieve multiple values from a query, you specify the "`SelectAll=yes`" attribute within your functions. VuGen adds a suffix of *_index* to indicate multiple parameters. For example, if you defined a parameter by the name *EmployeeName*, VuGen creates *EmployeeName_1*, *EmployeeName_2*, *EmployeeName_3*, and so on.

```
lr_xml_set_values("XML={XML_Input_Param}",
"ResultParam=NewXmlParam", "ValueParam=ExtensionParam",
"SelectAll=yes", "Query=//extension", LAST);
```

With functions that *write* to a parameter, the values written to the parameter can then be evaluated. For example, the following code retrieves and prints multiple matches of a query:

```
NumOfValues = lr_xml_get_values("Xml={XmlParam}", "Query=//name",
          "SelectAll=yes", "ValueParam=EmployeeName", LAST);
```

For functions that *read* from parameters, the values of the parameters must be pre-defined. The parameter must also use the convention *ParamName_IndexNumber*, for example *Param*_1, *Param*_2, *Param*_3, and so on. This collection of parameters is also known as a parameter set.

In the following example, lr_xml_set_values reads values from the parameter set and then uses those values in the XPath query. The parameter set that represents the employee extensions, is called ExtensionParam. It has two members: ExtensionParam_1 and ExtensionParam_2. The **lr_xml_set_values** function queries the XML input string and sets the value of the first match to 1111 and the second match to 2222.

```
lr_save_string("1111", "ExtensionParam_1");
lr_save_string("2222", "ExtensionParam_2");

lr_xml_set_values("XML={XML_Input_Param}",
    "ResultParam=NewXmlParam", "ValueParam=ExtensionParam",
    "SelectAll=yes", "Query=//extension", LAST);
```

## XML Attributes

VuGen contains support for attributes. You can use a simple expression to manipulate attributes of XML elements and nodes, just as you can manipulate the elements themselves. You can modify the desired attribute or only attributes with specific values.

In the following example, **lr_xml_delete** deletes the first cubicle element with the name attribute.

**Example:**

```
lr_xml_delete("Xml={ParamXml}",
               "Query=//cubicle/@name",
                "ResultParam=Result",
                LAST
               );
```

In the next example, **lr_xml_delete** deletes the first cubicle element with a name attribute that is equal to Paul.

**Example:**

```
lr_xml_delete("Xml={ParamXml}",
         "Query=//cubicle/@name="Paul",
        "ResultParam=Result",
         LAST
        );
```

## Structuring XML Scripts

Initially, you create a new script in your preferred protocol. You can record a session in that protocol, or you may program the entire script without recording. Structure the Actions section of the script as follows:

- XML input declaration

- The Actions section

The XML input section contains the XML tree that you want to use as an input variable. You define the XML tree as a char type variable. For example:

```
char *xml_input=
"<acme_org>"
    "<employee>"
        " <name>John Smith</name>"
        "<cubicle>227</cubicle>"
        "<extension>2145</extension>"
    "</employee>"
    "<employee>"
        "<name>Sue Jones</name>"
        "<cubicle>227</cubicle>"
        "<extension>2375</extension>"
    "</employee>"
"</acme_org>";
```

The Action section contains the evaluation of the variables and queries for the element values. In the following example, the XML input string is evaluated using **lr_save_string**. The input variable is queried for employee names and extensions.

```
Action() {

    /* Save the input as a parameter.*/
    lr_save_string(xml_input, "XML_Input_Param");
    /* Query 1 - Retrieve an employee name from the specified element.*/
    lr_xml_get_values("XML={XML_Input_Param}",
        "ValueParam=OutputParam",
        "Query=/acme_org/employee/name", LAST);

    /* Query 2 - Retrieve an extension under any path below the root.*/
    lr_xml_get_values("XML={XML_Input_Param}",
        "ValueParam=OutputParam",
        "Query=//extension", LAST);

    return 0;
}
```

## Enhancing a Recorded Session with XML

You can prepare an XML script by recording a session and then manually adding the relevant XML and Vuser API functions.

The following example illustrates how a recorded session was enhanced with Vuser API functions. Note that the only function that was recorded was **web_submit_data**, which appears in bold.

The first section contains the XML input declaration of the variable SOAPTemplate, for a SOAP message:

```
#include "as_web.h"
// SOAP message
const char* pSoapTemplate=
  "<soap:Envelope xmlns:soap=\"http://schemas.xmlsoap.org/soap/envelope/\">"
      "<soap:Body>"
          "<SendMail xmlns=\"urn:EmailIPortTypeInft-IEmailService\"/>"
      "</soap:Body>"
  "</soap:Envelope>";
```

The following section represents the actions of the user:

**Example:**

```
Action1()
{
// get response body
 web_reg_save_param("ParamXml", "LB=", "RB=", "Search=body", LAST);
// fetch weather by HTTP GET
web_submit_data("GetWeather","Action=http://glkev.net.innerhost.com/glkev_ws/
     WeatherFetcher.asmx/GetWeather",
      "Method=GET",
      "EncType=",
      "RecContentType=text/xml",
    "Referer=http://glkev.net.innerhost.com/glkev_
ws/WeatherFetcher.asmx?op=GetWeather",
        "Snapshot=t2.inf",
        "Mode=HTTP",
         ITEMDATA,
        "Name=zipCode", "Value=10010", ENDITEM,
         LAST);


// Get City value
  lr_xml_get_values("Xml={ParamXml}",
                "Query=City",
                "ValueParam=ParamCity",
                LAST
                  );
 lr_output_message(lr_eval_string("***** City = {ParamCity} *****"));


//Get State value
 lr_xml_get_values("Xml={ParamXml}",
                "Query=State",
                "ValueParam=ParamState",
                LAST
```

```
                    );
 lr_output_message(lr_eval_string("***** State ={ParamState}*****"));
```

```
// Get several values at once by using template
lr_xml_get_values_ex("Xml={ParamXml}",

"Template="
            "<Weather>"
               "<Time>{ParamTime}</Time>"
               "<Temperature>{ParamTemp}</Temperature>"
               "<Humidity>{ParamHumid}</Humidity>"
               "<Conditions>{ParamCond}</Conditions>"
            "</Weather>",
               LAST
               );


   lr_output_message(lr_eval_string("***** Time = {ParamTime},
                Temperature =  {ParamTemp}, "
                   "Humidity = {ParamHumid},
                Conditions = {ParamCond} *****"));


 //Generate readable forecast
    lr_save_string(lr_eval_string("\r\n\r\n*** Weather Forecast for {ParamCity},
             {ParamState} ***\r\n"
          "\tTime: {ParamTime}\r\n"
          "\tTemperature: {ParamTemp} deg. Fahrenheit\r\n"
          "\tHumidity: {ParamHumid}\r\n"
          "\t{ParamCond} conditions expected\r\n"
          "\r\n"),
          "ParamForecast"
    );
  // Save soap template into parameter
    lr_save_string(pSoapTemplate, "ParamSoap");

// Insert request body into SOAP template
    lr_xml_insert("Xml={ParamSoap}",
         "ResultParam=ParamRequest",
         "Query=Body/SendMail",
         "position=child",
         "XmlFragment="
             "<FromAddress>john1@mydomain.com</FromAddress>"
             "<ToAddress>softwaresupport@mydomain.com</ToAddress>"
```

```
            "<ASubject>Weather Forecast</ASubject>"
            "<MsgBody/>",
 LAST);
"<soap:Envelope
xmlns:soap=\"http://schemas.xmlsoap.org/soap/envelope/\">"  "<soap:Body>""<SendMai
l xmlns=\"urn:EmailIPortTypeInft-
IEmailService\"/>""<FromAddress>john1@mydomain.com</FromAddress>"
         "<ToAddress>softwaresupport@mydomain.com</ToAddress>"
         "<ASubject>Weather Forecast</ASubject>"
         "<MsgBody/>"
      "</SendMail>"
   "</soap:Body>""</soap:Envelope>";

// Insert actual forecast text
    lr_xml_set_values("Xml={ParamRequest}",
                      "ResultParam=ParamRequest",
                      "Query=Body/SendMail/MsgBody",
                      "ValueParam=ParamForecast",
                       LAST);

// Add header for SOAP
 web_add_header("SOAPAction", "urn:EmailIPortTypeInft-IEmailService");

// Get response body
 web_reg_save_param("ParamXml", "LB=", "RB=", "Search=body", LAST);

// Send forecast to recipient, using SOAP request
 web_custom_request("web_custom_request",
   "URL=http://webservices.matlus.com/scripts/emailwebservice.dll/soap
        /IEmailservice",
        "Method=POST",
        "TargetFrame=",
        "Resource=0",
        "Referer=",
        "Body={ParamRequest}",
        LAST);

// Verify that mail was sent
    lr_xml_find("Xml={ParamXml}",
                "Query=Body/SendMailResponse/return",
                "Value=0",
                 LAST);
    return 0;
}
```

# Use Result Parameters

Some of the **lr_xml** functions return a result parameter, such as **ResultParam**. This parameter contains the resulting XML data after the function is executed. The result parameters will be available from the parameter list in the Select or Create Parameter dialog box.

For example, for **lr_xml_insert**, ResultParam contains the complete XML data resulting from the insertion of the new XML fragment

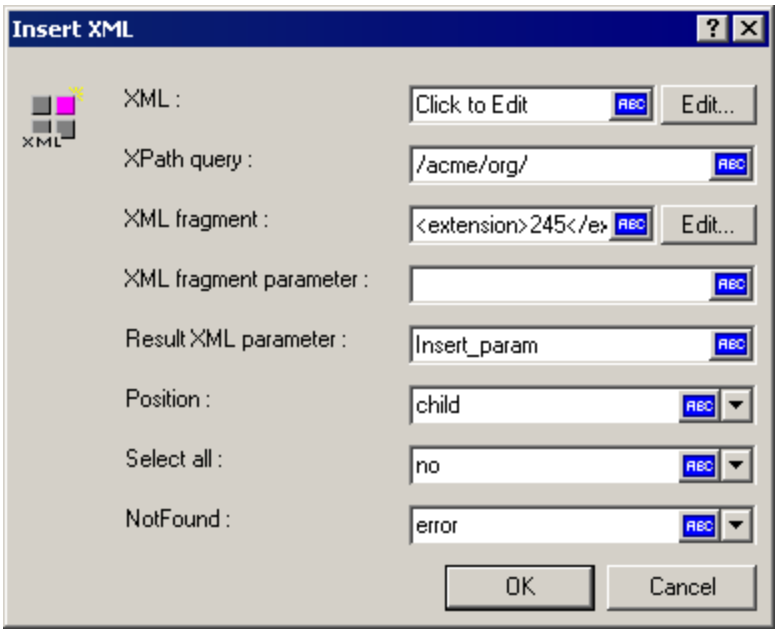
You can use the result parameters as input to other XML related functions such as Web Service calls. During replay, VuGen captures the value of the result parameter. In a later step, you can use that value as an input argument.

The functions that support result parameters are **lr_xml_insert**, **lr_xml_transform**, **lr_xml_replace**, **lr_xml_delete**, and **lr_xml_set_values**.

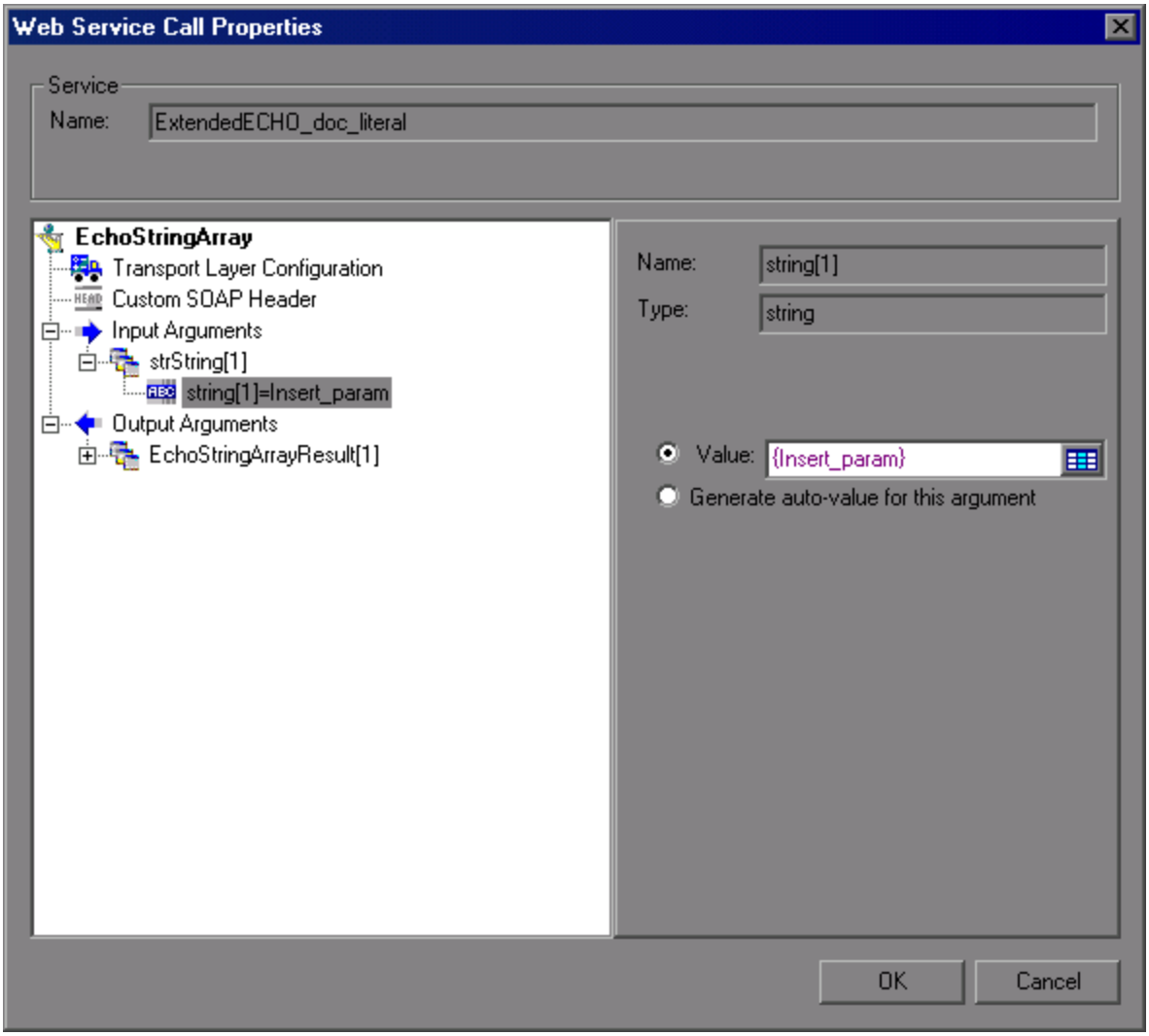
The following functions save values to a parameter other than the resultParam: **lr_xml_get_values** saves values to ValueParam and **lr_xml_extract** saves values to XMLFragmentParam. These values are also available for parameter substitution.

### Use the Result Parameter as Input

1. In the **Step Navigator**, double-click on an XML step to view its Properties.
2. In the Result XML Parameter box, specify a name for the **Result XML parameter** (or ValueParam and XMLFragmentParam).




3. Reference the parameter name as in input argument.




For more information, see "New Web Service Call Dialog Box" on page 778.

# Non-English Language Support

## Non-English Language Support Overview

VuGen supports multilingual environments, allowing you to use languages other than English on native language machines when creating and running scripts.

When working with languages other than English, the primary issue is ensuring that VuGen recognizes the encoding of the text during record and replay. The encoding applies to all texts used by the script. This includes texts in HTTP headers and HTML pages for Web Vusers, data in parameter files, and others.

If you need to use non-English symbols in paths to scripts, scenarios, results, or analysis sessions, make sure to select the appropriate locale in the your machine's **Region and Language** settings. Script names, however, must be in English.

Windows 2000 and higher lets you save text files with a specific encoding directly from Notepad: ANSI, Unicode, Unicode big endian, or UTF-8.

By default, VuGen works with the local machine encoding (ANSI). Some servers working with foreign languages, require you to work with UTF-8 encoding. To work against this server, you must indicate in the Advanced recording options, that your script requires UTF-8 encoding.

## Page Request Header Language

Before running a Web script, you can set the page's request header to match your current language. In the Internet Protocol runtime settings, you set the language of the *Accept-Language* request header. This header provides the server with a list of all of the accepted languages.

To set this value, select **Replay > Runtime Settings > Internet Protocol > Preferences > Advanced > Options > Accept-Language request header** and select the desired language.

For user interface details, see "Preferences View - Internet Protocol" on page 309.

## Convert Encoding Format of a String

You can manually convert a string from one encoding to another (UTF-8, Unicode, or locale machine encoding) using the **lr_convert_string_encoding** function with the following syntax:

```
lr_convert_string_encoding(char * sourceString, char * fromEncoding, char * toEncoding, char * paramName)
```

The function saves the result string (including its terminating NULL) in the third argument, *paramName*. It returns a 0 on success and -1 on failure.

The format for the **fromEncoding** and **toEncoding** arguments are:

| | |
|---|---|
| LR_ENC_SYSTEM_LOCALE | NULL |
| LR_ENC_UTF8 | "utf-8" |
| LR_ENC_UNICODE | "ucs-2" |

In the following example, **lr_convert_string_encoding** converts "Hello world" from the system locale to Unicode.

```
Action()
{
    int rc = 0;
```

```
    unsigned long converted_buffer_size_unicode = 0;
    char               *converted_buffer_unicode = NULL;
    rc = lr_convert_string_encoding("Hello world", NULL, LR_ENC_UNICODE,
"stringInUnicode");
    if(rc < 0)
    {
        // error
    }
return 0;
}
```

In the replay log, the output window shows the following information:

```
Output:
Starting action Action.
Action.c(7): Notify: Saving Parameter "stringInUnicode = H\x00e\x00l\x00l\x00o\x00
\x00w\x00o\x00r\x00l\x00d\x00\\x00"
Ending action Action.
```

The result of the conversion is saved to the *paramName* argument.

## Convert Encoding Format of Parameter Files

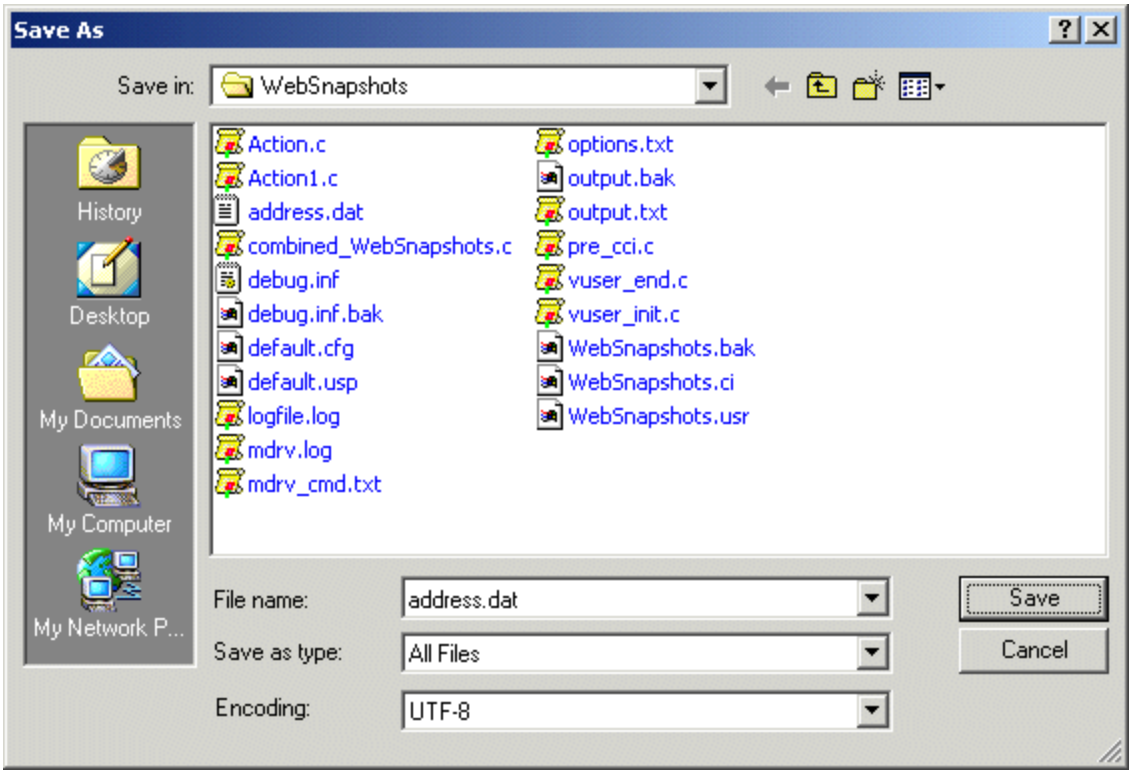
The parameter file contains the data for parameters that were defined in the script. This file, stored in the script's folder, has a **.dat* extension. When running a script, Vusers use the data to execute actions with varying values.

By default, VuGen saves the parameter file with your machine's encoding. When working with languages other than English, however, in cases where the server expects to receive the string in UTF-8, you may need to convert the parameter file to UTF-8. You can do this directly from Notepad, provided that you are working with Windows 2000 or higher.

### Apply UTF-8 Encoding to a Parameter File

1. Select **Vuser > Parameter List** and view the parameter properties.
2. In the right pane, locate the parameter file in the **File path** box.
3. With the parameter table in view, click **Edit in Notepad**. Notepad opens with the parameter file in csv format.
4. In the **Save as type** box, select *All Files*.

   In the **Encoding** box, select *UTF-8* type encoding.




5. Click **Save**. Notepad asks you to confirm the overwriting of the existing parameter file. Click **Yes**.

   VuGen now recognizes the parameter file as UTF-8 text, although it still displays it in regular characters.

## Record Web Pages with Foreign Languages

When working with Web or other Internet protocols, you can indicate the encoding of the Web page text for recording. The recorded site's language must match the operating system language. You cannot mix encodings in a single recording—for example, UTF-8 together with ISO-8859-1 or shift_jis.

### Automatically Record Foreign Language Web Pages.

To be recognized as a non-English Web page, the page must indicate the charset in the HTTP header or in the HTML meta tag. Otherwise, VuGen will not detect the EUC-JP encoding and the Web site will not be recorded properly. To instruct VuGen to record non-English requests as **EUC-JP** or **UT-8**, select **Record > Recording Options > HTTP Properties > Advanced > support charset** and select the appropriate option in the Recording Options dialog box, **HTPP Properties: Advanced** node. For user interface details, see "HTTP Properties > Advanced Recording Options" on page 195.

By selecting the **EUC-JP** or **UTF-8** option in the Recording Options, you are forcing VuGen to record a Web page with the selected encoding, even when it uses different encoding. If, for example, a non-EUC encoded Web page is recorded as EUC-JP, the script will not replay properly.

## Manually Record Foreign Language Web Pages

You can manually add full support for recording and replaying of HTML pages encoded in EUC-JP using the **web_sjis_to_euc_param** function. This also allows VuGen to display Japanese EUC-encoded characters correctly in Vuser scripts.

When you use **web_sjis_to_euc_param**, VuGen shows the value of the parameter in the Execution Log using EUC-JP encoding. For example, when you replay the **web_find** function, VuGen displays the encoded values. These include string values that were converted into EUC by the **web_sjis_to_euc_param** function, or parameter substitution when enabled in the **RuntimeSetting > Log > Extended Log**.

# Troubleshooting and Limitations for Non-English Languages

This section describes troubleshooting and limitations when working with non-English languages.

### Script/Scenario Names

- When recording COM, FTP, IMAP, SMTP, or POP3 protocols, the length of the script name is limited to 10 multi-byte characters (21 bytes).
- Non-English symbols in script names, and in paths to scripts, scenarios, results, or analysis sessions are not supported.
- The name and path of a scenario cannot contain multi-byte characters. . It is also recommended to use English characters for argument and parameter names.
- The GWT DFE extension does not support non-English characters in its classpath.

### Browser Configuration

If, during recording, non-English characters in the script are displayed as escaped hexadecimal numbers (For example, the string becomes **%DC%26**), you can correct this by configuring your browser not

to send URLs in UTF-8 encoding. In Internet Explorer, select **Tools > Internet Options and** click the **Advanced** tab. Clear the **Always Send URLs as a UTF-8** option in the Browsing section.
For more information, use the **web_sjis_to_euc_param** function described in the Function Reference.

### Jenkins Reports

After running a LoadRunner job in Jenkins, the links to several reports, such as the Performance Report and Transaction Summary, may not be translated.

### Protocol Limitations

**SMTP:** If you work with the SMTP protocol through MS Outlook or Outlook Express, the Japanese text recorded in a Vuser script is not displayed correctly. However, the script records and replays correctly.

### ContentCheck in Multilingual Environments

- This version supports ContentCheck rules in French, German, Spanish, and Italian. The correct language file should be installed according to the system locale.

- The suitable language file can also be copied from the installation disk here: **..\lrunner\MSI\setup\international\<lang>\dat\LrwiAedInstallation.xml** to the product's **dat** directory.

### Language Packs

- **LoadRunner Language Pack.** While installing the language pack, a warning message may be displayed that the LoadRunner Launcher Process is in use.
  **Workaround**: Click **Continue** to resume the installation.
- **LoadRunner Language Pack.** If you are working with a LoadRunner language pack, it is recommended that you install it before running LoadRunner for the first time.
- **Match Windows locale.** Install the language pack that is appropriate for the Windows locale. For example, if you are installing the Russian language pack, the Windows locale must also be Russian.
- **Framework Language Pack.** The language pack of .NET Framework needs to be installed to show the localized strings.
- **License Utility warnings.** A License Utility dialog box may open when installing a language pack. It will not affect the installation. Close it, and continue installing the language pack.
- **Certificate warnings.** VuGen may issue warnings about certificates when recording a Web - HTTP/HTML script on Windows 2012 R2 and Windows 8.1. This only occurs when you install VuGen on a non-English operating system.
  **Workaround**: Install the language pack for the language of the operating system. Do not delete the certificate after recording.
- **Recording functions.** VuGen cannot record a Vuser script for certain protocols if the installation is on a Chinese operating system, and the installation path contains Chinese characters.
- **Tutorial scripts.** After the language pack installation, all sessions and scripts for the tutorial will remain in English.
- **Menus and toolbars.** If you install a language pack after running LoadRunner for the first time, the menus and toolbars may not get translated.
  **Workaround**: Close the application and delete the **HKEY_CURRENT_USER\Software\<Folder Name>** folder from the registry, where <Folder Name> is the drive on which you installed the product.
  For example, if LoadRunner is installed on the C drive, the registry folder name would be **HKEY_CURRENT_USER\Software\C**. Restart LoadRunner.
- **Report Templates in Analysis.** If you install a language pack after running LoadRunner for the first time, the Report Templates in Analysis (**Reports > Report Templates**) may not get translated.
  **Workaround**: Close LoadRunner and copy the files from your LoadRunner installation package's **\Reporting** folder to the **<LoadRunner root>\bin\dat\Reporting** folder. Restart LoadRunner.
- **Uninstalling LoadRunner**. LoadRunner fails to uninstall on a Chinese operating system if the installation path contains Chinese characters.

### Non-Localized Installations on Foreign Language Operating Systems

- **Language support.** LoadRunner supports English and the native language of the machine's operating system. For example, if you are using Japanese Windows XP, you can work with

LoadRunner in Japanese and in English.

- **Installation path.** The path in which installation files for LoadRunner are located, and the path in which LoadRunner is installed, can contain only English characters.
- **Diagnostics add-in.** To use the Diagnostics add-in with Controller on a computer with a non-English operating system the Diagnostics_9.0_8.0_LR_Addin_QCCR1I52206 hotfix should be installed. For further assistance, contact customer support.
- **.NET Framework 3.5 failure.** Installing LoadRunner on a localized machine may result in a failure in the .NET Framework 3.5 installation process, and you will be asked to terminate the installation. This happens because the .NET 3.5 Framework installation attempts to download the Framework Language Pack but fails.
  **Workaround**: Terminate the LoadRunner installation according to the Installation wizard's instructions and invoke the installation again.
- **Online Help.** The search functionality may not function as expected for strings that contain Chinese/Japanese characters (except Japanese full-width Katakana).
  **Workaround**: Add a half-width space after each character in the search string.
- **Online Help.** For optimum performance of the online Help, install the latest JRE.
- **Japanese characters in Web - HTTP/HTML scripts.** If you set the advanced recording option to specify the encoding of an application, and the application uses different character encoding for different pages, then the recording log or script may display invalid Japanese characters. This does not cause any errors in the script replay.
- **Non-breaking spaces** in Web protocols for Far Eastern languages. A non-breaking space (      `\xA0', etc.) cannot be represented in some Far Eastern locale character sets (in which it is considered a lead byte). Instead, non-breaking spaces are converted to regular spaces (` `, `\x20', etc.), both during script code generation and replay. This may cause replay problems, such as mismatches in length due to eliminating multiple regular spaces.
  **Workaround**: Remove/add space(s) from/to the script so the comparison succeeds or specify regular expressions to avoid the issue.
- **Standalone installations.** The installation interface of the VuGen and Analysis standalones are in English and not localized.
- **Flex AMF call properties.** Multibyte symbols in Flex AMF call properties will be corrupted in the script text view.
- **rdp_type.** The **rdp_type** function does not support native language characters for both record and replay.
- **Word Completion.** Word completion does not work when Windows is configured to use the Ctrl+Space combination. This is common when using a Chinese keyboard.
  **Workaround**: Select **Complete word** from the **Edit** menu. Advanced users can disable Ctrl+Space for Chinese keyboards, by setting the following registry keys:
  - `[HKEY_CURRENT_USER\Control Panel\Input Method\Hot Keys\00000010]`
    `"Key Modifiers"=hex:00,c0,00,00`
    `"Target IME"=hex:00,00,00,00`
    `"Virtual Key"=hex:ff,00,00,00`

- [HKEY_CURRENT_USER\Control Panel\Input Method\Hot Keys\00000070]
  "Key Modifiers"=hex:00,c0,00,00
  "Target IME"=hex:00,00,00,00
  "Virtual Key"=hex:ff,00,00,00

- **ODBC and Oracle-2 Tier protocols.** When recording a script in VuGen using the ODBC or Oracle-2 Tier protocols, if you stop the recording while the AUT is still open, VuGen may crash.
  **Workaround**: Close VuGen and open the file **<installation folder>\dat\protocols\options\script\general.opt** in a text editor.
  Comment out the following line by adding a semicolon at the beginning of the line:
  `Option=DumpProcesses` so it looks like this: `;Option=DumpProcesses`
- **PDF reports.** In Analysis, a PDF report may be generated with unreadable characters if it contains non-English characters.
  **Workaround**: Before you generate the PDF file, change the font in the Report Template that you are using:
  a. Select **Reports > Report Templates**.
  b. Select the template that you want to use.
  c. In the **Detailed Report** section, select the **Format** tab.
  d. For each UI element in the list, change the font to a font that supports the language in which the report is written.

# Share Software Content Resources

Share software content resources with other VuGen users using Micro Focus AppDelivery Marketplace. You can download or contribute all types of content, including new protocols, documents, installation packages, and so on.

## To download content from Marketplace:

1. Go to Marketplace > Performance Engineering:
   - In VuGen: Click the **Marketplace** button on the toolbar, or select **Help > AppDelivery Marketplace**
   - Access directly: https://marketplace.microfocus.com/appdelivery/category/performance-engineering.
2. Click a file to download, and click **Download**.

## To contribute content to Marketplace:

1. **Prerequisite:** Join Marketplace as a developer: https://marketplace.microfocus.com/appdelivery/join
2. On your **Dashboard** page, click **ADD NEW ITEM**.

3. Give the item a name.
4. Complete the form and submit for review.

# Additional Components

You can install additional components that provide advanced features for working with LoadRunner. The setup files are located in the **Additional Components** folder inside the root folder of your LoadRunner installation package.

## Where to install additional components

The table below indicates which additional components are available, and where you should install each component:

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **Agent for Citrix Server** | SetupCitrixAgent.exe | Installs the Citrix Agent which enhances VuGen's capabilities in identifying Citrix client objects during Citrix protocol record and replay. For installation instructions, see "Install the LoadRunner Citrix agent on the Citrix server. (Optional)" on page 480.<br><br>The agent also enables you to use additional Citrix API functions. For details, see the Function Reference (**Help > Function Reference**). | Citrix server |
| **Agent for Microsoft Terminal Server** | SetupMSTerminalAgent.exe | Installs a utility that enhances the RDP protocol's recording mechanism in VuGen. For installation instructions, see "Installing the Microsoft Terminal Server Agent" on page 913. | RDP server |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **Assembly Crawler for Analysis API** | AssemblyCrawlerConsole.exe | Installs a command-line utility to build a .NET configuration file for a LoadRunner Analysis API application. For more information, open the **Analysis API Reference** from the **Start > Documentation** menu on the LoadRunner machine (not available with VuGen Standalone). | LoadRunner Analysis machine |
| **HostID Generator** | Host ID Generator tool, licidgenerator.exe | Opens the Host ID Generator utility that displays the computer's Host ID. This is useful when requesting a license. | LoadRunner Controller machine |
| **Network Virtualization** | • NV4ControllerSetup.exe<br>• NV4LGSetup.exe<br>• Linux setup packages: **<LoadRunner installation package>\Standalone Applications\linux_x64\NV4LG** | • NV4ControllerSetup.exe installs Network Virtualization for Controller, enabling emulation functionality.<br>• NV4LGSetup.exe installs the NV Insights Report component for VuGen, and Network Virtualization for the Windows load generator machines.<br>• The Linux packages install Network Virtualization for the Linux load generator machines.<br>For details, see "Network Virtualization Integration" on page 1237. | LoadRunner Controller, VuGen, and load generator machines |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **IDE Add-Ins** | • EclipseAddin \hp.lr.vugeneclipse42addin.jar<br>• LRVS2013IDEAddInSetup.exe<br>• LRVS2015IDEAddInSetup.exe | Installs add-ins for supported versions of Visual Studio or Eclipse enabling you to create Vuser scripts in your standard development environment using the LoadRunner API. This integration also allows you to run the test directly from Visual Studio or Eclipse, to test its functionality.<br>• Only 32-bit versions of Eclipse are supported. For more details on supported versions, see the System Requirements.<br>• To install the Visual Studio Add-in, Visual Studio must be installed in the default location.<br>• For the LRVS2015IDEAddIn for Visual Studio, Visual C ++ language must be installed to work with C++ .Net Vuser projects.<br>For details, see "Create Scripts in External IDEs" on page 877. | Visual Studio / Eclipse machine with VuGen |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **IDE Add-Ins Dev** | LREclipseIDEAddInDevSetup.exe<br>LRVS2013IDEAddInDevSetup.exe<br>LRVS2015IDEAddInDevSetup.exe | Setup files for developer add-ins for supported versions of Visual Studio and Eclipse, enabling you to create NUnit or JUnit tests in your standard development environment using the LoadRunner API.<br>• Only 32-bit versions of Eclipse are supported. For more details on supported versions, see the System Requirements.<br>• To install the Add-in, Visual Studio must be installed in the default location.<br>• For the LRVS2015IDEAddIn for Visual Studio, Visual C ++ language must be installed to work with C++ .Net Vuser projects.<br>For details, see "Create Scripts in External IDEs" on page 877. | Visual Studio or Eclipse machine with VuGen |
| **LoadRunner ProtocolSDK** | SetupLoadRunnerProtocolSDK.exe | Allows you to create and distribute custom LoadRunner protocols. For details, see "Protocol SDK" on page 912. | Any machine with Virtual Studio 2015 and WiX Toolset 3.10.3 - 3.11 |
| **mobileRemote Agent** | Select the relevant component for your operating system. | Enables you to capture a pcap file with Linux Redhat | |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **SAP Tools** | SapSpy.exe<br>VerifyScripting.exe | • **SAPGUI Spy.** Examines the hierarchy of GUI Scripting objects, on open windows of **SAPGUI Client for Windows**.<br>• **SAPGUI Verify Scripting.** Verifies that the SAPGUI Scripting API is enabled.<br>For details, see "Configure the SAP Environment" on page 677. | VuGen machine with SAPGUI client |
| **Third Parties** | Source files | The folder contains the source code of some third party software components which are being used in LoadRunner. | N/A |
| **Virtual Table Server** | SetupVTS.exe | Virtual Table Server (VTS) offers an alternative to standard LoadRunner parameterization. For details, see "Parameterization Overview" on page 357. | Any machine |

# Standalone Applications

The following LoadRunner standalone applications are available in your LoadRunner installation package's **/Standalone Applications** folder.

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **Analysis Standalone** | SetupAnalysis.exe | Installs LoadRunner Analysis as a standalone application. Install this to open LoadRunner results and create graphs and reports on a separate machine. For details, see "Introducing Analysis" on page 1269. | Any machine |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **Load Generator** | SetupLoadGenerator.exe | Installs the LoadRunner agent on the machine in order to run load tests. After you install this software, you access this machine from the Controller. For details, see "Load Generators" on page 983. | Any machine |
| **MI Listener** | SetupMIListener.exe | Installs the MI Listener, which servers as a router between the Controller and the LoadRunner agent. For details, see "Set Up Your LoadRunner System Over Firewalls" on page 1136. | Dedicated machine |
| **Monitors Over Firewall** | SetupMoFW.exe | Installs the Monitors Over Firewall component, allowing you to monitor servers located over a firewall. For details, see "Set Up Your LoadRunner System Over Firewalls" on page 1136. | Dedicated machine |
| **TruClient Standalone** | SetupTruClient.exe | Installs TruClient as a standalone application. Install this to tool to record Web applications with TruClient technology. You save the recordings to a script that can be used in a LoadRunner test run. For details, see the TruClient Help Center (select the relevant version). | Any machine |
| **VuGen Standalone** | SetupVuGen.exe | Installs LoadRunner Virtual User Generator (VuGen) as a standalone application, allowing you to create scripts for a load test. For details, see "Introducing Virtual User Generator" on page 56. | Any machine |

# Protocol SDK

The Protocol SDK package allows you to create custom LoadRunner protocols from within Visual Studio.

LoadRunner provides this package as an extension to Visual Studio 2015.

## Install the Protocol Library Package

1. **Prerequisite.** Make sure the following are installed:
   - Visual Studio 2015 with the Visual C++ language enabled
   - WiX toolset 3.10.3 to 3.11
   - WiX toolset add-in for Visual Studio 2015. If the WiX toolset was installed before Visual Studio 2015, you will need to reinstall it.
2. Locate the installation file, **SetupLoadRunnerProtocolSDK.exe**, in your LoadRunner installation package's **Additional Components\LoadRunnerProtocolSDK** folder. You can install this extension on a machine that does not have an installation of LoadRunner.
3. Follow the installation wizard to completion.
4. Create a new test using the **LoadRunner Protocol SDK** template. For details, see the Protocol SDK documentation, accessible from the Start menu or from the following location: **<LoadRunner root>\LoadRunner Protocol SDK\documents\LR_Protocol_SDK\webframe.html**.

## Upgrade Protocol SDK Projects from Visual Studio 2012

The Protocol SDK is now an extension to Visual Studio 2015, and no longer an extension to Visual Studio 2012. Consequently, all projects created with LoadRunner 12.50 and earlier must be upgraded. For details, see **Upgrading to Visual Studio 2015** in the Protocol SDK documentation, available after installing the Protocol SDK.

For more details about the Protocol SDK, see the LoadRunner knowledge base.

# Installing the Microsoft Terminal Server Agent

## Install the Agent for Microsoft Terminal Server

1. **Prerequisite:**
   - Choose the RDP server machine on which you want to install the Microsoft terminal server agent. (Do not install the agent on a Load Generator machine.)
   - If you are upgrading, uninstall any earlier agent versions. See "Uninstall the Agent for Microsoft Terminal Server" on the next page.
2. If your server requires administrator permissions to install software, log in as an administrator to the server.
3. Locate the installation file, **Setup.exe**, in the LoadRunner installation package, in the **Additional Components\Agent for Microsoft Terminal Server** folder.
4. Follow the installation wizard to completion.

> **Note:** To use the agent, you must set the recording options before recording a Vuser script. In

the Start Recording dialog box, click **Options**. In the Advanced Code Generation node, check **Use RDP Agent**.

### Uninstall the Agent for Microsoft Terminal Server

1. If your server requires administrator privileges to remove software, log in as an administrator to the server.
2. From **Add or remove a program**, remove **Software Agent for Microsoft Terminal Server**.

## Troubleshooting and Limitations for Additional Components

**Secure Channels**

- You cannot use the Host Security Manager utility to update security settings on Linux load generators that use rsh (remote shell) to connect to the Controller.
- You cannot use the Host Security Manager utility to change the security mode of the load generator located over a firewall from off to on.
- When the load generator is located over a firewall, if the load generator and Controller have different security modes, communication cannot be established.
- If the Controller machine is using secure channel communication, the MI Listener should not be installed on the same machine as the Controller.

# Troubleshooting and Limitations for VuGen

This section describes general troubleshooting and limitations for VuGen. For additional protocol-specific limitations, see the troubleshooting sections for each of the protocols.

## Internet Explorer Issue

When using IE on Windows server machines, the browser's enhanced security (ESC) blocks certain actions. This may prevent the automatic download of files that are necessary for your workflow.

## Error Messages

For Media Player - MMSscripts, if you specify a non-default bandwidth in the runtime settings, the Vuser may cause an error during replay.

**Note:** This protocol is currently supported for replay only. All support will be removed in a future version.

# Slow replay of JavaScript language scripts

If your JavaScript language script runs slow in VuGen, disable the debugging option: In the **Internet Protocol** runtime settings, open the **Preferences** view and locate the JavaScript section. Clear the **Enable JavaScript debugging mode** option. For details, see "Preferences View - Internet Protocol" on page 309.

# Installing and Upgrading JVMs

If you install or upgrade a JVM while VuGen is open, you will need to restart VuGen before continuing to record or develop a script.
**Workaround:** Add an entry "**about:internet**" to the Trusted Sites in Internet Explorer.

# McAfee Compatibility Issues

- When McAfee On-Access file scan is enabled, it may block/revert some of the file-writing operations that VuGen performs during code generation, particularly when asynchronous communications have been detected. As a result, the content of some of the script actions may be lost; these actions will be empty instead of containing the required generated Async code.

  **Workaround:** Exclude LoadRunner files from the scan.
- When McAfee Host Intrusion Prevention (HIP) is enabled, a crash may occur while recording a Web-based protocol script, including Java over HTTP, especially if the current user account is a limited account (non-admin). In addition, you may experience browser malfunctions, even when working with a full-privileged user account.
- It is recommended that you close all anti-virus applications, such as McAfee or Aladdin's eSafe, before installing LoadRunner.
- McAfee's anti-virus application blocks port 443, which is the default port of the LoadRunner agent.

  **Workaround:** Manually enable this port. To enable the port, open the McAfee Configuration dialog box. In the Firewall Policy tab, add a new rule to allow Port 443 - Action: Permit IP: TCP, Incoming traffic for the LoadRunner Agent Process.
- When recording a .NET script on non-English operating systems with McAfee anti-virus active, it may issue the following message "The solution has been changed externally".

  **Workaround**: Add the vugen.exe process to the Low-Risk processes in the McAfee antivirus On-Access Scan Properties.
- When you correlate a value from a snapshot, VuGen may create a boundary-based correlation, even though the recording correlation option is set to use regular expressions.

# Controller

Controller is a component of LoadRunner, enabling you to run and monitor LoadRunner tests.

To learn more, see "Introducing Controller" below.

# Introducing Controller

Welcome to LoadRunner Controller.

Controller is a tool for creating and controlling LoadRunner *scenarios*. A scenario defines the events that occur during each testing session. It controls the number of users to emulate, the actions they perform, and the machines on which they run their emulations. You use scenarios to create load tests to check the reliability and strength of your servers.

## Primary Items in Scenario

The following are the primary items that you define in your scenario:

- **Scenario type.** A goal-oriented or manual scenario.
- **Tests.** The LoadRunner scripts or unit tests to run.
- **Machines.** The machines upon which to run the tests.
- **Vusers.** The number of virtual users (Vusers) to run on each machine.
- **Scheduling.** How to load the Vusers.
- **Monitors.** Which measurements to monitor during the test run.

### Scenario type

When you open Controller for the first time, it prompts you to select a type of scenario: goal-oriented or manual.

- **Goal-oriented** scenario. Define the goals you want your test to achieve and LoadRunner automatically builds a scenario for you based on these goals. For example you can define a goal for a specific number of Vusers to run simultaneously. Alternatively, you can define a goal to test your server performance such as Pages per Minute, Hits per Second, or Transactions per Second. For details, see "Goals Types for Goal-Oriented Scenarios" on page 943.
- **Manual** scenario. Add Vusers and select scripts/unit tests manually. You then distribute them on the available machines. For details, see "Manual Scenarios" on page 942.

## Scripts and Test Types

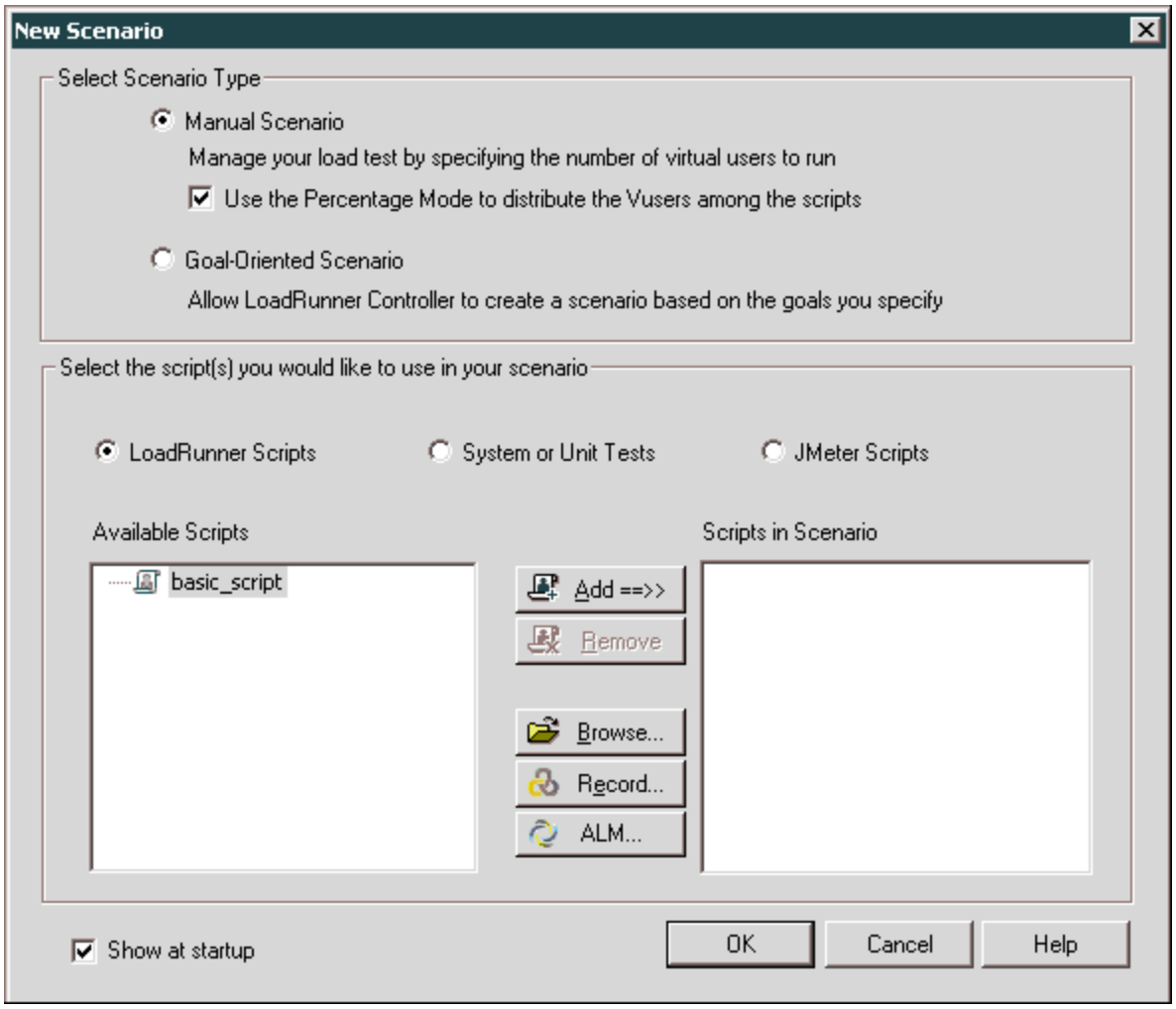
During a scenario run, Controller runs Vuser scripts or system/unit tests. Vuser scripts are test scripts created with the *LoadRunner Virtual User Generator*, also known as VuGen.

System/Unit tests refer to Selenium tests or *NUnit* and *JUnit* tests created in external development environments, such as Microsoft Visual Studio or Eclipse. You can work in your native application and prepare unit tests in binary form, such as **.dll** or **.jar** files, and then run them from Controller.

> **Note:** For more best practice information, see the LoadRunner Resources site.

LoadRunner add-ins allow you to integrate the LoadRunner API with supported versions of Microsoft Visual Studio or Eclipse, and run tests from your native environment. For details, see "Additional Components" on page 1566.

Controller's opening dialog box prompts you to select the scripts and/or system/unit tests to include in the scenario. For details, see "New Scenario Dialog Box" on page 969.




All of your selections, along with the test paths, are saved in a scenario file (**.lrs**). You define all of the other aspects of your scenario in Controller's **Design** tab. For details, see "Design Tab" on page 962.

# Licenses

To view your current license or add new ones, see "LoadRunner License Utility" on page 930.

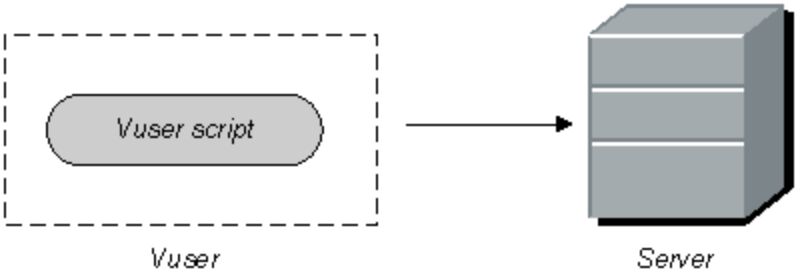
For details about the license bundles, see the **License Bundles** document on the LoadRunner Resources site.

### See also:

- Load Testing with LoadRunner

# Controller Technology

In Controller, you define a number of Vusers (excluding GUI Vusers) to generate load on a server by submitting input directly to the server. Vusers do not operate client applications—they access the server using LoadRunner API functions. These API functions emulate the input from an actual application.




Because Vusers are not reliant on client software, you can use Vusers to test server performance even before the client software has been developed. Since Vusers do not have a user interface, the amount of system resources required is minimal. This allows you to run large numbers of Vusers on a single workstation.

## Example

The following example illustrates the use of Vusers in a scenario: Suppose that you have a Web-based database server that maintains your customer information. The information is accessed by numerous customer service personnel who are located throughout the country. The server receives the queries, processes the requests, and returns responses via the Web to field personnel.

You want to test the response times of the entire system when numerous service personnel simultaneously access the server. Using LoadRunner, you could create a scenario with several hundred Vusers, each one accessing the server database. The Vusers enable you to emulate and measure the performance of your database and Web servers under the load of many users.

To emulate the Vusers, you create a script to define their actions. A Vuser script includes functions that control the script execution and specify the input that the Vuser submits to the server. For more information, see "Vusers" on page 57.

For the database server example above, you could create a Vuser script that performs the following actions:

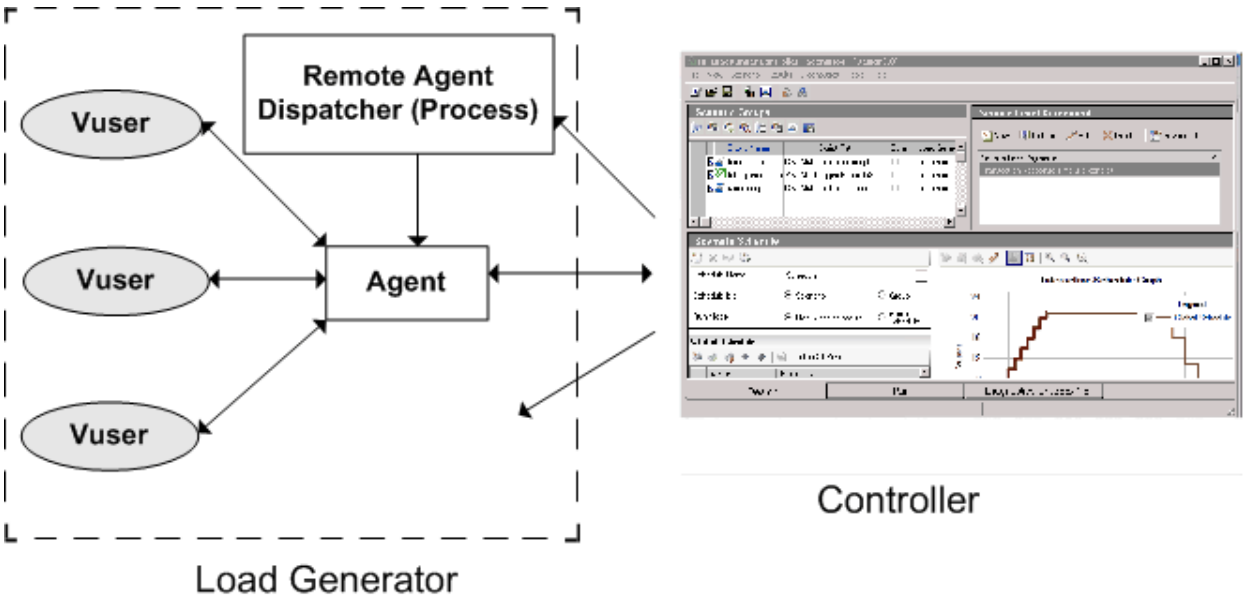
- Logs in to the Web application

- Connects to the database server
- Submits an SQL query
- Retrieves and processes the server response
- Disconnects from the server and the Web

# LoadRunner Agents

To maximize your testing coverage, you distribute Vusers over several load generator machines. A load generator machine is a machine upon which the **Remote Agent Dispatcher (Process)** and a LoadRunner **Agent** are installed.

When you run the LoadRunner installation, you specify to install only these components. These components allow Controller to communicate with the load generator machine.




- **Remote Agent Dispatcher.** The Remote Agent Dispatcher (Process) enables Controller to start applications on the load generator.
- **Agent.** The LoadRunner Agent enables Controller and the load generator to communicate with each other. When you run a scenario, Controller instructs the Remote Agent Dispatcher (Process) to launch the LoadRunner agent. The agent receives instructions from Controller to initialize, run, pause, and stop Vusers. At the same time, the agent also relays data on the status of the Vusers back to Controller.

# LoadRunner Terminology

The following terminology is commonly used in LoadRunner:

| | |
|---|---|
| **Scenario** | A scenario is a sequence of events that emulate the hypothetical actions of real users on your application. |
| **Vusers** | In the scenario, LoadRunner replaces real users with **virtual users** or **Vusers**. While a workstation accommodates only a single human user, many Vusers can run concurrently on a single workstation. In fact, a scenario can contain tens, hundreds, or even thousands of Vusers. |
| **Vuser Scripts** | The actions that a Vuser performs during the scenario are described in a Vuser script. When you run a scenario, each Vuser executes a **Vuser script**. The Vuser scripts include functions that measure and record the performance of your application's components. |
| **Transactions** | To measure the performance of the server, you define **transactions**. A transaction represents an action or a set of actions that you are interested in measuring. You define transactions within your Vuser script by enclosing the appropriate sections of the script with **start** and **end** transaction statements. For example, you can define a transaction that measures the time it takes for the server to process a request to view the balance of an account and for the information to be displayed at the ATM. |
| **Rendezvous points** | You insert **rendezvous points** into Vuser scripts to emulate heavy user load on the server. **Rendezvous points** instruct Vusers to wait during test execution for multiple Vusers to arrive at a certain point, so that they may simultaneously perform a task. For example, to emulate peak load on the bank server, you can insert a rendezvous point instructing 100 Vusers to deposit cash into their accounts at the same time. |
| **Controller** | You use the LoadRunner Controller to manage and maintain your scenarios. Using Controller, you control all the Vusers in a scenario from a single workstation. |
| **Load Generator** | When you execute a scenario, Controller distributes each Vuser in the scenario to a **load generator**. The load generator is the machine that executes the Vuser script, enabling the Vuser to emulate the actions of a human user. |
| **Performance analysis** | Vuser scripts include functions that measure and record system performance during load-testing sessions. During a scenario run, you can monitor the network and server resources. Following a scenario run, you can view **performance analysis** data in reports and graphs. |

# Plan the LoadRunner Implementation

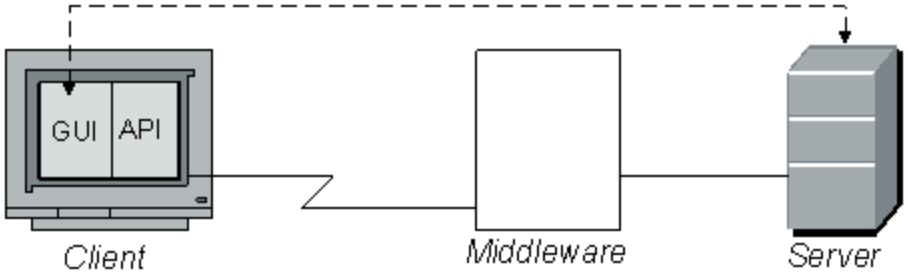
This task describes how to plan the LoadRunner implementation as part of the load test planning process.

# Define the scope of performance measurements

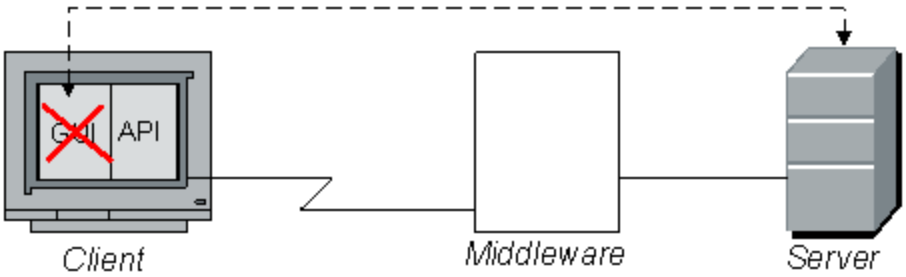
You can use LoadRunner to measure **response time** at different points in the application. Determine where to run the Vusers and which Vusers to run according to the test objectives:

- **Measuring end-to-end response time.** You can measure the response time that a typical user experiences by running a GUI Vuser at the front end. GUI Vusers emulate real users by submitting input to and receiving output from the client application.

  You can run GUI Vusers at the front end to measure the response time across the entire network, including a terminal emulator or GUI front end, network, and server.

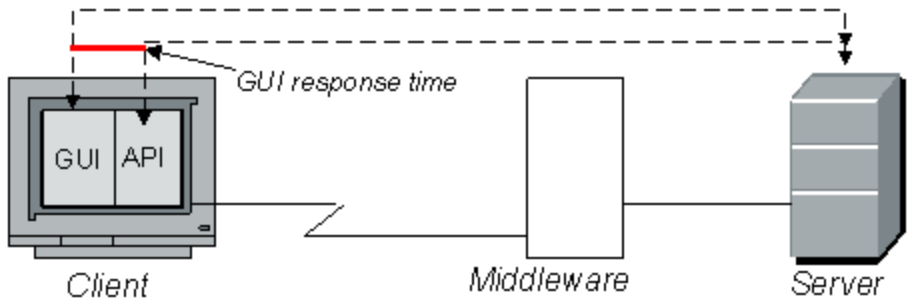



- **Measuring network and server response times.** You can measure network and server response time, excluding response time of the GUI front end, by running Vusers (not GUI) on the client machine. Vusers emulate client calls to the server without the user interface. When you run many Vusers from the client machine, you can measure how the load affects network and server response time.

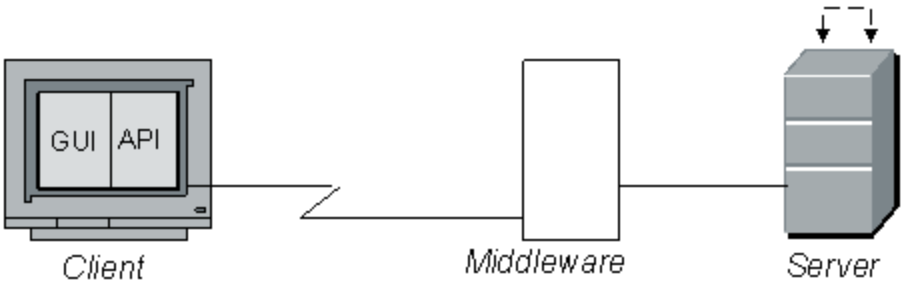



- **Measuring GUI response time.** You can determine how the client application interface affects response time by subtracting the previous two measurements:

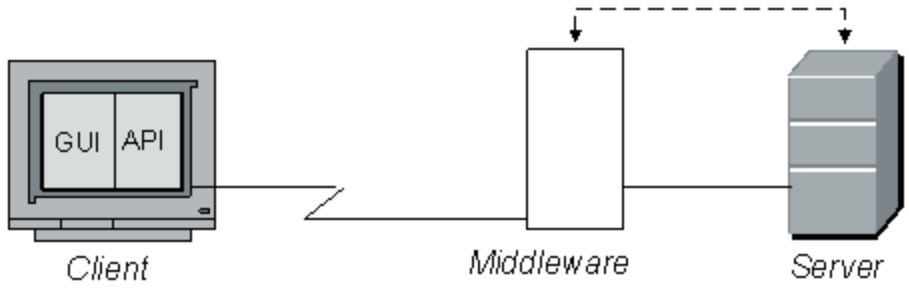
  **GUI response time = end-to-end - network and server**




- **Measuring server response time.** You can measure the time it takes for the server to respond to a request without going across the network. When you run Vusers on a machine directly connected to the server, you can measure server performance.




- **Measuring middleware-to-server response time.** You can measure response time from the server to middleware if you have access to the middleware and its API. You can create Vusers with the middleware API and measure the middleware-server performance.




## Define Vuser activities

Create Vuser scripts based on your analysis of Vuser types, their typical tasks, and your test objectives. Since Vusers emulate the actions of a typical end-user, the Vuser scripts should include the typical end-user tasks. For example, to emulate an online banking client, you should create a Vuser script that performs typical banking tasks. You would browse the pages that you normally visit to transfer funds or check balances.

You decide which tasks to measure based on your test objectives and define **transactions** for these tasks. Transactions measure the time that it takes for the server to respond to tasks submitted by

Vusers (end-to-end time). For example, to check the response time of a bank Web server supplying an account balance, define a transaction for this task in the Vuser script.

In addition, you can emulate peak activity by using **rendezvous points** in your script. Rendezvous points instruct multiple Vusers to perform tasks at exactly the same time. For example, you can define a rendezvous to emulate 70 users simultaneously updating account information.

> **Note:** You can also add JMeter scripts to your scenarios. For details, see "JMeter Tests" on page 945.

## Select Vusers

Before you decide on the hardware configuration to use for testing, determine the number and type of Vusers required. To decide how many Vusers and which types to run, look at the typical usage model, combined with the testing objectives. Some general guidelines are:

- Use one or a few GUI users to emulate each type of typical user connection.
- Run multiple Vusers to generate the rest of the load for each user type.

For example, suppose that you have five kinds of users, each performing a different business process:

| Usage Model | GUI | Other |
|---|---|---|
| 100 customer service users in New York (LAN connection) | 2 | 98 |
| 30 customers in Europe (dial-in ISDN connection) | 2 | 28 |
| 5 background batch processes | _ | 5 |
| 150 customers (terminal connection) | _ | _ |
| 6 managers | 2 | 4 |

## Choose testing hardware/software

The hardware and software should be powerful and fast enough to emulate the required number of virtual users. Refer to the Readme file for specific hardware requirements.

To decide on the number of machines and correct configuration, consider the following:

- It is advisable to run LoadRunner Controller on a separate machine.
- Each GUI Vuser requires a separate Windows-based machine; several GUI Vusers can run on a single Linux machine.

> **Note:** If you plan to run your Vuser scripts on Linux-based load generators, it is recommended that you first test Linux compatibility for the scripts in VuGen. For details, see "Check Linux Compatibility" on page 297.

- Configuration of the test machine for GUI Vusers should be as similar as possible to the actual user's machine.

> **Note:** The results file requires a few MB of disk space for a long scenario run with many transactions. The load generators also require a few MB of disk space for temporary files if there is no NFS. For more information about runtime file storage, see "Runtime File Storage Locations" on page 1068.

### So how many Vusers can I run on a Load Generator?

During testing done on a large scenario with a strong VM machine, the following setup was proved to sustain **2000-2500 Vusers** (and possibly more):

- 8 Core CPU
- 32 GIG of RAM
- 160 HD
- Network 1 GIG card

Notes on achieving this capacity:

- This is for web protocol usage. The number of Vusers may be less with non-web protocols.
- Disable Vmotion/generic allocation settings, depending on the virtualization server type.
- Monitor private cloud servers (like ESX matrix) to make sure the VM server is not overloaded.
- Results may be affected by factors such as AUT, business processes, or coding efficiency.

## The LoadRunner Testing Process

The following section provides a general overview of the LoadRunner testing process.

### 1. Planning the Test

Successful load testing requires that you develop a thorough test plan. A clearly defined test plan will ensure that the LoadRunner scenarios that you develop will accomplish your load testing objectives. For more information, see "Planning Load Test Scenarios" on page 936.

### 2. Creating the Vuser Scripts

Vusers emulate human users interacting with your Web-based application. A Vuser script contains the actions that each Vuser performs during scenario execution.

In each Vuser script, you determine the tasks that will be:

- Performed by each Vuser

- Performed simultaneously by multiple Vusers
- Measured as transactions

For more information on creating Vuser scripts, see "Enhancing a Script for Load Testing - Overview" on page 336.

### 3. Designing the Scenario

A scenario describes the events that occur during a testing session. A scenario includes a list of machines on which Vusers run, a list of scripts that the Vusers run, and a specified number of Vusers or Vuser groups that run during the scenario. When designing the scenario, you set the scenario configuration and scheduling which determines how all the load generators and Vusers behave while the scenario runs.

You design scenarios using Controller. For information about LoadRunner scenarios, see "Designing Scenarios" on page 942.

### 4. Running the Scenario

You emulate user load on the server by instructing multiple Vusers to perform tasks simultaneously. While the scenario runs, LoadRunner measures and records the transactions that you defined in each Vuser script. You can set the level of load by increasing and decreasing the number of Vusers that perform tasks at the same time and you can also monitor your system's performance online. For more information, see "Running Scenarios in Controller" on page 1093.

### 5. Monitoring the Scenario

You configure the LoadRunner monitoring components to identify bottlenecks on the system and determine which element is causing performance degradation, for example, file locking, resource contention, and network overload. Use LoadRunner in conjunction with the new network and machine monitoring tools to create load and measure performance at different points in the system. For more information on monitoring, see "Monitoring Load Test Scenarios" on page 1171.

### 6. Analyzing Test Results

During scenario execution, LoadRunner records the performance of the application under different loads. You use LoadRunner's graphs and reports to analyze the application's performance. For more information about LoadRunner's reports and graphs, see "Introducing Analysis" on page 1269.

## LoadRunner Licensing

To run Vusers with Controller there must be a valid LoadRunner license installed on the Controller machine.

You use the LoadRunner License Utility to manage your LoadRunner licenses. The LoadRunner License Utility enables you to:

- View the details of licenses that are currently installed
- Install additional licenses

> **Tip:** Use the Micro Focus SW Usage Hub tool to track license usage across your network.

## About LoadRunner Licenses

To run Vusers with Controller there must be a valid LoadRunner license installed on the Controller machine.

You use the LoadRunner License Utility to manage your LoadRunner licenses. The LoadRunner License Utility enables you to:

- View the details of licenses that are currently installed
- Install additional licenses

### License types

The following types of licenses are available:

| Type | Description |
|---|---|
| Time Limited | License is valid for a limited period only, up to 10 years. |
| VUDs | License is issued with a limited capacity. The capacity is defined by the measurement *Vuser-days* or *VUDs*.<br><br>For example, the capacity of a VUD license may be 1000 VUDs. Each day that the Controller is used to run Vusers, the maximum number of Vusers that ran simultaneously on that day is deducted from the remaining license capacity. If a maximum of 200 Vusers ran on day 1, then 800 VUDs will remain in the license. For more information, see "VUD Licenses" on the next page. |

### License bundles

There are several types of LoadRunner license bundles available.

The **Community** license is free of charge. It includes:

- 50 Vusers
- Access to all the protocols, with the exception of GUI (UFT)

Other bundles allow you to run additional Vusers on selected protocols. For more information, see the License Bundles documentation.

> **Note:** When using "Noise Scenarios" on page 950, for each valid license you can run 10 noise

generator-type Vusers.

## VUD Licenses

A **VUD license** is a Vuser-day license, issued with a limited capacity per Vuser day.

- For the calculation of VUDs consumed in a VUD license, you can specify at what time each new day begins. LoadRunner determines the maximum number of Vusers that ran during the 24-hour period after the start time.
- To change the time at which a new day begins, click the **VUDs will begin at time** link in the LoadRunner License Utility.
- A VUD license may have an expiration date and therefore be time-limited.
- If you have both a non-VUD license (eg, Time Limited license) and a VUD license for a particular Vuser protocol, LoadRunner always uses the non-VUD license before consuming the VUD license. That is, if 500 Vusers were run on a particular day and there is a non-VUD license for 400 Vusers, then 100 VUDs are deducted from the VUD license.
- A Vuser is included in the VUD count as soon as the Vuser reaches the **Initialization** status in a scenario. Consumed VUDs are deducted from the available VUD capacity at the start of the following day.

  For example, suppose you purchased 100 VUDs for the Web 2.0 bundle:

  - At some point of time, T, on the date D, you start a scenario, with X Web -HTTP/HTML Vusers, where X is less than or equal to 100.
  - During the next 24 hours you can start and stop any number of scenarios, as long as the number of Vusers in each of scenarios is less than or equal to 100.
  - When point T on the date D + 1 arrives (exactly 24 hours after the first scenario launch), the Controller looks back at the previous 24-hours interval, locates the largest amount of Vusers that were ran simultaneously and subtracts it from 100. For example, if during the last 24 hours you ran three scenarios, with 20, 40 and 30 Vusers, the Controller will detect the use of 40 VUDs and leave you with 60.
  - If a certain scenario is in progress during the time of the calculation (at the time T), it will be included in the calculation for both days on which it ran. For example, if you launch a scenario with 40 Vusers and it runs for 30 hours, then you will have used 40 VUDs for two days, and at the time T on the date D + 2 you will be left with 100 - 40 – 40 = 20 VUDs.
  - If a scenario runs for more than one day, the maximum VUD usage is calculated and deducted based on the day that the scenario began.

## License Consumption

- A multi-protocol Vuser runs only if licenses are available for all protocols in the Vuser script.
- If a multi-protocol script contains protocols from several bundles, you need licenses for both bundles to run the Vuser script. The license for each Vuser is subtracted from all relevant bundles. If, at a

certain point, the license of one of the bundles runs out, the Vuser will start to fail, even if there are still valid licenses for some of the other protocols.

- If you purchased licenses for several different bundles (such as Web 2.0 and Oracle E-business) or several different license types (such as Time Limited and VUD) that share common protocols, and you want to run Vusers that belong to both bundles/types, the licenses are consumed in the following order:

  **License type:**

  a. Temporary License

  b. Time Limited license

  c. VUD license

  **Protocol amount:** The bundle with the smaller number of protocols will be consumed before the bundle with a larger number of protocols.

  **Capacity:** The license with the larger capacity will be consumed before the license with the lower capacity.

## Install LoadRunner Licenses

To run Vusers from Controller, you must install the appropriate LoadRunner licenses on the Controller machine. For more information, see "About LoadRunner Licenses" on page 926.

Use the LoadRunner License Utility to install LoadRunner licenses.

In this topic:

- "LoadRunner Community license (free)" below
- "Paid LoadRunner license" on the next page

### LoadRunner Community license (free)

To install a LoadRunner Community license:

1. Open the LoadRunner License Utility: from the Windows Start menu, go to **Micro Focus > LoadRunner > License > LoadRunner License Utility.**
2. Obtain the license:
   a. Click **Sign up for free license**.
   b. Enter your details.
      - All the details are mandatory.
      - The email address must be valid and accessible.
   c. Accept the terms of service, and click **Sign up**.

   A message containing the license data and an activation code is sent to the email address you provided.
3. If this is the first time you are signing up for a free license, follow the license activation instructions in the email you received.

If you are a returning user, the relevant parts of the license are automatically activated and added to the License Summary.

4. If Controller is running, restart it.

## Paid LoadRunner license

If you have a valid order number or subscription (SAID), use your license file or key to install the license or licenses on your LoadRunner machine.

1. **Prerequisite: Obtain your license file or license key.**
   - **License file.** A license file contains the license keys for one or more licenses. When you use a license file to install the new licenses, the LoadRunner License Utility reads the license file and extracts all the license keys that are included in the license file. You can then select which of the available licenses to install. You may choose to use a license file to install LoadRunner licenses because the license file enables you to install multiple licenses simultaneously.
   - **License key.** A license key enables you to install just a single license at a time. You may receive the license key directly from your customer service representative, or the license key may be included in a license file that you received.

   **If you do not have a license file or key**, you can obtain one from the Software License and Downloads Portal as follows:

   a. Go to the Software Licenses and Downloads Portal and follow the onscreen instructions.
   b. If you are an existing customer, click the **Update** link to get a new license key for the latest version.

   For more information, see the instructions provided in the portal's Contact Us / Self Help page, or click Contact Licensing Delivery Center to request assistance if required.
2. **Install the license.**
   a. Open the LoadRunner License Utility: from the Windows Start menu, go to **Micro Focus > LoadRunner > License > LoadRunner License Utility**.
   b. Click **Install New Licenses**, and select and option for installing your license:

| | |
|---|---|
| Using a **license file** ... | i. Click the **Browse** button to the right of **License file**, and locate your license file<br>ii. Click **View License File Content** to display details of the licenses that are included in the license file.<br>iii. Select the licenses to install. |
| Using a **license key**... | i. Click **Install a license using a license key**.<br>ii. Enter your license key. |

   c. Click **Install**.
   d. Click **Close**. In the License Summary table, make sure that the new licenses appear in the list of

installed licenses.

e. If Controller is running, restart it.

## LoadRunner License Utility

You use the LoadRunner License Utility to install LoadRunner licenses, and to view your license information.

The **Community** license provides 50 Vusers at no cost. These Vusers are valid for all protocols except for GUI (UFT).

To launch the LoadRunner License Utility, from the Windows **Start** menu, go **to Micro Focus > LoadRunner > License > LoadRunner License Utility**.

The LoadRunner License Utility displays the following information:

| | |
|---|---|
| **Host ID** | Identifies the computer on which the Controller is installed. You may need to provide the Host ID when purchasing new LoadRunner licenses. |

| | |
|---|---|
| **License Summary** | Displays a list of the licenses that are installed on the computer. Click on any license in the table to display additional details about the license. The **Vuser protocols included in the selected license** box displays a list of the Vuser protocols that are included in the selected license.<br>• **Status.** Indicates the status of the license.<br>  • **Valid** **.** Indicates that the license in current and functional.<br>  • **Invalid** **.** Indicates that the license is no longer valid. For one of the following reasons:<br>  ◦ A **Time Limited** license expiration date has passed.<br>  ◦ The remaining capacity of a **VUD** license is zero.<br>  **Note:** A license may become temporarily invalid if the License Utility detects that the system clock has been tampered with. To restore the affected licenses, reset the system clock to the current date and time.<br>  • **To be Expired** **.** Indicates that the license will expire within 30 days.<br>  By default, the License Summary table does not show invalid licenses. Select the **Show invalid licenses** check box to show invalid licenses.<br>• **Locked.**<br>  • **Locked.** Indicates that the license can be installed only on the machine for which it was generated (with a unique HostID), and not on any other machine.<br>  • **Unlocked.** Indicates that the license can be installed on any computer.<br>• **License Bundle.**<br>  Indicates the name of the Vuser protocol bundle to which the license applies. The license enables the Controller to run Vusers of any protocol that is included in the protocol bundle. To display a list of the Vuser protocols that are included in a bundle, click the license in the **License Summary** table. A list of the associated Vuser protocols is displayed at the bottom of the LoadRunner License Utility.<br>  **Note:** A **Partner License** icon that appears to the left of a license bundle indicates that the license is for a LoadRunner partner, and not for standard Vuser protocols. Partner licenses enable third-party applications to be controlled by the LoadRunner Controller.<br>• **Type.** Indicates the type of license:<br>  • **Time Limited** licenses are valid for a limited period only, up to 10 years.<br>  • **VUD** licenses are issued with a limited capacity. The capacity is defined by the measurement *Vuser-days* or *VUDs*. For example, the capacity of a VUD license may be 1000 VUDs. Each day that the Controller is used to run Vusers, the maximum number of Vusers that ran simultaneously on that day is deducted from the remaining license capacity. If a maximum of 200 Vusers ran on day 1, then 800 |

| | |
|---|---|
| | VUDs will remain in the license.<br>For more information, see "About LoadRunner Licenses" on page 926.<br>• **Expiration Date.** Indicates the date and time when the **Time Limited** or **VUDs** license expires.<br>• **Capacity.**<br>  • **Time Limited:** Indicates the maximum number of Vusers (of the type specified by the license bundle) that can be run simultaneously from Controller.<br>  • **VUDs:** Indicates the number of VUDs that remain in the license. |
| **Show invalid licenses** | When selected, displays LoadRunner licenses that are installed that are invalid. |
| **Vuser protocols included in the selected license** | Displays the Vuser protocols that are included in the selected license. |
| **VUDs will begin at <time>** | Opens a dialog box where you can select when VUDs should begin. |
| **Install New Licenses** | Opens the New License dialog box which enables you to install new LoadRunner licenses. |
| **Sign up for free license** | Opens a dialog box where you can subscribe for a free Community license. |

## Troubleshooting Licenses

- If the computer system time is earlier than the starting time of the license, the license will not be valid. To overcome this, generate the license with appropriate starting time or wait until the starting time of the license.
- If you have a temporary license, contact your customer service representative to obtain a Time Limited license.
- If LoadRunner does not accept your license key, perform the following checks:

- Make sure you typed in the license key exactly as it was given to you.
  - The license key must include the required spaces.
  - The license key is case sensitive.
- If you receive a permission denied error message during Controller startup, you must grant **Full Control** permission for the Registry's HKEY_LOCAL_MACHINE key as described below.

  **To add registry permissions:**

  i. Run **regedit** to modify the registry.

  ii. Select the HKEY_LOCAL_MACHINE key.

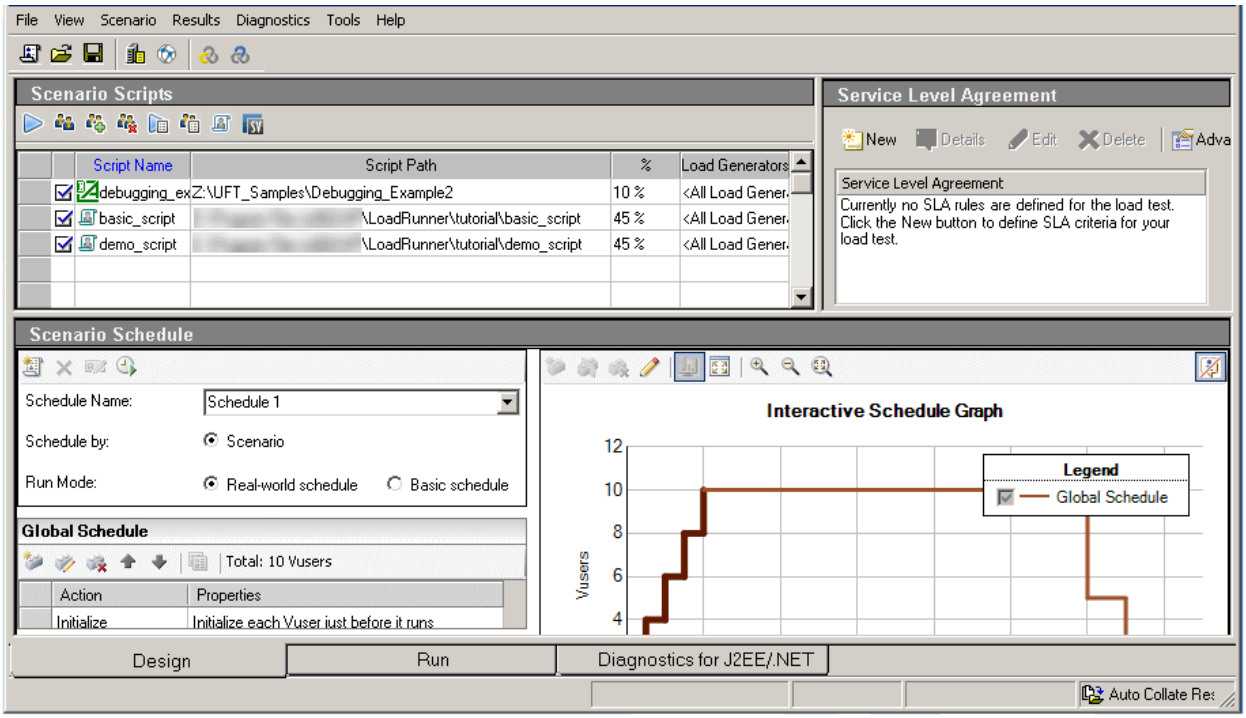
  iii. Select **Security > Permissions**.

  iv. Add **Full Control** permission to the user that is running the Controller.

  v. Click **OK**.

# Controller Window

The Controller window enables you to design and run load test scenarios, monitor their metrics, and view Diagnostics for J2EE/.NET data.




| | |
|---|---|
| **To access** | Choose one of the following:<br>• From the Windows Start menu, go to **Micro Focus > LoadRunner > Controller**<br>• The Controller shortcut on the desktop |

| | |
|---|---|
| **Important information** | By default, upon opening the Controller, the New Scenario dialog box is displayed. To disable this option, clear the **Show at Startup** option. For details, see "New Scenario Dialog Box" on page 969. |
| **Relevant tasks** | • "How to Design a Goal-Oriented Scenario" on page 951<br>• "How to Design a Manual Scenario" on page 953<br>• "How to Run a Scenario" on page 1094<br>• "How to Set Up a Monitoring Environment" on page 1171 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | **New Scenario.** Opens New Scenario dialog box. For user interface details, see "New Scenario Dialog Box" on page 969. |
| | **Open Scenario.** Enables you to open an existing scenario. |
| | **Save Scenario.** Enables you to save the active scenario.<br>By default, scenarios are saved in **%userprofile%\Documents\Controller\scenario**. |
| <br>(Goal-oriented scenario; Run view only) | **Edit scenario goal.** Opens the Edit Scenario Goal dialog box where you define goals for a goal-oriented scenario. For user interface details, see "Edit Scenario Goal Dialog Box" on page 965. |
| | **Load Generators.** Opens the Load Generators dialog box where you can add new load generators and view details about existing load generators. For user interface details, see "Load Generators Dialog Box" on page 1019. |
| | **Show Network Virtualization Settings.** When Network Virtualization is installed on the machine, opens the "Virtual Locations Settings Dialog Box" on page 1243. |
| <br>(Run view only) | **Initialize Vusers.** Initializes all Vusers (or those that are still in the **Down** state) in a selected Vuser group. The group's status changes from **Down** to **Pending** to **Initializing** to **Ready**. If the group fails to initialize, the status changes to **Error**.<br>By initializing all of the Vusers in a group before running them, you can ensure that they all begin executing the scenario at the same time. |

| | |
|---|---|
| (Run view only) | **Run Vusers Until Complete.** Runs all Vusers in a selected Vuser group until completion. If you run a Vuser group in the **Down** or **Error** state, LoadRunner initializes and then runs the group.<br>**Note:** You can instruct LoadRunner to randomly run only one Vuser in a Vuser group by right-clicking the group and selecting **Run one Vuser Until Complete**. A Vuser script log opens, displaying runtime information about the Vuser. For more information, see "Vuser Script Log" on page 1112. |
| (Run view only) | **Gradual Stop.** Gradually stops a Vuser group in the Run state if you selected the **Wait for the current iteration to end before exiting** or **Wait for the current action to end before exiting** options in the runtime settings tab of the Options dialog box. |
| (Run view only) | **Stop Vusers.** Immediately stops all Vusers in selected Vuser groups from executing their scripts. |
| (Run view only) | **Analyze Results.** Opens diagnostics results. |
| | **Invoke VuGen.** Invokes the Virtual User Generator. |
| | **Invoke Analysis.** Invokes LoadRunner Analysis. |
| **Design tab** | Enables you to design scenarios. For details, see "Design Tab" on page 962. |
| **Diagnostics for J2EE/.NET tab** | Enables you to view J2EE/.NET diagnostics data collected from a scenario run. This requires the LoadRunner J2EE/.NET Diagnostics add-in, available on the J2EE/.NET Diagnostics installation media.<br>For details, see "J2EE/.NET Diagnostics" on page 1257. |
| **Run tab** | Enables you to run and monitor scenario runs. For details, see "Run Tab" on page 1106. |
| **<Status bar>** | Displays the following features of Controller (if enabled):<br>• Application Lifecycle Management Connection<br>• IP Spoofer<br>• Auto Collate Results<br>• Auto Load Analysis |

# Designing Load Test Scenarios

This section describes how to plan and create LoadRunner Controller scenarios.

### What do you want to do?

- Plan a load test scenario
- Design a scenario
- Set up a load generator
- Provision a load generator on the cloud
- Set up a network profile
- Schedule a manual scenario
- Prepare an SLA (Service Level Agreement)

### See also:

- Managing Cloud accounts
- SSL utility
- Multiple IP Addresses
- Terminal Services
- Troubleshooting for Load Generators

# Planning Load Test Scenarios

## Load Test Planning Overview

Before running your load test, it is important to:

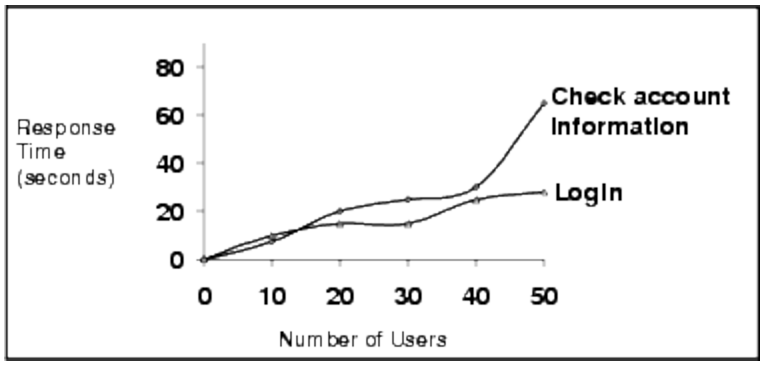
- **Understand which resources are required for testing.** Application testing requires hardware, software, and human resources. Before you begin testing, you should know which resources are available and decide how to use them effectively.
- **Define success criteria in measurable terms.** Focused testing goals and test criteria ensure successful testing. For example, it is not enough to define vague objectives like "Check server response time under heavy load." A more focused success criterion would be "Check that 50 customers can check their account balance simultaneously, and that the server response time will not exceed one minute."

## Load Testing Objectives

Your test plan should be based on a clearly defined testing objective. This section presents common application testing objectives that LoadRunner helps you test.

## Measuring End-User Response Time

Check how long it takes for the user to perform a business process and receive a response from the server. For example, suppose that you want to verify that while your system operates under normal load conditions, the end users receive responses to all requests within 20 seconds. The following graph presents a sample load vs. response time measurement for a banking application:




## Defining Optimal Hardware Configuration

Check how various system configurations (memory, CPU speed, cache, adaptors, modems) affect performance. Once you understand the system architecture and have tested the application response time, you can measure the application response for different system configurations to determine which settings provide the desired performance levels.

For example, you could set up three different server configurations and run the same tests on each configuration to measure performance variations.

## Checking Reliability

Determine the level of system stability under heavy or continuous work loads. You can use LoadRunner to create stress on the system: force the system to handle extended activity in a compressed time period to simulate the kind of activity a system would normally experience over a period of weeks or months.

## Checking Hardware or Software Upgrades

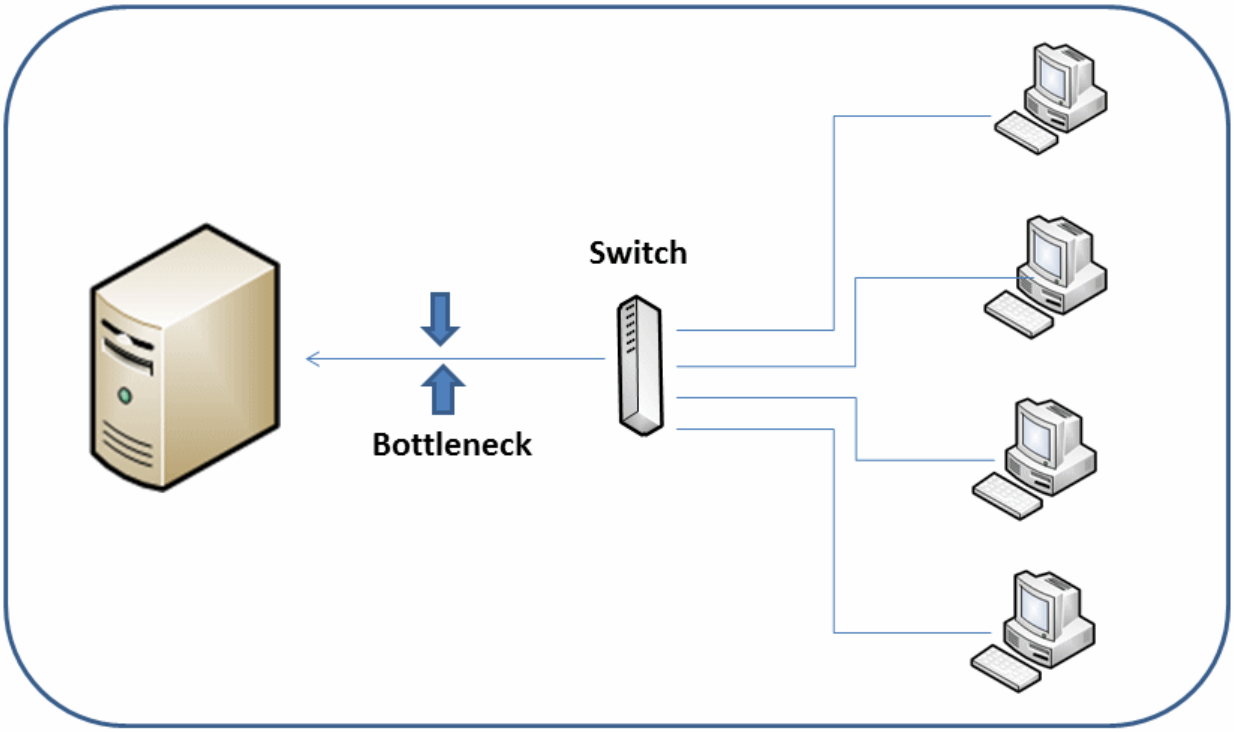
Perform regression testing to compare a new release of hardware or software to an older release. You can check how an upgrade affects response time (benchmark) and reliability. Application regression testing does not check new features of an upgrade; rather it checks that the new release is as efficient and reliable as the older release.

## Evaluating New Products

You can run tests to evaluate individual products and subsystems during the planning and design stage of a product's life cycle. For example, you can choose the hardware for the server machine or the database package based on evaluation tests.

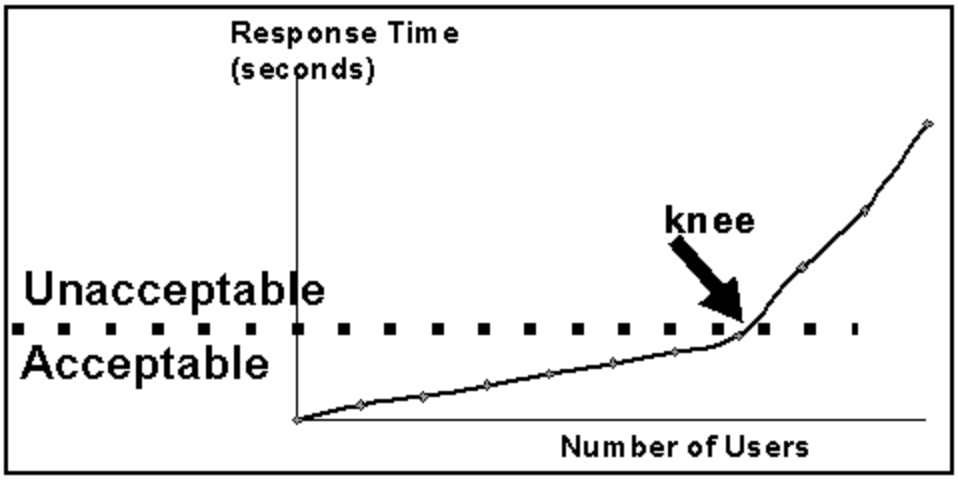
## Identifying Bottlenecks

You configure the LoadRunner monitoring components to identify bottlenecks on the system and determine which element is causing performance degradation, for example, file locking, resource contention, and network overload. Use LoadRunner in conjunction with the new network and machine monitoring tools to create load and measure performance at different points in the system.




## Measuring System Capacity

Measure system capacity, and determine how much excess capacity the system can handle without performance degradation. To check capacity, you can compare performance versus load on the existing system, and determine where significant response-time degradation begins to occur. This is often called the "knee" of the response time curve.




Once you determine the current capacity, you can decide if resources need to be increased to support additional users.

## Plan a Load Test

1. Analyze the application.

   You should become thoroughly familiar with the hardware and software components, the system configuration, and the typical usage model. This analysis ensures that the testing environment you create using LoadRunner will reflect the environment and configuration of the application under test. For task details, see "How to Analyze the Application" below.

2. Define the load testing objectives.

   Before you begin testing, you should define exactly what you want to accomplish. For task details, see "Define the Load Test Objectives" on page 941.

3. Plan LoadRunner implementation.

   Decide how to use LoadRunner to achieve your testing goals. For task details, see "Plan the LoadRunner Implementation" on page 920.

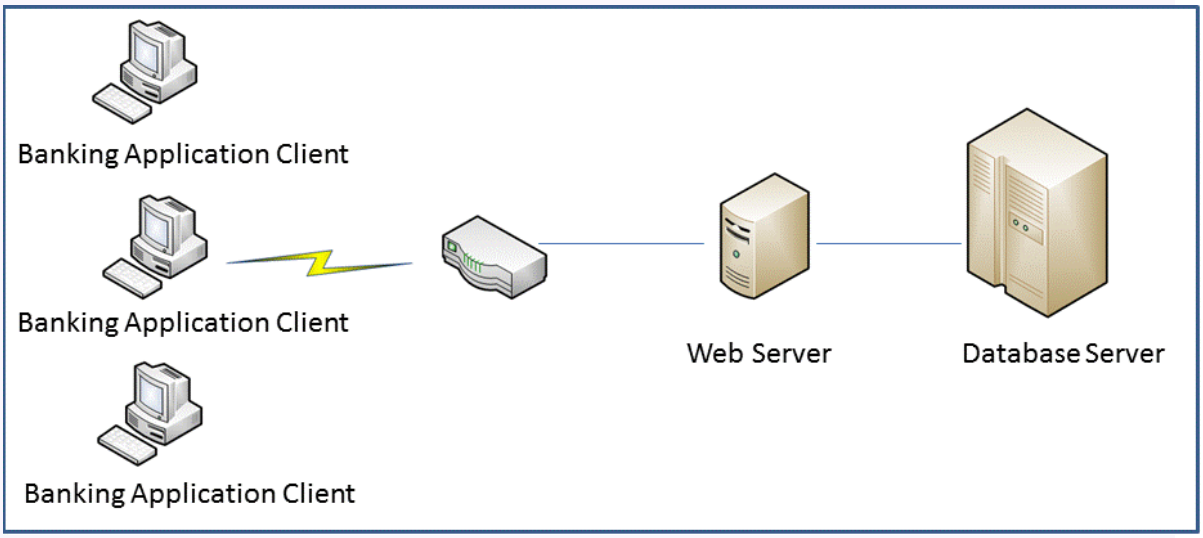
## How to Analyze the Application

This task describes how to analyze the application under test as part of the load test planning process. Each step in this task contains example information relating to an online banking system.

### Identify system components

Draw a schematic diagram to illustrate the structure of the application. If possible, extract a schematic diagram from existing documentation. If the application under test is part of a larger network system, you should identify the component of the system to be tested. Make sure the diagram includes all system components, such as client machines, network, middleware, and servers.

**Example:** The following diagram illustrates an online banking system that is accessed by many Web users. The Web users each connect to the same database to transfer funds and check balances. The customers connect to the database server through the Web, using multiple browsers.




## Describe the system configuration

Enhance the schematic diagram with more details. Describe each system component's configuration. You should be able to answer the following questions:

- How many users are anticipated to connect to the system?
- What is the application client's machine configuration (hardware, memory, operating system, software, development tool, and so forth)?
- What types of database and Web servers are used (hardware, database type, operating system, file server, and so forth)?
- How does the server communicate with the application client?
- What is the middleware configuration and application server between the front-end client and back-end server?
- What other network components may affect response time (modems and so forth)?
- What is the throughput of the communications devices? How many concurrent users can each device handle?

**Example:** The schematic diagram of the online banking system specified that there are multiple application clients accessing the system.

| Front-End Client Configuration | |
|---|---|
| Anticipated number of application clients | 50 concurrent application clients |
| Hardware / Memory | Intel Core i7-4930K @ 3.40GHz |
| Operating system & version | Windows Server 2012 64 bit |
| Client browser | Internet Explorer 10 |

### Analyze the usage mode

Define how the system is typically used and decide which functions are important to test. Consider who uses the system, the number of each type of user, and each user's common tasks. In addition, consider any background load that might affect the system response time.

**Example:** Suppose 200 employees log on to the accounting system every morning, and the same office network has a constant background load of 50 users performing various word processing and printing tasks. You could create a LoadRunner scenario with 200 virtual users signing in to the accounting database, and check the server response time.

To check how background load affects the response time, you could run your scenario on a network where you also simulate the load of employees performing word processing and printing activities.

### Examine task distribution

In addition to defining the common user tasks, examine the distribution of these tasks.

**Example:** Suppose the bank uses a central database to serve clients across many states and time zones. The 250 application clients are located in two different time zones, all connecting to the same Web server. There are 150 in Chicago and 100 in Detroit. Each begins their business day at 9:00 AM, but since they are in different time zones, there should never be more than 150 users signing in at any given time. You can analyze task distribution to determine when there is peak database activity, and which activities typically occur during **peak load** time.

## Define the Load Test Objectives

This task describes how to define the load test objectives as part of the load test planning process.

1. Decide on general objectives.

   For a list of suggested testing objectives, see "Load Testing Objectives" on page 936.

2. State the objectives in measurable terms.

   This helps you identify and focus on specific goals. To provide a baseline for evaluation, determine exactly what constitutes acceptable and unacceptable test results.

**Example:**

**General Objective.** Product Evaluation: choose hardware for the Web server.

**Focused Objective.** Product Evaluation: run the same group of 300 virtual users on two different servers. When all 300 users simultaneously browse the pages of your Web application, determine which hardware gives a better response time.

# Designing Scenarios

Select an image to learn more.

## Manual Scenarios

You build a manual scenario by selecting scripts to run, assigning load generators on which to run the scripts, and distributing Vusers to run among the scripts.

You can design a manual scenario in one of the following modes:

- **Vuser group mode.** In this mode, each script you select for the scenario is assigned to a Vuser group. You assign a number of Vusers to each Vuser group that you create. You can instruct all Vusers in a group to run the same script on the same load generator, or you can assign different scripts and load generators to the various Vusers in a group.
- **Percentage mode.** In this mode, you define a total number of Vusers to be used in the scenario, and assign load generators and a percentage of the total number of Vusers to each script.

After you define which Vuser groups/scripts to run in the scenario, you select or build a **schedule** by which to run the scenario. For more information, see "Scheduling Manual Scenarios" on page 1034.

You can also create **Service Level Agreements (SLAs)** which are specific goals that you define for your load test scenario. When you run the scenario, LoadRunner gathers and stores performance-related data. When you analyze the run, Analysis compares this data against the SLAs and determines SLA statuses for the defined measurements. For more information, see "Service Level Agreements" on page 1048.

### Changing Scenario Modes

You can convert a scenario from the Vuser group mode to the percentage mode and vice versa.

The following table describes what happens to the scenario when converting from the one mode to the other:

| | |
|---|---|
| **Vuser group mode to percentage mode** | • If a Vuser group contains multiple scripts, in percentage mode the scripts are listed one by one in the Scenario Scripts pane.<br>• In the percentage mode, all load generators are assigned to all Vuser scripts by default. If multiple load generators are assigned to a Vuser group, the Vusers assigned to the scripts in the percentage mode are distributed evenly among the load generators originally assigned to the group.<br>If you defined group schedules for the Vuser groups, these settings will be lost. All profiles will contain schedule **by scenario** settings only. For details about scheduling scenarios, see "Scheduling Manual Scenarios" on page 1034. |
| **Percentage mode to Vuser group mode** | • Each script is converted to a Vuser group.<br>• If you defined multiple load generators for a Vuser script, the Vuser group that is created when converting the scenario will also contain multiple load generators.<br>• If a schedule is defined for the scenario, all the schedule settings remain unchanged. |

**Note:** You can convert from one scenario mode to another at any time. For details, see "How to Change the Scenario Mode (Manual Scenario)" on page 954.

## Goals Types for Goal-Oriented Scenarios

In a goal-oriented scenario, you define the goals you want your test to achieve and LoadRunner automatically builds a scenario for you based on these goals.

This section discusses the types of goals for a goal-oriented scenario.

### Virtual Users

This goal tests if your application can run a specified number of Vusers simultaneously. Running this type of goal-oriented scenario is similar to running a manual scenario.

### Pages per Minute/Hits per Second/Transactions per Second

These goals test the strength of your server. For each of these goal types, you specify a minimum-maximum range of Vusers for the scenario to run, and in the case of the Transactions per Second goal type, you also specify a transaction name.

**Note:**
- **Pages per Minute** and **Hits per Second** goals are for Web Vusers only.
- Hits per second relates to HTTP requests per second.

When you define one of these goal type, Controller divides the target defined by the minimum number of Vusers specified, and determines the target number of hits/transactions per second or pages per minute that each Vuser should reach.

Controller then begins loading the Vusers according to the load behavior settings you defined, as follows:

- If you selected to run the Vusers automatically, LoadRunner loads 50 Vusers in the first batch. If the maximum number of Vusers defined is less than 50, LoadRunner loads all of the Vusers simultaneously.
- If you chose to reach your target after a certain period of the scenario elapses, LoadRunner attempts to reach the defined target within this period of time. It determines the size of the first batch of Vusers based on the time limit you defined and the calculated target number of hits, transactions, or pages per Vuser.
- If you chose to reach your target by gradation (x number of pages/hits every x amount of time), LoadRunner calculates the target number of hits or pages per Vuser and determines the size of the first batch of Vusers accordingly. (Not relevant for the Transactions per Second goal type).

After running each batch of Vusers, LoadRunner evaluates whether the target for the batch was achieved. If the batch target was not reached, LoadRunner recalculates the target number of hits, transactions, or pages per Vuser, and readjusts the number of Vusers for the next batch to be able to achieve the defined goal. By default, a new batch of Vusers is released every two minutes.

If the goal has not been reached after Controller has launched the maximum number of Vusers, LoadRunner attempts to reach the defined target once more by recalculating the target number of hits, transactions, or pages per Vuser, and running the maximum number of Vusers simultaneously.

A Pages per Minute or Hits/Transactions per Second goal-oriented scenario is assigned a **Failed** status if:

- Controller has twice attempted to reach the goal using the maximum number of Vusers specified, and the goal could not be reached.
- No pages per minute or hits/transactions per second were registered after the first batch of Vusers was run.
- The number of pages per minute or hits/transactions per second did not increase after Controller ran a certain number of Vuser batches.
- All the Vusers that ran failed.
- There were no available load generators for the type of Vusers you attempted to run.

## Transaction Response Time

This goal tests how many Vusers can be run simultaneously without exceeding a desired transaction response time. You specify the name of the transaction in your script that you want to test, and a minimum-maximum range of Vusers for LoadRunner to run. The transaction response time you specify should be a predefined threshold value.

For example, if you do not want a customer to wait more than five seconds to log in to your e-commerce site, specify a maximum acceptable transaction response time of five seconds. Set the minimum and

maximum number of Vusers to the minimum-maximum range of customers you want to be able to serve simultaneously.

If the scenario does not reach the maximum transaction response time that you defined, your server is capable of responding within a reasonable period of time to the number of customers you want to be able to serve simultaneously. If the defined response time is reached after only a portion of the Vusers has been executed, or if you receive a message that the defined response time will be exceeded if Controller uses the maximum number of Vusers defined, you should consider revamping your application and/or upgrading your server software and hardware.

**Note:**
- To achieve a Transactions per Second or Transaction Response Time goal, your script must contain transactions. For each of these goal types, you define the transaction in the script that you want to test.
- For a Transaction Response Time goal-oriented scenario to be effective, you must choose your transaction carefully, ensuring that it performs effective hits on the server.

# JMeter Tests

This topic provides an overview of running Apache JMeter tests from Controller.

**Note:** This feature is released as a beta version.

## JMeter tests overview

Apache JMeter is an open source load testing tool for analyzing and measuring the performance of a variety of services, with a focus on web applications. If you work with Apache JMeter tests, LoadRunner provides the ability to also run the tests through Controller.

By including JMeter (.jmx) scripts in your scenarios, you can run one or more JMeter tests side-by-side with any other LoadRunner protocol scenario, giving you a single entry point for executing your performance tests. As with other LoadRunner scripts, you can run your JMeter test on a local host LoadRunner machine or on a remote load generator, and on a Windows or Linux operating system.

When you run your JMeter tests from Controller, in addition to collecting JMeter test results, LoadRunner uses the Backend Listener for JMeter to collect data for LoadRunner measurements.

Measurements can be viewed online and offline (via Controller and Analysis), using the data points from the JMeter tests.

**Important:** JMeter is released as a "beta support" feature; currently you do not need a license to run JMeter tests in Controller. However, this may change for future versions.

## Using JMeter tests in a scenario

JMeter tests work with threads, JMeter's equivalent of Vusers, and thread groups. LoadRunner handles the JMeter tests somewhat differently from a Vuser script:

- When one Vuser runs a single .jmx file, it actually runs all the threads inside each of the thread groups in the file.
- The full .jmx test runs for each LoadRunner Vuser, so one .jmx file effectively corresponds to a full Controller scenario.
- The entire .jmx test is executed on each load generator. So, for example, if a JMeter script is running 100 threads, and your scenario is running 1 Vuser on three load generators, then each JMeter test runs 100 threads on **each** of the three load generators, and 300 threads are run in total.

## Using JMeter remote/distributed testing

LoadRunner supports JMeter distributed testing, where a JMeter client instance controls multiple, remote JMeter server (test node) instances.

Each remote node executes the testing plan in full. The load generator collects all the test results, received from the remote machines, from the JMeter client.

A scenario can contain JMeter scripts for remote testing, alongside scripts that do not support remote testing. You define remote testing support in the JMeter runtime settings.

JMeter and Java must be installed and configured on each remote node machine, in addition to on the load generator machine. You must also define a properties file containing the names (or IPs) for the remote node machines. For details, see "Set Up the JMeter Test" below.

> **Note:** When using remote testing, the graphs display the JMeter data as a single line for each load generator.

### See also:

- "Set Up the JMeter Test" below
- "JMeter Best Practices and Troubleshooting" on page 948

## Set Up the JMeter Test

This topic describes how to prepare your environment for running JMeter tests, and to define a JMeter test scenario.

We recommend that you check the best practices and limitations information before you begin set up.

Prerequisite setup for JMeter tests

### Install the following on each load generator and machine that will run a JMeter test:

- **Apache JMeter.** For supported versions of JMeter, see the System Requirements.
- **Java.** This should be the appropriate version for the installed JMeter version.

### Load generator setup:

- **Java setting for Windows:** Check that the environment variable **%PATH%** includes the directory of jvm.dll (for example: %JAVA_HOME%/jdk/jre/bin/server).
- **Java setting for Linux:** Check that the environment variable **$LD_LIBRARY_PATH** includes the directory of libjvm.so.
- We highly recommended that you define the following:
  - Set the **JAVA_HOME** environment variable to point to the Java JDK folder.
  - Set the **JMETER_HOME** environment variable to point to the JMeter folder (containing JMeter sub-folders: bin, lib, etc.).

### LoadRunner agent settings:

- If the agent is set as **Service**, change the **System** environment variables JMETER_HOME, JAVA_HOME, and PATH , and restart the load generator machine.
- If the agent is set as **Process**, change any **User** or **System** environment variables JMETER_HOME, JAVA_HOME, and PATH, and restart the agent.

## Add a JMeter test to a scenario

This task describes how to create a scenario containing a JMeter script, and define Runtime Settings and actions for the script.

### To define a JMeter scenario:

1. Make sure the load generator machines are set up to run JMeter tests, as described in "Prerequisite setup for JMeter tests" above.
2. On the main Controller toolbar, click the **New Scenario** button .
3. In the New Scenario dialog box, in the **Select the scripts...** area, select the **JMeter Scripts** radio button.
4. Click **Browse** and select the JMeter script (.jmx file).
5. Click **OK** in the New Scenario dialog box. The scenario containing the JMeter script opens in the Design tab.
6. Right-click the script and open the Runtime Settings dialog box, and configure the following:

- **Start measurements.** Make sure this check box is selected (default setting).
- **JMeter path.** Define the home path for the JMeter installation on the load generator. If the field is left empty, it will take the default path from the JMETER_HOME environment variable. (For more information on setting the default environment variable, see "Prerequisite setup for JMeter tests" on the previous page.)
- **JMeter port.** Leave it set to **default** to use the internal port range (4000 – 65535), or define a **range**.
- **Remote testing.** Select the check box to run the test on multiple remote JMeter machines. For details, see "Using JMeter remote/distributed testing" on page 946.
- **JMeter additional properties file.** This field is required if you are select **Remote testing**. Define the path to the additional properties file, listing the remote machines. For details on creating this properties file, see "JMeter additional properties file" below.

7. Open the Edit Action dialog box (described in see "Edit Action Dialog Box" on page 1043), and set the following:
   - For action **Start Vusers**, set to start **Simultaneously**.
   - For action **Duration**, it is recommended to set to **Run until completion**.

### JMeter additional properties file

When using JMeter distributed testing across multiple, remote machines, you must define a properties file containing the names of the remote hosts.

## To define the properties file:

1. Create a file named **additionalJmeter.properties**.
2. Inside the file, add the name or IP address of each remote host, separated by commas, using the format **remote_hosts=<server name>**.

   For example: **remote_hosts=server1,server2,ip_address3**
3. Save the **additionalJmeter.properties** file at a location accessible to the load generator, and make sure the correct path to the file is defined in the JMeter runtime settings.

## See also:

- "JMeter Tests" on page 945
- "JMeter Best Practices and Troubleshooting" below
- "JMeter Monitoring Graphs" on page 1191
- "JMeter Analysis Graphs" on page 1446

### JMeter Best Practices and Troubleshooting

This topic describes best practices and limitations when working with JMeter, and how to troubleshoot issues that may occur.

### Graceful shutdown

Generally, JMeter supports shutting down gracefully. This requires appropriate port configuration, and to set the **Run until completion** option.

If there is a timeout by Controller, JMeter will receive the shutdown gracefully command and stay there for 2 minutes.

Click **Stop Now** in the Run tab if you want to stop the scenario aggressively.

### Best practices

The following are recommended as best practices when working with JMeter tests:

- When running JMeter tests, it is recommended that you use dedicated load generator machines. Running tests from the Controller, using the **localhost** load generator, may cause some instability.
- It is recommended that you use scenario mode (Vuser group mode) when running JMeter scripts. If you do use percentage mode, note how many instances of the JMeter test you will be executing.
- Before running LoadRunner scenarios, check that your JMeter test can run with the JMeter installation on the load generator.
- Run a maximum of one .jmx test (using 1 Vuser - default value) per each load generator. If the .jmx test is running a lot of threads, more than one test may overload the system.
- Let the scripts run to completion.

### Limitations

Note the following when working with JMeter in LoadRunner:

- Network Virtualization is not supported for JMeter tests.
- If using percentage mode for a JMeter scenario, note that only one load generator can be assigned from the list.

### Troubleshooting JMeter tests

If you have any difficulty running JMeter tests, check if the following can resolve the issue.

## Error: Cannot find the JMX file 'C:\jmeter_tests\test.jmx‘.

Possible causes:

- The file is not present in this location.
- Not enough disk space to clone the file.

## Error: The JMETER_HOME environment variable is not configured.

**Resolution:** Define the JMETER_HOME environment variable, or add the **JMeter path** in Runtime Settings.

### Error pattern: Message contains “Problem loading XML from: ‘.....’ missing class ....

**Resolution:** The JMeter test may contain 3rd-party plugins, and the plugins are missing in the currently used JMeter instance. Either remove the plugins from the JMeter test file, or add the plugins to the JMeter instance.

### Error: Unable to bind to free port (for Shutdown/StopTestNow) in range 4445 - 4455. Please extend the port range in the JMeter Runtime Settings.

**Resolution:** In Runtime Settings, extend the **JMeter port** range, or change to **default**.

### There is no data in JMeter graphs in Controller or Analysis.

**Resolution:** In Runtime Settings, make sure that the **Start measurement** check box is selected.

### Errors: Failed to create a Java virtual machine (JVM); Failed to load the 'jvm.dll/libjvm.so’ library file.

**Resolution:** Check that Java is properly installed on the machine.

- For Windows: Check that the folder that contains jvm.dll is included.
- For Linux: Check that LD_LIBRARY_PATH includes directory of libjvm.so.

### See also:

- "JMeter Tests" on page 945
- "JMeter Best Practices and Troubleshooting" on page 948

## Noise Scenarios

You can approach Web performance testing in the following ways:

- Create a load test that runs complex Vuser scripts. These scripts perform a business process and contain transactions, complex flows, checkpoints, and so forth.
- Create a load on the server by having a large number of users (real or virtual) access the same URL simultaneously. This is commonly known as **noise testing**.

The first approach uses a standard Vuser script generated with VuGen or through a DevOp addin. The script performs the full business process and gathers the metrics. After the test run, you can retrieve meaningful information from the Analysis graphs and reports.

The second approach, **noise testing**, only allows you to determine the response times and whether the server can handle the load without crashing.

The LoadRunner Controller allows you to set up both types of scenarios. You can create a single scenario that contains both standard and noise generator type Vusers.

You set up a **noise generator** scenario from the Add Script dialog box. You select a noise generator type of script, and specify the URL of the server you want to access. During the scenario run, these Vusers access the URL simultaneously.

You cannot edit **Noise Generator** scripts in VuGen.

For user interface details, see the Add Script Dialog Box or Add Group Dialog Box.

## How to Design a Goal-Oriented Scenario

This task describes how to design a goal-oriented scenario. In this type of scenario, you define the goals you want your test to achieve and LoadRunner automatically builds a scenario for you based on these goals.

### 1. Prerequisites

- Before setting up the scenario, decide which goal you want the scenario to reach. For details on types of scenario goals, see "Goals Types for Goal-Oriented Scenarios" on page 943.
- Before you start designing the scenario, record the VuGen scripts that will run in the scenario. For details, see "Record a Vuser Script" on page 158.

### 2. Open a new goal-oriented scenario

1. On the Controller toolbar, click the **New Scenario** button .
2. In the New Scenario dialog box that opens, select **Goal-oriented Scenario**.
3. Select scripts to run in the scenario. Select scripts in the **Available Scripts** box, and click **Add** to move them to the **Scripts in Scenario** box.

> **Note:** If a script was created in a version of VuGen or TruClient that is later than the Controller version, the script may not run. In this case you may be prompted whether to allow the script to run. A notice will be put in the Load Generator log.

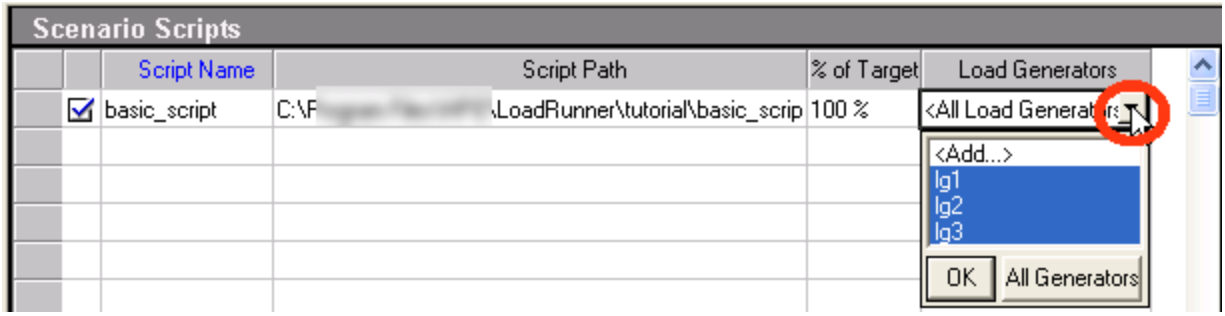
When you click **OK**, the Design tab opens and displays the new scenario.

### 3. Add load generators to the scenario

Click the **Load Generators** button . In the Load Generators dialog box that opens, click **Add** and enter the details of the load generator you are adding. For details about the Add Load Generator dialog box, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007.

### 4. Assign load generators to each script

In the Scenario Scripts pane, for each script, click the **Load Generators** column and select a load generator on which to run the script.




**Note:** By default, the script will run on all the load generators in the scenario.

## 5. Define a goal for the scenario

In the Scenario Goal pane, click the **Edit Scenario Goal** button. In the dialog box that opens, define the goal the scenario should reach. For details about filling in the scenario goal details, see "Edit Scenario Goal Dialog Box" on page 965.

## 6. Define a virtual location for each script in the scenario - optional

In the Scenario Scripts pane's **Virtual Location** column, select the location for the network virtualization. This only applies if you have Network Virtualization installed. For details, see " Network Virtualization Locations" on page 1238.

## 7. Assign each script a percentage of the total scenario target

In the Scenario Scripts pane's **% of Target** column, enter the percentage of the total goal you want each script to reach during the scenario.

**Note:** Assign percentages to the scripts starting with the first script in the list and moving down the list.

## 8. Define virtual services for the scenario - optional

If you have Service Virtualization installed, you can add virtual services in place of actual services. For details, see "Service Virtualization Integration" on page 1247.

## 9. Define service level agreements for the scenario - optional

You can define service level agreements (SLAs) to measure scenario goals over time intervals, or over a whole scenario run. When you later analyze the run using LoadRunner Analysis, this data is compared against the SLAs and SLA statuses are determined for the defined measurements. To define SLAs, see "Define Service Level Agreements" on page 1301.

# How to Design a Manual Scenario

This task describes how to design a manual scenario.

## 1. Prerequisites

- When designing a manual scenario, plan how you want to distribute the Vusers in the scenario. For more details, see "Manual Scenarios" on page 942.
- Before you start designing the scenario, record the VuGen scripts that will run in the scenario. For details, see "Record a Vuser Script" on page 158.

## 2. Open a scenario, or create a new one

1. On the main Controller toolbar, click the **New Scenario** button .
2. In the New Scenario dialog box, select **Manual Scenario**.
3. (Optional) To distribute the Vusers by percentage, select the **Use the Percentage mode...** option.

> **Note:** You can convert from one scenario mode to another at any time. For details, "How to Change the Scenario Mode (Manual Scenario)" on the next page.

4. (Optional) Select scripts to participate in the scenario. If you do not select the scripts here, you can select them later on.

> **Note:** If a script was created in a version of VuGen or TruClient that is later than the Controller version, the script may not run. In this case you may be prompted whether to allow the script to run. A notice will be put in the Load Generator log.

When you click **OK**, the scenario opens in the Design tab.

## 3. Add load generators to the scenario

Click the **Load Generators** button . In the Load Generators dialog box that opens, click **Add** and enter the details of the load generator you are adding. For details about adding load generators, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007.

## 4. Add Vuser groups/scripts to the scenario - Vuser group mode

1. In the **Scenario Groups** pane, click the **Add Group** button .
2. In the Add Group dialog box:
   - Give the group a name and assign a number of Vusers to the group.
   - Select a load generator on which to run the Vusers.

- Select a Vuser script.

### 5. Add Vuser groups/scripts to the scenario - Percentage mode

1. Click the **Add Group** button and select a Vuser script from the list. Repeat this step for all scripts that you want to include inthe test.
2. In the **Scenario Scripts** pane's **Load Generator** column, select load generators on which to run the scripts.
3. In the **Scenario Scripts** pane's **%** column, assign a percentage of the total number of Vusers for each script. Assign percentages to the scripts starting with the first script in the table and moving down the list.

### 6. Define a virtual location for the scenario - optional

If you have Network Virtualization installed, click in the **Virtual Location** column, and select a location. For details, see " Network Virtualization Locations" on page 1238.

### 7. Define a schedule for the scenario

Define a schedule by which to run the Vusers in the scenario. For details, see "Define a Schedule for the Scenario" on page 1039.

### 8. Define virtual services for the scenario - optional

If you have Service Virtualization installed, you can add virtual services in place of actual services. For details, see "Service Virtualization Integration" on page 1247.

### 9. Define service level agreements for the scenario - optional

You can define service level agreements (SLAs) to measure scenario goals over time intervals, or over a whole scenario run. When you later analyze the run using LoadRunner Analysis, this data is compared against the SLAs and SLA statuses are determined for the defined measurements. To define SLAs, see "Define Service Level Agreements" on page 1301.

## How to Change the Scenario Mode (Manual Scenario)

This task describes how to change a manual scenario from Vuser group mode to percentage mode, and vice versa.

For details about the scenario modes and the effects of changing from one to another, see "Manual Scenarios" on page 942.

- To convert the scenario from Vuser group mode to percentage mode, select **Scenario > Convert Scenario to the Percentage Mode**.
- To convert the scenario from percentage mode to Vuser group mode, select **Scenario > Convert**

**Scenario to the Vuser Group Mode**.

> **Note:** By default, every time you convert from one mode to another, a message appears warning you that scenario and schedule settings may change. To show/hide this warning message, select **Scenario > Show Convert Scenario Mode Warning**.

## How to View/Modify Scripts in the Scenario

This section describes how to view and modify scripts used in your load test scenario.

You view/modify the details of the scripts in the Group Information dialog box (see "Group Information Dialog Box" on page 967) or in the Script Information dialog box (see "Script Information Dialog Box" on page 978).

### View script details

You can view the details of a script by right-clicking the script in the Scenario Groups/Scripts pane and selecting **Details**.

In the Group/Script Information dialog box that opens, you can:

- View details about the script, including:

> **Note:** If you do not see some of the details listed below, click **More**.

- Script path
- Command line options
- Rendezvous points included in the script
- Vusers associated with the script
- Files associated with the script
- Open the script in VuGen by clicking the **View Script** button
- View the script's runtime settings by clicking the **Runtime Settings** button

### Modify a script's runtime settings

- To view or modify a script's runtime settings, in the Scenario Groups/Scripts pane right-click the script and select **Runtime Settings**.
- To view or modify runtime settings of a script associated with a particular Vuser, in the Vusers dialog box (Scenario Groups pane > **Vusers** ) right-click the Vuser and select **Runtime Settings**.

Modifying the runtime settings for one Vuser in a group modifies the runtime settings for all the Vusers in that group that are using the same script.

## Modify multiple scripts' runtime settings

This section describes how to modify runtime settings of multiple scripts or of a Vuser group that includes multiple scripts.

1. In the Scenario Groups/Scripts pane select multiple scripts or the Vuser group that includes multiple scripts.
2. Right-click the selection and select **Runtime Settings**.
3. In the Multiple runtime settings Mode dialog box that opens:
   - To modify runtime settings for all of the scripts simultaneously, click **Shared RTS**.
   - To modify runtime settings per script, click **Individual RTS**.

     For user interface details, see "Multiple Runtime Settings Mode Dialog Box" on page 968.
   - For details about specific runtime settings, see "Runtime Settings" on page 301.
   - When you modify the runtime settings from the Controller, LoadRunner runs the script using the modified settings.

## View/Edit a script in VuGen

To view/edit a script included in your scenario, right-click the script and select **View Script**. The script opens in VuGen. For more information on editing scripts, see the "Debugging Overview" on page 328.

## Specify command line options

You can specify command line options to use when running a script.

1. In the Scenario Groups/Scripts pane, right-click the script and select **Details**.
2. In the Group/Script Information dialog box that opens, if **Command line** is not displayed near the bottom, click **More**.
3. Enter a command in the command line, for example: `-x value -y value.`

   For information about passing command line argument values to a script, see "Enhance a Java Script" on page 555.

## View rendezvous points included in the script

1. In the Scenario Groups/Scripts pane, right-click the script and select **Details**.
2. In the Group/Script Information dialog box, if the **Rendezvous** tab is not displayed near the bottom, click **More**.

   If there are rendezvous points included in the script, they are displayed in the Rendezvous tab. For details about rendezvous points, see "Rendezvous Points Overview" on page 1090.

## View Vusers associated with the script

1. In the Scenario Groups/Scripts pane, right-click the script and select **Details**.
2. In the Group/Script Information dialog box, if the **Vusers** tab is not displayed near the bottom, click

**More**.

The Vusers tab displays the Vusers associated with the script.

### View files associated with the script

1. In the Scenario Groups/Scripts pane, right-click the script and select **Details**.
2. In the Group/Script Information dialog box, if the **Files** tab is not displayed near the bottom, click **More**.

By default, the Files tab lists all the files in the script's folder (only after your script has been added to the script list). These files include the configuration settings file, the init, run, and end portions of the script, the parameterization definitions file, and the **.usr** file. To add a file to the list, click **Add**.

**Example:**

To run Visual C++ Vusers on a remote load generator, you must add the **.dll** of the Vuser to the list of files.

You can delete the files that you add, but not the other files listed.

## Relative Paths for Scripts

You can specify a relative location for a script in your scenario. The location can be relative to the current scenario folder, or relative to the LoadRunner installation folder.

When you run a scenario, the script is automatically copied from this relative location to a temporary folder on the load generator running the script. This enables the load generator to access the script locally instead of over a network.

To specify a path relative to the current scenario director, type either of the following notations at the start of the script path:

| Notation | Description |
|---|---|
| ./ | Indicates that the path is relative to the location of the scenario folder |
| ../ | Indicates that the path is relative to the location of the parent folder of the scenario folder |

For example, if the current scenario is located at `F:\scenarios`, to specify that the script, **user1**, is located in `F:\scenarios\scripts`, you could type:

```
./scripts/user1
```

To specify a path relative to the LoadRunner installation folder (**%lr_path%**), type a percent sign (**%**) at the beginning of the script path.

For example, if the LoadRunner installation folder is located at `C:\LoadRunner`, to specify that the script, **user1**, is located `in C:\LoadRunner\scripts`, you could type:

```
%\scripts\user1

%scripts\user1
```

> **Note:** When specifying a relative path, you can include standard DOS notation (.\ and ..\) inside the path, as shown in the following example: `M:\LR\my_tests\..\..\test.usr`.

## Vuser Statuses

The following table describes the possible statuses of Vusers before, during, and after a scenario run.

| Status | Description |
|---|---|
| **Down** | The Vuser is down. |
| **Pending** | The Vuser is ready to be initialized and is waiting for an available load generator, or is transferring files to the load generator. The Vuser will run when the conditions set in its scheduling attributes are met. |
| **Initializing** | The Vuser is being initialized on the remote machine. |
| **Ready** | The Vuser already performed the init section of the script and is ready to run. |
| **Running** | The Vuser is running. The Vuser script is being executed on a load generator. |
| **Rendezvous** | The Vuser has arrived at the rendezvous point and is waiting to be released by LoadRunner. |
| **Done.Passed** | The Vuser has finished running. The script passed. |
| **Done.Failed** | The Vuser has finished running. The script failed. |
| **Error** | A problem occurred with the Vuser. Check the Status field on the Vuser dialog box or the output window for a complete explanation of the error. |
| **Gradual Exiting** | The Vuser is completing the iteration or action it is running (as defined in **Tools > Options > Runtime Settings**) before exiting. |
| **Exiting** | The Vuser has finished running or has been stopped, and is now exiting. |
| **Stopped** | The Vuser stopped when the Stop command was invoked. |

## Add Group Dialog Box

This dialog box enables you to add Vuser groups to participate in a scenario.

| | |
|---|---|
| **To access** | Manual scenario > **Design** tab > **Scenario Groups/Scripts** pane > **Add Group** |
| **Important information** | While a scenario is running, you can add Vuser groups to the scenario and enable them. However, if you add a Vuser group after all the Vusers in the scenario have started running, the new group will not run in the scenario. |
| **Relevant tasks** | "How to Design a Manual Scenario" on page 953 |
| **See also** | "How to Design a Manual Scenario" on page 953 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Enables you to add Vuser or JMeter scripts to the list of scripts. |
| | Opens VuGen where you can record a Vuser script. For more information on recording Vuser scripts, see "Record a Vuser Script" on page 158. |
| **Group Name** | The name of the Vuser group.<br>When you select a script the Vuser group is automatically given the same name as the script. You can modify the group name.<br>**Note:** The name is limited to a maximum of 55 characters. |
| **Vuser Quantity** | The number of Vusers to add to the group. The default value is 1. |
| **Load Generator Name** | The load generator assigned to the Vuser group.<br>To add a load generator to this list, select **Add** from the list. For user interface details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |
| **Use existing LoadRunner Script** | Lists the available scripts that have been added to the scenario. When you select a script, its name and path are displayed above the list.<br>To display the scripts with their full paths, right-click the list area and select **Show Paths**.<br>**Note:** If a script uses Unique file parameterization, running more than one Vuser group with that script in the same scenario may cause unexpected scenario results. For more information about Unique file parameterization, see "Data Assignment Methods for File-Type Parameters" on page 365. |

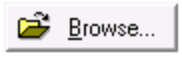
| | |
|---|---|
| **Use a Noise Generator script with the following URL** | The URL to which you want to apply noise testing. You can provide a server name, IP address, or a full URL. After a URL is used once, it appears in a dropdown list.<br><br>LoadRunner automatically generates a script name using the following format: **Noise_<domain>_<index>**, e.g. Noise_mydomain_1.<br><br>**Note:** There is no built-in validation, so you must verify that the URL you provide is operational. |

## Add Script Dialog Box

This dialog box enables you to add Vuser scripts to a scenario.

| | |
|---|---|
| **To access** | Use one of the following:<br>• All scenarios: **Design** tab > Right-click in **Scenario Scripts** pane > **Add Script**<br>• Goal-oriented scenario: **Design** tab > **Scenario Scripts** pane > **Add Script**<br>• Manual scenario (percentage mode): **Design** tab > **Add Group** |
| **Important information** | While a scenario is running, you can add scripts to the scenario and enable them. However, if you add a script after all the Vusers in the scenario have started running, the added script will not run in the scenario. |
| **Relevant tasks** | • "How to Design a Manual Scenario" on page 953<br>• "How to Design a Goal-Oriented Scenario" on page 951 |
| **See Also** | • Noise Generators |

User interface elements are described below:

| UI Element | Description |
|---|---|
|  | Enables you to add Vuser scripts to the list of scripts. |
| | Opens VuGen for recording a new Vuser script. For more information, see "Record a Vuser Script" on page 158. |
| **Use existing LoadRunner script** | Lists the available scripts that have been added to the scenario. When you select a script, its name and path are displayed above the list.<br>To display the scripts with their full paths, right-click the list area and select **Show Paths**. |

| UI Element | Description |
|---|---|
| **Use a Noise Generator script with the following URL** | The URL to which you want to apply noise testing. You can provide a server name, IP address, or a full URL. After a URL is used once, it appears in a dropdown list.<br><br>LoadRunner automatically generates a script name using the following format: **Noise_<domain>_<index>**, e.g. Noise_mydomain_1.<br><br>**Note:** There is no built-in validation, so you must verify that the URL you provide is operational. |

## Add Vusers Dialog Box

This dialog box enables you to add Vusers to a Vuser group.

| | |
|---|---|
| **To access** | **Design** tab > **Scenario Groups** pane > **Vusers** > **Add Vusers** |
| **Important information** | • Relevant for manual scenarios in Vuser group mode only.<br>• Available when a group is selected in the Scenario Groups pane. |
| **See also** | "Run/Stop Vusers Dialog Box" on page 1108 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Browse... | Enables you to add Vuser scripts to the list of scripts. |
| Record... | Opens VuGen where you can record a Vuser script. For more information on recording Vuser scripts, see "Record a Vuser Script" on page 158. |
| Run-Time Settings... | Opens the runtime settings dialog box, where you can edit the script's runtime settings.<br>When you modify the runtime settings from the Controller, LoadRunner runs the script using the modified settings. |
| Parameter list... | Opens the Parameter list in VuGen where you can create, view, modify, and delete script parameters. For details, see "Parameterization Overview" on page 357. |
| **Group Name** | The name of the group to which to add Vusers. |

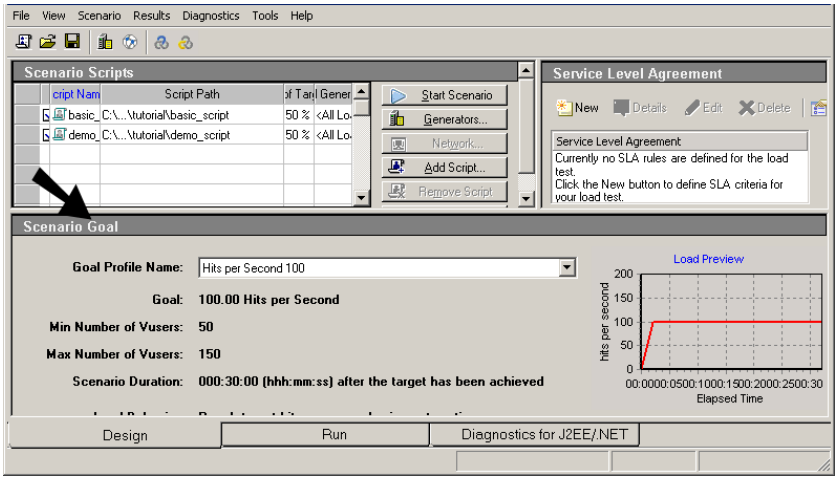
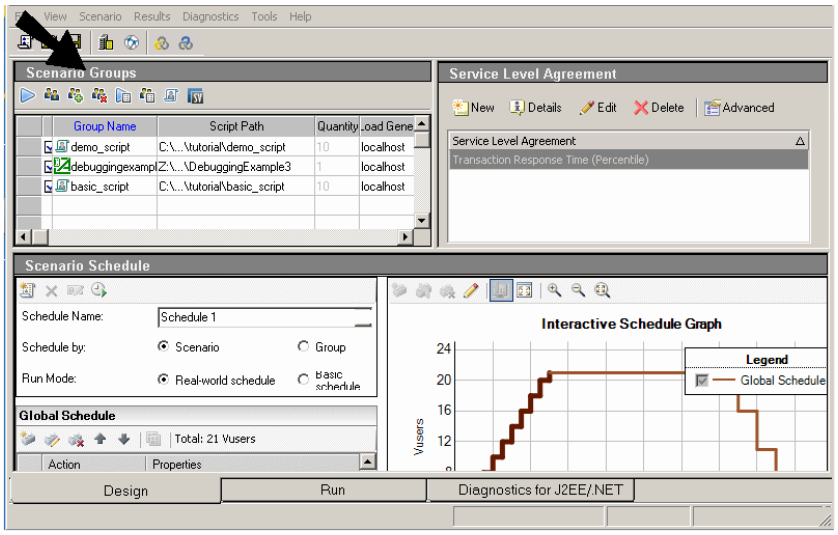
| UI Element | Description |
|---|---|
| **Load Generator Name** | The load generator assigned to the Vuser group.<br>To add a load generator to this list, select **Add** from the list. For user interface details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |
| **Quantity to add** | The number of Vusers to add to the Vuser group. |
| **Select Script** | Lists the scripts available for the scenario. When you select a script, its name and path are displayed above the list.<br>To display the scripts with their full paths, right-click the list area and select **Show Paths**. |

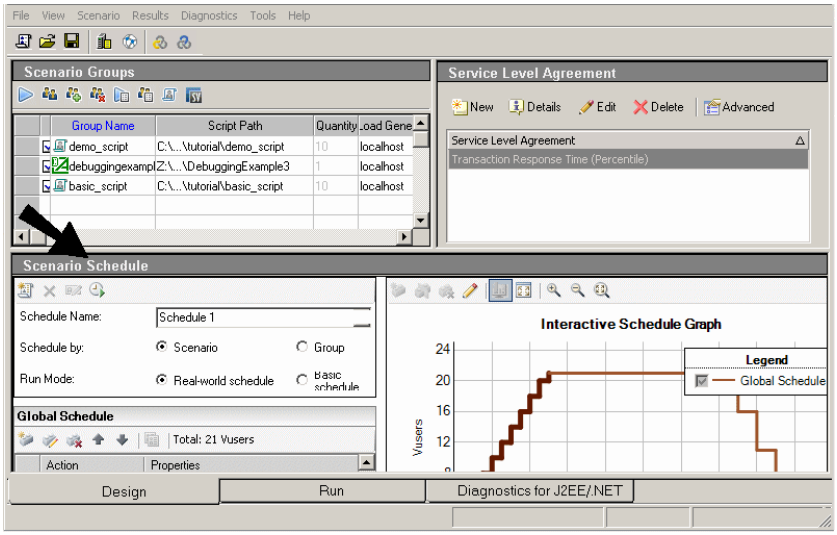
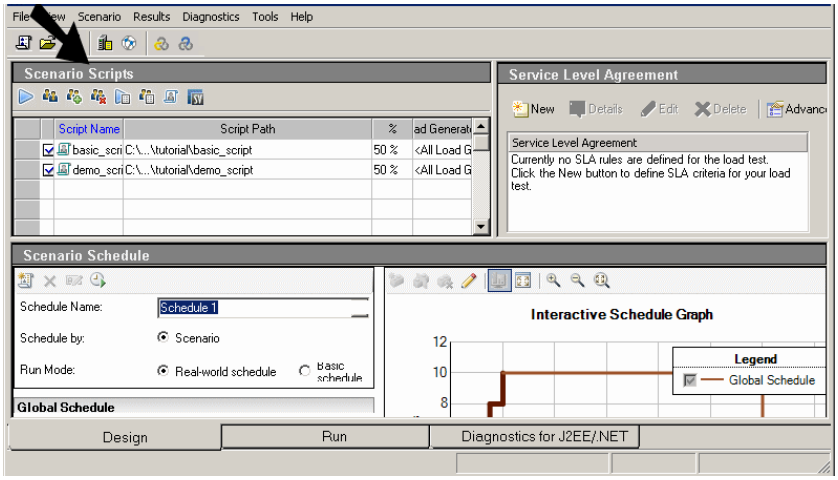
## Design Tab

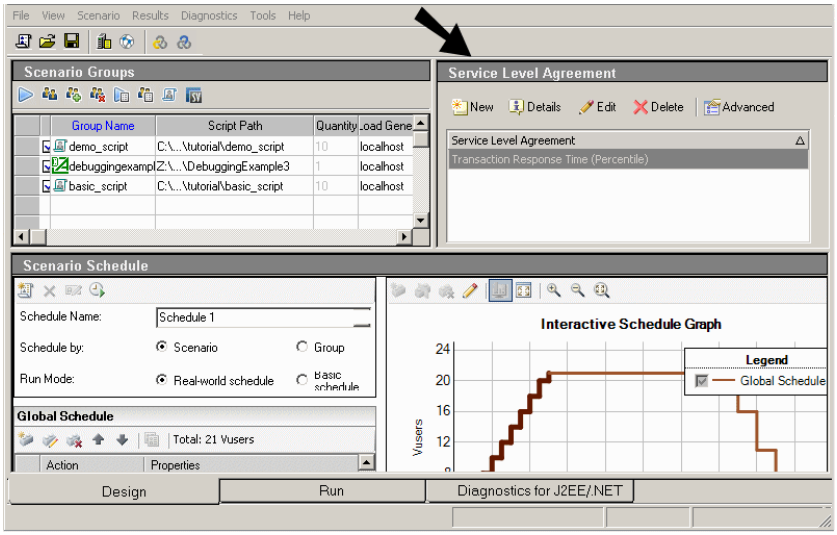
The Design tab enables you to design load test scenarios.

| | |
|---|---|
| **To access** | **Design** tab |
| **Relevant tasks** | • "How to Design a Manual Scenario" on page 953<br>• "How to Design a Goal-Oriented Scenario" on page 951<br>• "Define a Schedule for the Scenario" on page 1039<br>• "Define Service Level Agreements" on page 1301<br>• "How to Change the Scenario Mode (Manual Scenario)" on page 954<br>• "How to View/Modify Scripts in the Scenario" on page 955 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Scenario Goal pane** (Goal-oriented scenario) | In goal-oriented scenarios, displays information regarding the scenario goal.<br>  <br><br>For user interface details, see "Scenario Goal Pane" on page 971. |
| **Scenario Groups pane** (Manual scenario in Vuser Group mode) | Displays the Vuser groups created for the scenario.<br>  <br><br>For user interface details, see "Scenario Groups/Scripts Pane - Manual Scenarios" on page 972. |

| UI Element | Description |
|---|---|
| **Scenario Schedule pane** (Manual scenario) | Displays the scenario schedule.<br>  <br><br>For user interface details, see "Scenario Schedule Pane" on page 1043. |
| **Scenario Scripts pane** | Displays the Vuser scripts selected for the scenario.<br>  <br><br>For user interface details:<br>• See "Scenario Groups/Scripts Pane - Manual Scenarios" on page 972.<br>• See "Scenario Scripts Pane - Goal-Oriented Scenarios" on page 975. |

| UI Element | Description |
|---|---|
| **Service Level Agreement pane** | Lists the service level agreements defined for the scenario.<br>  <br><br>For user interface details, see "Service Level Agreements" on page 1048. |

## Edit Scenario Goal Dialog Box

This dialog box enables you to set goals for your scenario.

| | |
|---|---|
| **To access** | Goal-oriented scenario > **Design** tab > **Scenario Goal** pane > **Edit Scenario Goal** |
| **Important information** | • Available for goal-oriented scenarios only.<br>• When you run a goal-oriented scenario, the goal you defined is displayed in the appropriate graph, along with the scenario results. This enables you to compare the results with your target goal. |
| **Relevant tasks** | "How to Design a Goal-Oriented Scenario" on page 951 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Rename | Enables you to rename the selected goal profile. |
| Delete | Enables you to delete the selected goal profile. |
| New | Enables you to define a new goal profile. |

| | |
|---|---|
| Scenario Start Time... | Opens the Scenario Start dialog box where you can set the scenario start time as follows:<br>• **Without delay.** As soon as the Start Scenario command is issued.<br>• **With a delay of HH:MM:SS.** The specified time after the Start Scenario command is issued.<br>• **At HH:MM:SS on <date>.** At a specified time on a specified date. |
| **Define Scenario Goal** | The scenario goal:<br>• **Goal Type.** The type of goal. For more details, see "Goals Types for Goal-Oriented Scenarios" on page 943.<br>• **Transaction Name.** (When goal type is **Transactions per second/Transaction Response Time**) The static script transaction for your scenario to test, or the name of an automatic/dynamic script transaction that you have recorded.<br>• **Reach Goal of <value> <goal type>.** The desired goal limits.<br>• **Using a minimum of <value> and a maximum of <value> Vusers.** The minimum and maximum number of Vusers to use in the scenario. |
| **Do not change recorded think time** | If selected, LoadRunner runs the scenario using the think time recorded in the script.<br>**Note:** If you select this option, you may need to increase the number of Vusers in your scenario in order to reach your target. |
| **Goal Profile Name** | The goal profile name. |
| **Load Behavior tab** | Enables you to specify how and when the Controller should reach the target.<br>**Ramp Up.** How the Vusers should start running.<br>• **Automatic.** The Controller starts running the default number of Vusers in a batch, that is, 50 Vusers every two minutes. If the maximum number of Vusers defined is less than 50, then it runs all the Vusers.<br>• **Reach target X after.** The amount of scenario time after which Controller should reach the target.<br>• **Step up by.** How rate at which the Controller should reach the target (x number of Vusers/hits/pages every x amount of time). (Not available for the Transactions per Second and Transaction Response Time goal types.) |
| **Load Preview graph** | A graphical representation of the goal and load behavior defined for the scenario. |

| | |
|---|---|
| **Scenario Settings tab** | Enables you to specify the actions to take when the target is reached, or if the target is not reached:<br>• **Runtime**. The amount of time (in hours, minutes, and seconds) to run the scenario after the target has been reached.<br>• **If target cannot be reached.** The action to be taken if the target cannot be reached.<br>  • **Receive Notification.** If selected, the Controller sends an error message indicating that the target could not be reached. |

## Group Information Dialog Box

This dialog box displays details about the selected Vuser group, and enables you to modify the group's settings.

| | |
|---|---|
| **To access** | Manual Scenario > **Design** tab > **Scenario Groups** pane > **Details** |
| **Important information** | • Relevant for manual scenarios in Vuser group mode only.<br>• Available when a group is selected in the Scenario Groups pane. |
| **Relevant Tasks** | • "How to Design a Manual Scenario" on page 953<br>• "How to View/Modify Scripts in the Scenario" on page 955 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Updates script settings as follows:<br>• **Script.** If a script was modified during a scenario run, updates the script details in the scenario.<br>• **Runtime Settings.** If you modified a script's runtime settings from the Controller, restores the initial runtime settings. |
| | Opens VuGen where you can view and edit the script. For more information on editing scripts, see "Enhancing a Script for Load Testing - Overview" on page 336. |
| | Opens the runtime settings dialog box, where you can edit the Vuser script's runtime settings.<br>When you modify the runtime settings from the Controller, LoadRunner runs the script using the modified settings. To restore the initial settings previously set using VuGen, click the **Refresh** button and select **Runtime Settings**. |

| UI Element | Description |
|---|---|
| **Command Line** | The command line options to use when running the script.<br><br>**Example:** `-x value -y value.`<br><br>For information about passing command line argument values to a script, see "Enhance a Java Script" on page 555. |
| **Files tab** | Displays all files used by the script, including the configuration settings file, the init, run, and end portions of the script, the parameterization definitions file, and the **.usr** file.<br>• To exclude a file from the list, clear the check box adjacent to it.<br>• To add a file or folder used by the script, click the **Add** button.<br><br>**Note:** To run Visual C++ Vusers on a remote load generator, you must add the **.dll** of the Vuser to this list.<br><br>• You can delete the files that you add, but not the other files listed. |
| **Group Name** | The name of the Vuser group.<br>To modify, type a new name in the **Group Name** box. The name is limited to a maximum of 55 characters. |
| **Load Generator Name** | The load generator assigned to the Vuser group.<br>To add a load generator to this list, select **Add** from the list. For user interface details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |
| **Rendezvous tab** | Displays the rendezvous points defined for the selected script. |
| **Script** | The name, path, and type of the Vuser script selected for the Vuser group. |
| **Vusers tab** | Displays all Vusers associated with the selected script. |

## Multiple Runtime Settings Mode Dialog Box

This dialog box enables you to select the mode for modifying runtime settings of multiple selected scripts.

| | |
|---|---|
| **To access** | Use one of the following:<br>• In the **Scenario Groups/Scripts** pane, right-click a multiple selection of scripts, and select **Runtime Settings**<br>• Right-click a Vuser group that includes multiple scripts and select **Runtime Settings** |

| | |
|---|---|
| **Important information** | • If one of the selected scripts does not support shared runtime settings, then you will only have the option of modifying each script's individual runtime settings (**Individual RTS**).<br>• **Shared RTS** mode is disabled for GUI or Astra LoadTest Vusers.<br>• Some runtime settings cannot be modified in **Shared RTS** mode. These settings will not appear in the runtime settings window. To modify them, open the runtime settings for each individual script.<br>The following nodes will not appear in Shared RTS mode:<br>• **Run Logic** node - for protocols which support the **Run Logic** node, the **Iterations** box appears in the **Pacing** node<br>• **Additional Attributes** node<br>• **Internet Protocol:ContentCheck** node<br>• **Java Environment Settings:Classpath** node<br>• Nodes with tables in the format **Property, Value** for the protocols: Citrix ICA and Oracle NCA. For example: **Oracle NCA:Client Emulation** node |
| **Relevant tasks** | "How to View/Modify Scripts in the Scenario" on page 955 |

User interface elements are described below:

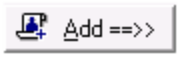
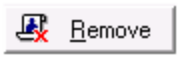
| UI Element | Description |
|---|---|
| **Individual RTS** | Opens a separate runtime settings dialog box (one at a time) for each selected script. In this mode, you modify each script's settings individually. |
| **Shared RTS** | Opens the runtime settings Shared Mode dialog box containing all of the runtime settings in blank mode. In this mode, any settings that are change are applied to all selected scripts. All other runtime settings remain unchanged.<br>See **Important information** above. |

## New Scenario Dialog Box

This dialog box enables you to create a new scenario and select Vuser scripts or test modules to run in the scenario.

| | |
|---|---|
| **To access** | Controller toolbar > **New Scenario** |
| **Important information** | Before you create a scenario, you should have a good idea as to the type of scenario you want to create. See "Introducing Controller" on page 916. |
| **Relevant tasks** | • "How to Design a Manual Scenario" on page 953<br>• "How to Design a Goal-Oriented Scenario" on page 951 |

User interface elements are described below:

| UI Element | Description |
|---|---|
|  | Moves the selected item in the **Available Scripts/Modules** box to the **Scripts/Modules in Scenario** box. |
|  | Removes the selected script or module from the **Scripts/Modules in Scenario** box. |
| | Enables you to add items to the list of available scripts or unit tests. Scripts have a **.usr** extension, while unit tests can have a **.dll**, **.jar**, or **.class** extension. |
| | Opens VuGen so that you can record a Vuser script. For details, see "Record a Vuser Script" on page 158. |
| | Opens the Connection to ALM dialog box, where you can connect to Application Lifecycle Management to download scripts.<br>For details, see "Managing Scenarios Using Application Lifecycle Management - Overview" on page 1126. |
| **Select Scenario Type** | Select the type of scenario to create. For details, see "Introducing Controller" on page 916.<br>• **Manual Scenario.** You manually build the scenario creating Vuser groups and specifying the script, load generators, and the number of Vusers to include in each Vuser group.<br>  • **Use the Percentage mode...** You define the total number of Vusers to be used in the scenario and assign a percentage of the total number of Vusers to each Vuser script.<br>  **Note:** You can convert from one scenario mode to another at any time. For details, see "How to Change the Scenario Mode (Manual Scenario)" on page 954.<br>• **Goal-Oriented Scenario.** You define the goals you want your test to achieve, and LoadRunner automatically builds a scenario for you, based on these goals. |

| | |
|---|---|
| **Test Scripts/ Test Modules** | The type of test to use in the scenario:<br>• **VuGen Scripts.** Vuser scripts created with VuGen.<br>• **System or Unit Tests.** NUnit, JUnit, or Selenium test modules created within an external application, such as Visual Studio or Eclipse. These must be files with **.dll**, **.jar**, or **.class** extensions.<br>• **JMeter Scripts.** Scripts created with Apache JMeter, with **.jmx** extension. For details, see "JMeter Tests" on page 945.<br>The **Browse** button allows you to locate the appropriate type of test. |
| **Select the script(s) you would like to use in your scenario** | (Optional) Select the scripts to use in the scenario.<br>• **Available Scripts/Modules.** Lists the fifty most recently used items.<br>• **Scripts/Modules in Scenario.** Lists the items selected for the scenario.<br>Click **Add/Remove** to move selected items between the two lists.<br>**Note:** You can change the maximum number of scripts or modules displayed in the **Available Scripts/Modules** box by modifying the following registry key:<br>• For scripts: **HKEY_CURRENT_USER\Software\Mercury Interactive\RecentScripts\max_num_of_scripts**<br>• For system and unit tests: **HKEY_CURRENT_USER\Software\Mercury Interactive\RecentModules\max_num_of_scripts** |
| **Show at startup** | When selected, the New Scenario dialog box is displayed upon opening the Controller.<br>**Note:** This option can also be enabled/disabled from the Controller's View menu. Select **View > Show New Scenario Dialog.** |

## Scenario Goal Pane

This pane displays the goals defined for the scenario.

| | |
|---|---|
| **To access** | Goal-oriented scenario > **Design** tab |
| **Important information** | • Available for goal-oriented scenarios only.<br>• The goal profiles include the type of goal, the minimum and maximum number of Vusers that should be used in the scenario, the duration of the scenario, and the load behavior. |
| **Relevant tasks** | "How to Design a Goal-Oriented Scenario" on page 951 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Opens the Edit Scenario Goal dialog box, where you set the goals for the scenario. See "Edit Scenario Goal Dialog Box" on page 965. |
| **Goal** | The defined goal, including the type of goal and the expected target. |
| **Goal Profile Name** | The name of the goal profile. |
| **Load Behavior** | How and when the Controller should reach the defined goal. |
| **Load Preview graph** | A graphical representation of the goal and load behavior defined for the scenario. |
| **Max Number of Vusers** | The maximum number of Vuser to run in the scenario. |
| **Min Number of Vusers** | The minimum number of Vuser to run in the scenario. |
| **Scenario Duration** | The amount of time the scenario should continue running after reaching the defined goal. |

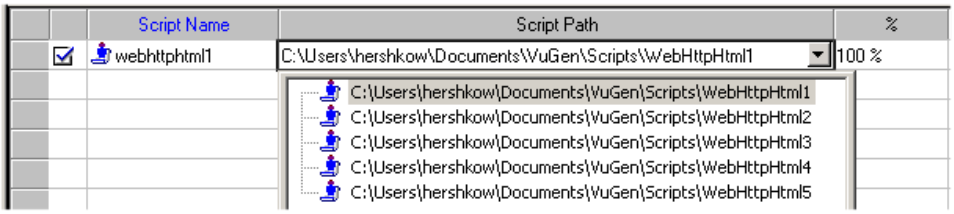
## Scenario Groups/Scripts Pane - Manual Scenarios

This pane displays the Vuser groups/scripts that were added to the scenario.

| | |
|---|---|
| **To access** | Manual Scenario > **Design** tab |
| **Important information** | The Design tab displays either the **Scenario Groups** pane or the **Scenario Scripts** pane, depending on the manual scenario mode.<br>• **Vuser group mode:** The actual number of Vusers assigned to the group. (default).<br>• **Percentage mode:** The percentage of the total number of Vusers, assigned to the group.<br>For more information, see "Manual Scenarios" on page 942. |
| **Relevant tasks** | • "How to Design a Manual Scenario" on page 953<br>• "How to Change the Scenario Mode (Manual Scenario)" on page 954 |
| **See also** | "Introducing Controller" on page 916 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | **Start Scenario.** Starts running the scenario. |
| | **Virtual Users.** Opens the Vusers dialog box where you can define properties for individual Vusers within a group. You can assign a different script or module, and load generator to each Vuser.<br>For user interface details, see "Vusers Dialog Box" on page 981. |
| | **Add Group.**<br>• **Vuser group mode:** Opens the Add Group dialog box where you create Vuser groups for the scenario. See "Add Group Dialog Box" on page 958.<br>• **Percentage mode:** Opens the Add Script dialog box where you select Vuser scripts for the scenario. See "Add Script Dialog Box" on page 960. |
| | **Remove Group.** Deletes the selected Vuser group/script. |
| | **Runtime Settings.** Opens the runtime settings dialog box.<br>• When you modify the runtime settings from the Controller, LoadRunner runs the script using the modified settings. To restore the initial settings that were assigned in VuGen, click the **Refresh** button and select **Runtime Settings**.<br>• If runtime settings were not defined in VuGen for the current script, the Controller uses default settings for Log and Think Time. Default VuGen settings are displayed for all other nodes. |
| | **Details.** Opens the Group/Script Information dialog box where you can view and modify the Vuser group/script's settings.<br>For details, see "Group Information Dialog Box" on page 967 or "Script Information Dialog Box" on page 978. |
| | **View Script.** Opens the script in VuGen where you can view and edit the Vuser script. For more information on editing scripts, see "Enhancing a Script for Load Testing - Overview" on page 336. |
| | **Service Virtualization.** Opens the Service Virtualization dialog box. For more information, see the *Service Virtualization User Guide*. |

| UI Element | Description |
|---|---|
| **<Groups/Scripts table>** | Displays the following information about the Vuser groups/scripts selected for the scenario:<br>• . Indicates that the Vuser group/script is participating in the scenario.<br>• **Group/Script Name.** The name of the Vuser group/script.<br>• **Script Path.** The path of the Vuser script. An icon adjacent to the path, indicates whether the entry is a Vuser script, a unit test module, such as .NET or Java, or a UFT/QTP test.<br>If the Vuser group includes more than one script, the scripts' names are listed. Clicking the cell's drop-down arrow displays the scripts' full paths.<br>  <br>If you want to access the script from a location that is relative to the current scenario folder, you can replace the actual path with the relative path. For details, see "Relative Paths for Scripts" on page 957.<br>• **Virtual Location.** A drop-down list of all available network virtualization locations for the group. A icon next to the virtual location name indicates that the script is run according to the per Load Generator mode. For more information on the modes available for network virtualization, see "Per Group versus Per Load Generator" on page 1246. To run this group without network virtualization, select **None**. Click **Browse** to open the Network Virtualization Settings dialog box.<br>**Note:** This list is only available when the Network Virtualization software is installed. This column does not appear if you disable network virtualization. To enable network virtualization or edit the locations, see "Virtual Locations Settings Dialog Box" on page 1243.<br>• **Quantity.** (Vuser group mode) The number of Vusers assigned to the Vuser group.<br>This column is read-only when the defining a real-world schedule (default schedule). In this case, the quantity of Vusers is defined when designing the scenario schedule.<br>• **%.** (Percentage mode) The percentage of Vusers assigned to run the Vuser script. |

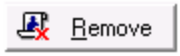
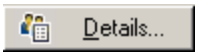
| UI Element | Description |
|---|---|
| | If you modify the percentage assigned to one group, the percentages assigned to the other scripts change to create a total of 100% for all of the Vuser scripts.<br><br>**Note:** Modify percentages to the scripts starting with the first script in the list and moving down the list.<br><br>• **Load Generators.** The load generators assigned to the Vuser group/script.<br>If you select multiple load generators for a group/script, the Vusers assigned to the Vuser group/script are distributed evenly among the load generators.<br>**Default value (in percentage mode)**: All Load Generators<br><br>**Note:** To add a load generator to this list, select **Add** from the list. For more details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |
| **<Groups/Scripts table>** (continued) | • **Load Generators.** The load generators assigned to the Vuser group/script.<br>If you select multiple load generators for a group/script, the Vusers assigned to the Vuser group/script are distributed evenly among the load generators.<br>**Default value (in percentage mode)**: All Load Generators<br><br>**Note:** To add a load generator to this list, select **Add** from the list. For more details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |
| **<Right-click menu>** | • **Auto Sort.** When adding a Vuser group/script, automatically sorts the table according to the defined sort.<br>• **Sort Groups/Scripts.** Enables you to sort the table by Vuser group/script name, script path, quantity/percentage of Vusers, or load generator. To sort the table in ascending/descending order, click the relevant table heading. |

## Scenario Scripts Pane - Goal-Oriented Scenarios

This pane lists the Vuser scripts selected for the goal-oriented scenario.

| | |
|---|---|
| **To access** | Goal-oriented scenario > **Design** tab |
| **Relevant tasks** | "How to Design a Goal-Oriented Scenario" on page 951 |
| **See also** | "Introducing Controller" on page 916 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | Starts running the scenario. |
| | Opens the Load Generators dialog box where you can add new load generators and view details about existing load generators. See "Load Generators Dialog Box" on page 1019. |
| | Opens the Add Script dialog box where you can select Vuser scripts to add to the scenario. See "Add Script Dialog Box" on page 960. |
|  | Deletes the selected Vuser script. |
| | Opens the runtime settings dialog box, where you can edit the Vuser script's runtime settings.<br><br>**Note:**<br>• When you modify the runtime settings from the Controller, LoadRunner runs the script using the modified settings.<br>• If runtime settings for a script were not defined in VuGen, the Controller displays its own default settings for Log and Think Time. Default VuGen settings are displayed for all other nodes. |
|  | Opens the Script Information dialog box where you can view the Vuser script's settings. See "Script Information Dialog Box" on page 978. |
| | Opens the script in VuGen where you can view and edit the Vuser script. For more information on editing scripts, see "Scripting Options Tab" on page 119. |
| **<Right-click menu>** | • **Auto Sort.** When adding a script, automatically sorts the table according to the defined sort.<br>• **Sort Scripts.** Enables you to sort the table by Vuser script name, script path, percentage of Vusers, or load generator. To sort the table in ascending/descending order, click the relevant table heading. |

| UI Element | Description |
|---|---|
| **<Scripts table>** | Displays the following information about the Vuser scripts selected for the scenario:<br>• . Indicates that the script is participating in the scenario.<br>• **Script Name.** The name of the Vuser script.<br>• **Script Path.** The path of the Vuser script. If you want to access the script from a location that is relative to the current scenario folder, you can replace the actual path with the relative path. For details, see "Relative Paths for Scripts" on page 957.<br>• **Virtual Location.** A drop-down list of all available network virtualization locations for the group. A icon next to the virtual location name indicates that the script is run according to the per Load Generator mode. For more details, see "Per Group versus Per Load Generator" on page 1246. To run this script without network virtualization, select **None**. Click **Browse** to open the Network Virtualization Settings dialog box.<br>**Note:** This list is only available when the Network Virtualization software is installed. This column does not appear if you disable network virtualization. To enable network virtualization or edit the locations, see "Virtual Locations Settings Dialog Box" on page 1243.<br>• **% of Target.** The percentage of the overall target number of Vusers, pages per minute, hits per second, transactions per second, or transaction response time that is automatically distributed to each Vuser script.<br>• **Load Generators.** The load generators assigned to the script. If you select multiple load generators for a script, the Vusers assigned to the script are distributed evenly among the load generators.<br>**Default value**: All Load Generators<br>**Note:** To add a load generator to this list, select **Add** from the list. For more details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |

## Scenario Start Time Dialog Box

This dialog box enables you to set when the scenario will start.

| | |
|---|---|
| **To access** | Goal-oriented scenario > **Design** tab > **Scenario Schedule** pane > **Scenario Start Time** button. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Without delay | Starts the scenario as soon as the Start Scenario command is issued. |
| With a delay of HH:MM:SS | Starts the scenario the specified time after the Start Scenario command is issued. |
| At HH:MM:SS on <date> | Starts the scenario at a specified time on a specified date. |

## Script Information Dialog Box

This dialog box displays details about the selected Vuser group, and enables you to modify the group's settings.

| | |
|---|---|
| **To access** | **Design** tab > **Scenario Scripts** pane > **Details** |
| **Important information** | Available when a script is selected in the Scenario Scripts pane.<br>Relevant to:<br>• Manual scenarios in percentage mode<br>• Goal-oriented scenarios |
| **Relevant tasks** | • "How to View/Modify Scripts in the Scenario" on page 955 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Updates script settings as follows:<br>• **Script.** If the script was modified during a scenario run, updates the script details in the scenario.<br>• **Runtime Settings.** If you modified the runtime settings from the Controller, restores the initial runtime settings. |
| | Opens VuGen where you can view and edit the script. For more information on editing scripts, see "Enhancing a Script for Load Testing - Overview" on page 336.<br>**Note:** If you use VuGen to make changes to a script while the Controller is running, click **Refresh** and select **Script** to update the script details in the scenario. |

| UI Element | Description |
|---|---|
| | Opens the runtime settings dialog box, where you can edit the Vuser script's runtime settings.<br><br>**Note:**<br>• When you modify the runtime settings from the Controller, LoadRunner runs the script using the modified settings. To restore the initial settings previously set using VuGen, click the **Refresh** button and select **Runtime Settings**.<br>• If runtime settings for a script were not defined in VuGen, the Controller displays its own default settings for Log and Think Time. Default VuGen settings are displayed for all other nodes. |
| | Opens the "Virtual Locations Settings Dialog Box" on page 1243. This dialog box lets you enable Network Virtualization, or change the working mode (**Per Group** or **Per Load Generator**). It also provides a link to the Network Virtualization software screens, allowing you to configure the virtualization properties. |
| **Command Line** | The command line options to use when running the script.<br><br>**Example:** `-x value -y value.`<br><br>For information about passing command line argument values to a script, see "Enhance a Java Script" on page 555. |
| **Files tab** | Displays all files used by the script, including the configuration settings file, the init, run, and end portions of the script, the parameterization definitions file, and the **.usr** file.<br>• To exclude a file from the list, clear the check box adjacent to it.<br>• To add a file or folder used by the script, click the **Add** button.<br><br>**Note:** To run Visual C++ Vusers on a remote load generator, you must add the **.dll** of the Vuser to this list.<br><br>• You can delete the files that you add, but not the other files listed. |
| **Rendezvous tab** | Displays the rendezvous points defined for the selected script. |
| **Script** | The name, path, and type of the selected Vuser script. |
| **Vusers tab** | Displays all Vusers associated with the selected script. |

# Vuser Information Dialog Box

This dialog box displays details about a specific Vuser in a group, and lets you modify the load generator and script settings for the Vuser.

| | |
|---|---|
| **To access** | Use one of the following:<br>• Manual scenario > **Design** tab > **Scenario Groups** pane > **Vusers** > <br>• In Vusers dialog box, double-click Vuser. |
| **Relevant tasks** | • "How to View/Modify Scripts in the Scenario" on page 955<br>• "How to Run a Scenario" on page 1094 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Enables you to add Vuser scripts to the list of scripts. |
| | Opens VuGen where you can record a Vuser script. For more information on recording Vuser scripts, see "Record a Vuser Script" on page 158. |
| | Opens the runtime settings dialog box, where you can edit the Vuser script's runtime settings.<br>**Note:**<br>• Modifying the runtime settings for one Vuser modifies the runtime settings for all the Vusers in the group that are using the same script.<br>• When you modify the runtime settings from the Controller, LoadRunner runs the script using the modified settings.<br>• If runtime settings for a script were not defined in VuGen, the Controller displays its own default settings for Log and Think Time. Default VuGen settings are displayed for all other nodes. |
| | Opens the Parameter list in VuGen where you can create, view, modify, and delete Vuser script parameters. For details, see "Parameterization Overview" on page 357. |
| **Group Name** | The name of the group to which the selected Vuser belongs. |

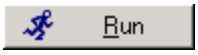
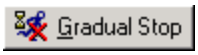
| | |
|---|---|
| **Load Generator Name** | The load generator assigned to the Vuser's Vuser group.<br>To add a load generator to this list, select **Add** from the list. For user interface details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |
| **Select Script** | Lists the available scripts that have been added to the scenario. When you select a script, its name and path are displayed above the list.<br>To display the scripts with their full paths, right-click the list area and select **Show Paths**. |
| **Vuser Name** | The name of the selected Vuser. |

## Vusers Dialog Box

This dialog box displays the status of the Vusers in the selected Vuser group.

| | |
|---|---|
| **To access** | Manual scenario > **Design** tab > **Scenario Groups** pane > **Vusers** |
| **Relevant tasks** | "How to Run a Scenario" on page 1094 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | **Show Vuser Log.** Displays a log containing runtime information about the Vuser that is refreshed, by default, every 1000 milliseconds. For more information, see "Vuser Script Log" on page 1112. |
| | **Hide Vuser Log.** Closes the Vuser log. |
|  | Starts running the selected Vuser. |
|  | Instructs the Controller to complete the current iteration or action before stopping the Vuser.<br><br>**Note:** This option is only available when the Vuser is in the **Run** state, and if you selected the **Wait for the current iteration to end before exiting** or **Wait for the current action to end before exiting** option in the runtime settings tab of the Options dialog box. |
| | Stops running the selected Vuser immediately. |

| | |
|---|---|
| Reset | Resets the status of the Vuser to Down. |
| Details... | Opens the Vuser Information dialog box, where you can view details about the selected Vuser. |
| Add Vuser(s)... | Opens the Add Vusers dialog box where you can add more Vusers to the Vuser group. |
| Refresh | Refreshes the contents of the columns in the Vuser list. This is relevant when you make changes outside of this window. The changes may be in the group or test information or, when working with network virtualization, updating the virtual location. |
| | Indicates that this Vuser is collecting Web page diagnostic data for analysis.<br>For configuration details, see "Diagnostics Distribution Dialog Box" on page 1261.<br>For details on the analysis graphs, see "Web Page Diagnostics Graphs Overview" on page 1494. |
| **<Filter by script>** | Filters the Vusers table by the selected script. |
| **<Filter by status>** | Filters the Vusers table by the selected Vuser status. For details, see "Vuser Statuses" on page 958. |
| **<Right-click menu>** | • **Filter Vusers.** Enables you to filter the Vuser list by Vuser status.<br>• **Renumber.** Renumbers the list of Vusers in sequential order, starting from 1.<br>• **Sort Vusers.** Enables you to sort the table by a selected column. To sort the table in ascending/descending order, click the relevant table heading. |
| **<Vusers table>** | Displays the following information about the Vusers:<br>• **ID.** The Vuser's ID number<br>• **Status.** The Vuser's status. For details, see "Vuser Statuses" on page 958.<br>• **Script.** The script run by the Vuser.<br>• **Virtual Location.** The virtual location being emulated (read-only). This column is only visible when Network Virtualization is enabled and set to **Per Load Generator** mode. For details, see "Virtual Locations Settings Dialog Box" on page 1243.<br>• **Load Generator.** The load generator on which the Vuser is running.<br>• **Elapsed Time.** Amount of time that elapsed in the scenario since the Vuser began running. |

# Load Generators

Load generators are the machines that run the Vuser scripts in your scenarios. Each Vuser script that runs on a load generator results in a single Vuser. Each load generator can run multiple Vuser scripts, thereby resulting in multiple Vusers. While these Vusers run, they create load on your system, enabling you to analyze your system under load.

## Creating load generators

To create a load generator, you install LoadRunner's Load Generator software on a host computer. Load generators can be either Windows-based or Linux-based. For Docker platforms, you can use one of the available Dockerized load generator images (see "Configure Dockerized Load Generators" on page 985).

For details on how to install the appropriate Load Generator software, see the *LoadRunner Installation Guide*.

**Including load generators in a scenario**

After you have installed the Load Generator software on a host computer to create a load generator, you can include the new load generator in a scenario. You use Controller to add the new load generator to a scenario.

To enable a scenario to develop significant load on a system, a typical scenario may include multiple load generators. You define and maintain the list of load generators that are available in a scenario. When you add a load generator to the list of load generators in a scenario, you define various attributes of the load generator, such as the platform on which the load generator runs. You can modify some - but not all - of these attributes after the load generator is created.

**Enabling and disabling a load generator**

If a specific load generator is included in the list of load generators for a scenario, but is temporarily unavailable for a particular scenario run, you can disable the load generator instead of removing it permanently from the list. You can then enable the load generator if and when it becomes available.

> **Note:** The LoadRunner standalone Load Generator (LG SA) and standalone Monitor over Firewall (MOFW SA) cannot be installed on the same machine. However, LG SA can be used for monitoring purposes, the same way as the MOFW SA. Note that a single machine cannot be used simultaneously for both running Vusers and monitoring.

## Local versus cloud-based load generators

A load generator can be located on either a local computer or in the cloud. A scenario can include both local and cloud-based load generators. The procedure for defining and maintaining load generators is similar for local and cloud-based load generators. For details, see "Add Load Generators to a Scenario" on page 1002.

## Load generator configuration

To create and run accurate real-life scenarios, you can configure numerous settings for each load generator that is included in a scenario. For details, see "How to Modify Load Generator Settings" on page 1004.

## Status of a load generator

Controller's Load Generators dialog box shows the status of each load generator in a scenario. The status of a load generator changes during a scenario run. For example, when you add a load generator to the list of load generators in a scenario, the load generator has the status **Down**. The status of the load generator changes to **Ready** when the load generator is connected to Controller and is available to run Vuser scripts, and then to **Running** while the load generator executes a Vuser script. For a full list of load generator statuses, see "Load Generators Dialog Box" on page 1019.

### See also:

- "Setting up a Load Generator Environment" below
- "Add Load Generators to a Scenario" on page 1002

## Setting up a Load Generator Environment

Follow the guidelines below to set up the environments of your load generators.

- In general, you should always make sure to configure the load generator machine the same as the machine upon which you recorded or prepared the script or test.
- For Java scripts and JUnit tests, verify that the script or test dependencies are available on the load generator machine with the same paths defined in the classpath Runtime Settings. You can do this by mapping network drives, manually copying files, and so forth.
- For NUnit tests, make sure that the NUnit framework is installed on the load generator machine in the same path as the Controller machine.
- For Linux machines, make sure to configure the environment variables as described in "Linux Environment Variables" on page 1006.
- **Load generator optimization.** For load generators that are installed on **dedicated machines**, you can configure the system setup for load generator optimization when necessary.

> **Caution:** It is recommended to always use default load generator settings. The impact of making changes to these settings for load generator optimization are described below.

| Optimization | Description and impact |
|---|---|
| **Increase number of available ports** | Determines the highest port number that TCP can assign when an application requests an available user port from the system. Typically, **ephemeral** ports (those used briefly) are allocated to port numbers 1024 through 5000.<br>**Impact:** Increasing the value enables the operating system to use a larger port range, resulting in a larger number of concurrent TCP connections.<br>Run the following from the command prompt:<br>• **Windows:** netsh int ipv4 set dynamicport tcp start=1025 num=64510<br>• **Linux:** sysctl -w net.ipv4.ip_local_port_range="1024 65535" |
| **Increase number of maximum open files (Linux only)** | Additional configuration to support increase in number of available ports on Linux:<br>a. Edit file **/etc/security/limits.conf**.<br>b. Add the following lines:<br>root soft nofile 65535<br>root hard nofile 131070 |
| **Shorten TCP connection "wait time"** | Determines the time that must elapse before TCP can release a closed connection and reuse its resources.<br>Reducing the value of this entry allows TCP to release closed connections faster, providing more resources for new connections.<br>**Impact:** If the value is too low, TCP might release connection resources before the connection is complete, requiring the server to use additional resources to reestablish the connection.<br>• **Windows:** Edit the Windows registry:<br>◦ **Key:** HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\TCPIP\Parameters<br>◦ **Value:** Set **TcpTimedWaitDelay** to **5**<br>• **Linux:** Run the following from the command line:<br>sysctl -w net.ipv4.tcp_fin_timeout=5 |

## Configure Dockerized Load Generators

This section describes how to set up load generators on a Linux or Windows distribution using Docker.

**Note:**
- The Windows Dockerized load generator feature is provided as a tech preview version, for Web - HTTP/HTML and Java protocols.

- Dockerized load generators are not supported when running over a firewall.

## About Dockerized load generators

Docker is a platform that allows you to develop, ship, and run applications via a container. Docker uses a standard container format that lets developers prepare applications inside containers, while system administrators and other teams, such as Quality Assurance engineers, run the container in order to deploy the application. This is a quick and effective way to port applications across systems and machines, and run them within their own secure environments.

You can run load generator hosts inside Docker containers, using Linux or Windows Dockerized load generator images.

For general details regarding Docker, see https://docs.docker.com.

## Dockerized load generator image

You can pull load generator Docker images for Linux or Windows from the following pages in the Docker hub:

- Linux Dockerized load generator
- Windows Dockerized load generator (tech preview version)

**Note:** The image for Linux is also provided with the LoadRunner installation package, in the **InsData** folder of the ISO file.

For instructions on running the Dockerized load generator, see the LoadRunner Installation Guide.

## Assign load generator containers to scenarios

To assign the Dockerized load generator containers to scenarios, in the **Add New Load Generator** dialog box, configure the load generator **Name** as follows:

```
<host machine name>_dkr_<port_number>
```

Replace **<port_number>** with the port that is mapped to the Docker container (the <host_port> used when running the container).

For example:

```
myHost_dkr_10110
```

**Note:** Any errors encountered by Dockerized load generators are written to the Output window under the Docker node host name and port.

# Load Generators on the Cloud

Load generators can be hosted on local computers or in a cloud account. To host a load generator on the cloud, you must have access to a cloud account that will host the load generator, and your cloud account must be set up for provisioning load generators.

> **Note:** All costs incurred by the provisioning of cloud machines through LoadRunner, are external to LoadRunner. They are the sole responsibility of the user, and are subject to the cloud vendor's pricing schedule.

## Define your cloud account in LoadRunner

Controller is used to maintain the list of cloud accounts that are available to host load generators.

LoadRunner supports cloud accounts on specified cloud providers, such as Amazon EC2 or Microsoft Azure. For a full list of the supported cloud providers, see the System Requirements.

### To add your cloud account to LoadRunner:

1. In Controller, select **Tools > Manage Cloud Accounts**. The Manage Cloud Accounts dialog box opens, and lists all the accounts that are available to host load generators. For details, see "Manage Cloud Accounts Dialog Box" on page 1022.
2. Click the button.
3. Enter a name for the cloud account. If required, you can modify the name later.
4. Select the provider of the cloud account.
5. Specify the provider-specific information for your account, For example, for Amazon EC2, provide the **Access Key ID** and **Secret Access Key** that are required to access the account.

   > **Note:** When adding multiple accounts, each must have a unique provider and access key.

6. You can specify how long to wait after the end of a failed scenario before removing the load generators from the account. This is useful if the collation did not complete successfully and you need time to determine why it failed.
7. Select the network profile. The network profile defines the connection between the Controller and the cloud providers. Set the connection parameters in the proxy settings section of the Network Profile Manager dialog box. For details on how to create a network profile, see "Network Profile Manager Dialog Box" on page 1028.
8. Click **Save**.

## Set up your cloud account for provisioning load generators

The process of creating a new load generator in your cloud account is known as **provisioning** the load generator.

Load generators are provisioned when setting up a load test scenario. The load generator software is installed by Controller on the cloud, using an "image" of the required load generator contained in the default system configuration.

Alternatively you can specify an image that is included in a custom configuration. For details, see "Create Custom Image for Load Generators on the Cloud" on page 992.

> **Note:** You can choose to provision either a single load generator or to simultaneously provision multiple load generators with the same configuration.

Before you provision load generators, perform the following configuration in your cloud accounts.

### To set up your cloud account for provisioning load generators:

1. **Create a security group.**

   A security group is a configuration for allowing or disallowing specific traffic to and from a machine. To use load generators on the cloud, you must use a security group allowing load generator communication.

   Create a security group in your account management console, with a custom TCP rule that allows inbound traffic on the port used for the communication between the load generator and the Controller. By default, LoadRunner uses port 54345. This can easily be changed using a network profile, described below.

   The source IP should be **Anywhere**, unless you will be using a Controller from a specific, single IP address.

2. **Create a key pair. (Optional)**

   A key pair is a security measure used for interactive logging in to cloud machines. This is not strictly required for using load generators on the cloud, but may be useful for cases in which you need to log into a machine manually, third-party software installation, or local log inspection.

   If required, create a key pair in your account management console and save the generated **.pem** file.

3. **Create an access key.**

   Controller uses Access keys to communicate with the cloud provider.

   For EC2, an access key is composed of an **access Key ID** and a **secret access key**. This access key is different from your EC2 Web Console access credentials and must be created separately. Refer to the AWS documentation for information on creating an Access Key. Save the access key ID and secret access key.

   For Microsoft Azure, you need to provide a Subscription ID and path to the relevant certificate file. For details, refer to the Microsoft Azure documentation.

## Set up secure communication:

SSL encryption in used to ensure secure communication between the local Controller and load generator on the cloud machine. You must own a CA certificate file and either a CA private key file or an SSL certificate file.

## To set up secure communication:

1. **Generate certificates.**

   To create new certificates, run the **gen_ca_cert.exe** utility . This process creates two files in the folder from which the utility was run: the CA Certificate file, used to sign SSL certificates (cacert.cer), and the CA Private Key, used with the CA certificate to generate an SSL certificate (capvk.cer). For details, see "Configure Secure Communication with TLS (SSL)" on page 1153.

2. **Install the certificates.**

   In Controller, select **Tools > Authentication Settings**, and indicate the CA and Private Key or SSL certificates that you created earlier. For details, see "Authentication Settings Tool" on page 1163.

3. **Configure the proxy server. (Optional)**

   If you are using Controller behind a proxy server (for example, an enterprise proxy server), the proxy server must be configured to allow communication.

   The following ports must be enabled in the CONNECT method:

   - 443 – Used for communication with Amazon AWS API.
   - The port (by default 54345) used for communication between the load generator and Controller. You can configure this with a network profile as described below.

   If your environment includes an HTTP 1.0 proxy server using NTLM, make sure to enable the port defined in the network profile for communication with the load generator machine, in order to bypass the NTLM authentication.

4. **If you are using a proxy server, configure the proxy locally.**

   Through your Windows Control Panel, go to **Internet Options**, and in the Connection tab, click **LAN Settings** to specify your proxy server information.

5. **Create a Controller network profile.**

   If you have not already set up a Controller network profile:

   a. In Controller select **Tools > Network Profile Manager**.

   b. In the Network Profile Manager dialog box, click the **Add** button. Click on an empty line in the **Network Profile Name** list and create a new profile. For details, see "Network Profile Manager Dialog Box" on page 1028.

   c. If required, select **Use proxy** and fill the proxy server details.

   d. In the **Connection Mode** section, optionally specify a different port, to be used for Controller's communication with the load generators using this network profile.

6. **Add a virtual account.**

a. In Controller, select **Tools > Manage Cloud Accounts**.
b. Enter an account name and select an account provider.
c. Fill in the access credentials, such as the access and secret keys, that you created above. You can retrieve these keys from the cloud provider's management console.
d. Optionally, specify a time delay to terminate the load generator if collation fails. For details, see "Manage Cloud Accounts Dialog Box" on page 1022.
e. Click **Save**. LoadRunner validates the credentials against the cloud provider.

### See also:

- "Manage Load Generators on the Cloud" below
- "Manage Cloud Accounts Dialog Box" on page 1022
- "Create Cloud Load Generator Dialog Box" on page 1024
- "Use Cloud Load Generator Dialog Box" on page 1027

## Manage Load Generators on the Cloud

This section describes how to manage load generators on the cloud for LoadRunner load test scenarios.

### Provision load generators on the cloud

The process of creating a new load generator in a cloud account is known as *provisioning* the load generator. You can provision either a single load generator or multiple load generators with the same configuration. You can use either the default system configuration, or define and apply a custom configuration.

You provision load generators when setting up load test scenario. For details, see "Add Load Generators to a Scenario" on page 1002.

### Rename provisioned load generators

You can rename load generators that you provisioned. The renaming affects the provider side too, so that the new name appear in the list of available cloud load generators.

> **Note:** The load generator name must not exceed 128 characters.

### Delete provisioned load generators from scenario

You can delete cloud load generators directly from a scenario:

1. In the Load Generators dialog box, select the machine you want to delete and click **Delete**.
2. In the Delete Load Generator window, select one of the following options:
   - **Delete from Scenario.** The machine will be removed from this Controller, but it is still up and running.

- **Delete and Terminate.** When not being used, cloud machines may still be incurring costs to your account. If you select this option, the load generator is removed from the scenario and also terminated on the cloud.

### Terminate provisioned load generators automatically after scenario run

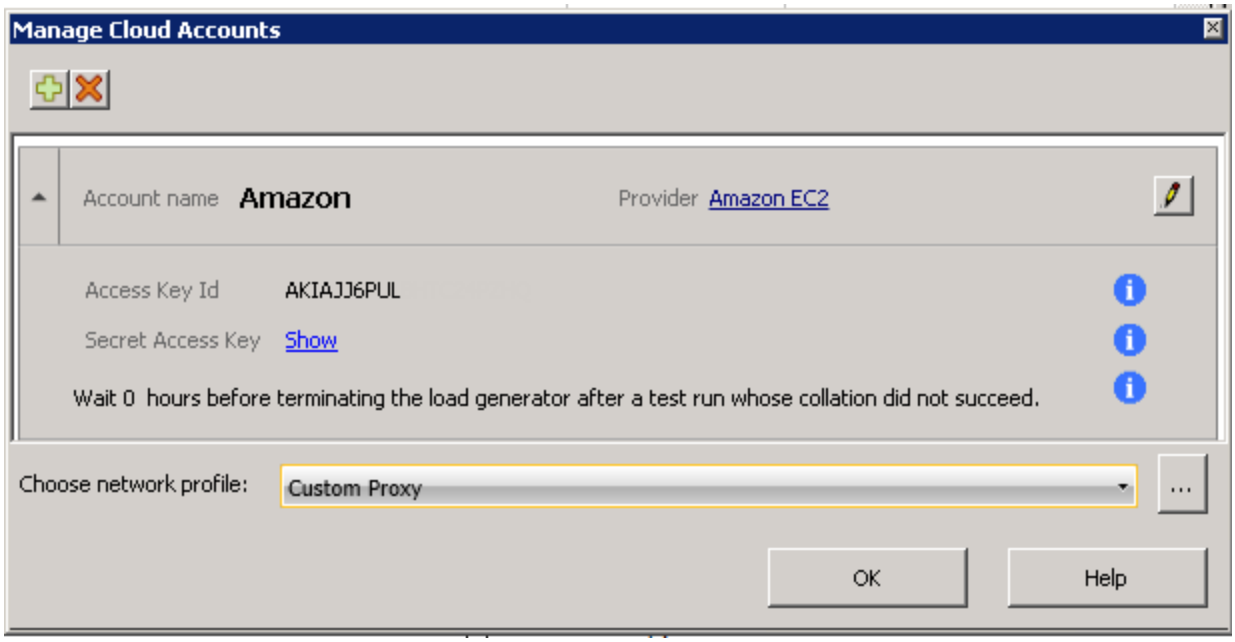
You can configure Controller to automatically terminate a load generator on the cloud after a scenario is completed. Load generators are removed in order to reduce cloud usage and the associated costs. For details on how to specify whether or not to remove load generators at the end of a scenario, see "Add Load Generators to a Scenario" on page 1002.

If you specify to terminate load generators at the end of a scenario run, its termination differs slightly depending on whether the scenario is completed successfully or fails.

When a scenario completes successfully, the load generator is no longer required and is terminated immediately.

If a scenario fails, however, you may want to give the scenario some time to rerun. You can specify to terminate the load generator after a scenario fails.

This delay may be useful because it enables you to access information on the load generator that may indicate why the scenario failed. After the load generator is removed, this information is no longer available.




## Create Certificates for Azure Cloud

This section describes how to create a self-signed certificate. To create a certificate signed by CA, refer to the OpenSSL documentation.

To create a self-signed certificate:

1. Install the prerequisite software.
   a. Download and install Microsoft Visual C++ 2008 Redistributable Package (x86) from http://www.microsoft.com/en-us/download/details.aspx?id=29.
   b. Download and install the latest version of OpenSSL from http://slproweb.com/products/Win32OpenSSL.html. Use all of the default settings.
2. Set up the OpenSSL environment.

   Run the following commands:
   a. set OPENSSL_CONF=C:\OpenSSL-Win32\bin\openssl.cfg
   b. cd C:\OpenSSL-Win32\bin
   c. openssl req -newkey rsa:2048 -new -nodes -x509 -days 3650 -keyout myazurekey.pem -out myazurekey.pem

   Enter the required information. When prompted, press **Enter** to confirm each operation. A file titled **myazurekey.pem** is created in the C:\OpenSSL-Win32\bin folder.
3. Create the certificate.

   Run the following command:

   openssl x509 -inform PEM -in myazurekey.pem -outform DER -out myazurekey.cer.

   The **myazurekey.cer** file is created in the C:\OpenSSL-Win32\bin folder.
4. Upload the certificate to the Azure machine.

   Use the Azure Console > Settings > Manage Certificates view to upload the myazurekey.cer file to Azure.
5. Add the certificate in the Controller.

   In the LoadRunner Controller, select **Tools > Manage Cloud Accounts.** Select **Microsoft Azure** as the provider, and specify the **myazurekey.pem** file as the certificate. For details, see "Manage Cloud Accounts Dialog Box" on page 1022.

## Create Custom Image for Load Generators on the Cloud

The LoadRunner installation includes images for load generators for various cloud environments. This topic describes how to create custom images for those environments.

### Azure – Windows

To create a custom image for Azure – Windows:

1. Create new instance from the cloud console, using a Windows Server 2012 R2 Standard x64 image. For details, see http://azure.microsoft.com/en-us/documentation/articles/virtual-machines-create-custom/.
2. Connect to the instance that you just created. For details, see http://azure.microsoft.com/en-us/documentation/articles/virtual-machines-log-on-windows-server/.
3. Install an LG SA (Standalone Load Generator) in Performance Center (PC) Mode.
4. When the Load Generator is installed:

a. Open the Server Manager on the machine.
b. From the **Tools** menu, select **Computer Management**.
c. Under **System Tools > Local Users and Groups**, select **Groups**.
d. On the right, open the properties of the **Administrators** group.
e. In the **General** tab, under **Members**, remove **IUSR_METRO**.
f. Click **OK**.

5. Turn off firewalls using the following command:

```
netsh advfirewall set allprofiles state off
```

6. Register the LGCloudAgent service using the following command:

```
sc create LGCloudAgent binPath= "c:\Program Files
(x86)\Micro Focus\Load Generator\bin\LGCloudAgent.exe" start= auto
```

7. Disable autostart for the LoadRunnerAgent service:

```
sc config LoadRunnerAgent start= demand
sc config remotemanagementagent start= demand
```

8. Run the System Preparation Tool using the **Generalize** and **Shutdown** options, as described in http://azure.microsoft.com/en-us/documentation/articles/virtual-machines-capture-image-windows-server/. Wait until the instance stops.
9. You must create an SSL certificate in order to upload to Azure. For details, see "Create Certificates for Azure Cloud" on page 991. Upload the *.cer file to Azure using **Settings > Management Certificates**.
10. Create an image from the instance.

```
%LR_root%\bin\AzureCaptureRole.exe -s $subscription_id -c $path_to_
certificate_file -n $vmname -i $new_image_name -v $lr_version –d $lr_
drop --proxy proxy_url
```

Follow these naming guidelines:

| | |
|---|---|
| **$path_to_ certificate_file** | The path to the .pem file corresponding to .cer file, that you previously uploaded to Azure |
| **$vmname** | The name of the instance from the Web console |
| **$lr_version, $lr_ drop** | The version and drop of LoadRunner, for example "12.60" |

| | |
|---|---|
| **$proxy_url** | The url and port of a proxy server, if relevant |
| **$subscription_id** | The subscription id of the image<br>To determine this value, see the **Settings > Subscriptions** page on the Azure console. |

### Azure - Ubuntu

To create a custom image for Azure – Ubuntu:

1. Create new instance from the cloud console, using an Ubuntu 12.04 x64 image.
2. Connect to the instance that you just created.
3. Run the following commands to update the image:

```
sudo apt-get update
sudo apt-get upgrade
```

4. Upload the Load Generator standalone installation for Linux to the machine.
5. Execute following command for uploaded files:

```
chmod a+x inst64.bin installer.csh installer.sh unzip
```

6. Install the Load Generator:

```
sudo apt-get install libc6-i386 lib32stdc++6 lib32ncurses5
libkrb5support0:i386 libkeyutils1:i386 gsoap:i386 mc libglib2.0-0:i386
```

7. Run the following command:

```
bash installer.sh
```

8. Install the Standalone Load Generator, accepting all of the defaults. Skip the Certificate Installation step and the Load Generator startup.
9. Insert the following lines into **/etc/rc.local**:

```
- egrep '/([[:alnum:]]+:)?CustomData' /var/lib/waagent/ovf-env.xml | sed -re 's,.*<
([[:alnum:]]+:)?CustomData>,,;s,</([[:alnum:]]+:)?CustomData>.*,,' | base64 -d >
/var/lib/waagent/CustomData
- export LD_LIBRARY_PATH=/opt/MF/MF_LoadGenerator/bin
- export PRODUCT_DIR=/opt/MF/MF_LoadGenerator
- export M_LROOT=/opt/MF/MF_LoadGenerator
```

- /opt/MF/MF_LoadGenerator/bin/LgCloudAgent

10. Run the following:

```
sudo waagent -deprovision
```

11. Shut down the instance and verify that it has stopped.
12. You must create an SSL certificate in order to upload to Azure. For details, see "Create Certificates for Azure Cloud" on page 991.
13. Create an image from the instance.

```
%LR_PATH%\bin\AzureCaptureRole.exe -s $subscription_id -c $path_to_
certificate_file -n $vmname -i $new_image_name -v $lr_version --proxy
proxy_url
```

Follow these naming guidelines:

| | |
|---|---|
| **$path_to_ certificate_file** | The path to the .pem file corresponding to .cer file, that you previously uploaded to Azure |
| **$vmname** | The name of the instance from the Web console |
| **$lr_version** | The version of LoadRunner, for example "12.60" |
| **$proxy_url** | The url and port of a proxy server, if relevant |
| **$subscription_id** | The subscription id of the image<br>To determine this value, see the **Settings > Subscriptions** page on the Azure console. |

### Amazon EC2 – Windows

To create a custom image for Amazon EC2 – Windows:

1. Launch a new instance from the cloud console. Use the latest Windows Server 2012 R2 Standard x64 image. Make sure the Shutdown behavior is set to Terminate. For details, see the AWS Documentation.
2. Connect to your Amazon EC2 instance. For details, see the AWS Documentation.
3. Install an LG SA (Standalone Load Generator) on the cloud machine in the default PC Mode. The setup file is in the LoadRunner installation package's Standalone Applications folder.
4. When the Load Generator is installed:
   a. Open the Server Manager on the machine.
   b. From the **Tools** menu, select **Computer Management**.
   c. Under **System Tools > Local Users and Groups**, select **Groups**.

d. On the right, open the properties of the **Administrators** group.
e. In the **General** tab, under **Members**, remove **IUSR_METRO**.
f. Click **OK**.

5. Turn off the firewall using the following command:

```
netsh advfirewall set allprofiles state off
```

6. Install any additional software required, for example, a Citrix client.
7. If you are planning to use the Citrix protocol, check that your network can access the cloud machine. We recommend that you disable firewalls completely on the cloud machine, and use the network management, such as security groups, provided by the cloud provider.
8. Register the **LGCloudAgent** service on the machine using the following command:

```
sc create LGCloudAgent binPath= "c:\Program Files
(x86)\Micro Focus\Load Generator\bin\LGCloudAgent.exe" start= auto
```

9. Disable autostart for the LoadRunnerAgent service:

- sc config LoadRunnerAgent start= demand
- sc config remotemanagementagent start= demand

10. Run the EC2Config utility and set the Administrator password to **Random**.
11. Create an AMI from an Instance.
    - Set the Name according to **LR-PC_Load_Generator_<version>_<OS> template**, e.g. **LR-PC_Load_Generator_12.60_Win2012R2**
    - Set the Description to **"LR LG, PC Compatible"** and **"version{<version>}"** template, e.g. **"LR LG, PC Compatible, version{12.60}"**
12. Set the AMI to public if required.
13. Copy the image to all required regions. Ensure each new image has the correct tags and permissions set after it reaches the active state.

### Amazon EC2 - Ubuntu

To create a custom image for Amazon EC2 – Ubuntu:

1. Launch a new instance from the cloud console, using an **Ubuntu Server 12.04 x64** image.
2. Connect to the instance that you just created, signing in as **ubuntu**.
3. Run the following commands to update the image:

```
sudo apt-get update
sudo apt-get upgrade
```

```
sudo apt-get -y install libc6-i386 lib32stdc++6 lib32ncurses5
gsoap:i386 mc
```

4. Upload the Load Generator standalone installation to the Linux machine. You can find the installation files in the LoadRunner installation package's folders:

```
\Linux\Web_Installers_64\InstData\Linux\VM
or
\Linux\Web_Installers_32\InstData\Linux\VM
```

5. Execute the following command for the uploaded files:

```
chmod a+x inst64.bin installer.csh installer.sh unzip
```

6. Run the following:

```
bash installer.sh
```

7. Install the standalone load generator, accepting all of the defaults. Skip the Certificate Installation step and the Load Generator startup.
8. Insert the following lines into **/etc/rc.local**:

- export LD_LIBRARY_PATH=/opt/MF/MF_LoadGenerator/bin
- export PRODUCT_DIR=/opt/MF/MF_LoadGenerator
- export M_LROOT=/opt/MF/MF_LoadGenerator
- /opt/MF/MF_LoadGenerator/bin/LgCloudAgent

9. Create an AMI from the instance.
   - Set the Name according to **LR-PC_Load_Generator_<version>_<OS> template**, e.g. **LR-PC_Load_Generator_12.60_Ubuntux64**
   - Set the Description to **"LR LG, PC Compatible"** and **"version{<version>}"** template, e.g. **"LR LG, PC Compatible, version{12.60}"**
10. Set the AMI to public if necessary.
11. Copy the AMI to all required regions. Make sure to copy all tags and permissions to each new image.

### Google Cloud - Windows

To create a custom image for Google Cloud – Windows:

1. Install the Google Compute SDK. It can be downloaded from https://cloud.google.com/sdk/.
2. Using the SDK's Web interface from your browser:

a. Define a network with any name (in our example, **network1**) and add the following rules:

| Name | Source tag / IP range | Allowed protocols / ports | Target tags |
|---|---|---|---|
| default-allow-lg | 0.0.0.0/0 | tcp:54345 (LR agent port) | hp-lr-lg |
| default-allow-rdp | 0.0.0.0/0 | tcp:3389 | rdp |

b. Create an instance:
   i. Provide a name for the instance (for our example **instance1**).
   ii. Specify a region for the instance (for our example, **us-central1-a**).
   iii. Specify a machine size **g1-small** (1 vCPU, 1.7 GB RAM), or higher.
   iv. Specify **Windows Server 2012 R2 Standard x64** as the operating system.
   v. Select **Create New image -> Management, Disk, Networking, Access & Security Options -> Management** and specify the tag "**rdp**".
   vi. Select **Create New image -> Management, Disk, Networking, Access & Security Options -> Networking** and specify **network1** as the network.
   vii. Confirm that you want to create the instance.

c. Connect to the instance that you just created via RDP.

d. Download Load Generator Standalone installation file to the instance, and install in Performance Center (PC) Mode.

e. When the Load Generator is installed:
   i. Open the Server Manager on the machine.
   ii. From the **Tools** menu, select **Computer Management**.
   iii. Under **System Tools > Local Users and Groups**, select **Groups**.
   iv. On the right, open the properties of the **Administrators** group.
   v. In the **General** tab, under **Members**, remove **IUSR_METRO**.
   vi. Click **OK**.

f. Turn off firewalls using the following command:

```
netsh advfirewall set allprofiles state off
```

g. Register the LGCloudAgent service using the following command:

```
sc create LGCloudAgent binPath= "c:\Program Files
(x86)\Micro Focus\Load Generator\bin\LGCloudAgent.exe" start= auto
```

h. Disable autostart for the LoadRunnerAgent service:

```
sc config LoadRunnerAgent start= demand
sc config remotemanagementagent start= demand
```

i. Run the System Preparation Tool using the Generalize and Shutdown options:

```
cd %windir%\system32\sysprep
sysprep (generalize, shutdown)
```

3. Using the Google Compute SDK console:
   a. Authenticate yourself:

```
gcloud auth login
```

   b. Set current project:

```
gcloud config set project <project ID>
```

4. Using the SDK's Web interface, determine the Project ID:
   a. Terminate the instance using the SDK console:

```
gcloud compute instances delete instance1 --keep-disks boot
```

   b. Create an image from the instance disk.

```
gcloud compute images create <new image name> --description "LR LG,
PC Compatible, version{12.60}, windows" --source-disk instance1 --
source-disk-zone us-central1-a
```

### Google Cloud - Ubuntu

To create a custom image for Google Cloud – Ubuntu:

1. Install the Google Compute SDK. It can be downloaded from https://cloud.google.com/sdk/.
2. Using the SDK's Web interface from your browser:
   a. Define a network with any name (in our example, **network1**) and add the following rules:

| Name | Source tag / IP range | Allowed protocols / ports | Target tags |
|---|---|---|---|
| default-allow-lg | 0.0.0.0/0 | tcp:54345 (LR agent port) | hp-lr-lg |
| default-allow-ssh | 0.0.0.0/0 | tcp:22 | ssh |

   b. Create an instance:
      i. Provide a name for the instance (for our example **instance1**).
      ii. Specify a region for the instance (for our example, **us-central1-a**).
      iii. Specify a machine size **g1-small** (1 vCPU, 1.7 GB RAM), or higher.
      iv. Specify **Ubuntu 14.04 x64** as the operating system.

v. Select **Create New image -> Management, Disk, Networking, Access & Security Options -> Management** and specify the tag "**ssh**".

vi. Select **Create New image -> Management, Disk, Networking, Access & Security Options -> Networking** and specify **network1** as the network.

vii. Confirm that you want to create the instance.

c. Connect to the instance that you just created via SSH.

d. Download Load Generator Standalone installation file to instance1 from the LoadRunner installation package:

```
\Linux\Web_Installers_64\InstData\Linux\VM
or
\Linux\Web_Installers_32\InstData\Linux\VM
```

e. Run the following commands to update the image:

```
sudo apt-get update
sudo apt-get upgrade
```

f. Upload the Load Generator standalone installation to the machine. Execute the following command on the uploaded files:

```
chmod a+x inst64.bin installer.csh installer.sh unzip
```

g. Install the load generator machine's prerequisites:

```
sudo apt-get install libc6-i386 lib32stdc++6 lib32ncurses5
libkrb5support0:i386 libkeyutils1:i386 gsoap:i386 mc libglib2.0-
0:i386
```

h. Install the Load Generator, accepting all of the defaults. Skip the certificate installation and set the load generator to startup.

i. Insert the following lines into **/etc/rc.local**:

- export LD_LIBRARY_PATH=/opt/MF/MF_LoadGenerator/bin
- export PRODUCT_DIR=/opt/MF/MF_LoadGenerator
- export M_LROOT=/opt/MF/MF_LoadGenerator
- /opt/MF/MF_LoadGenerator/bin/LgCloudAgent

j. Shut down the instance and verify that it has stopped.

3. Using the SDK's console:
   a. Authenticate yourself with the following command:

```
gcloud auth login
```

   b. Set the current project.

```
gcloud config set project <project ID>
```

   c. Terminate the instance:

```
gcloud compute instances delete instance1 --keep-disks boot
```

   d. Create an image from the instance disk.

```
gcloud compute images create <new image name> --description "LR LG,
PC Compatible, version{12.60}, linux" --source-disk instance1 --
source-disk-zone us-central1-a
```

### DigitalOcean - Ubuntu

To create a custom image for DigitalOcean – Ubuntu:

1. Create new instance from the cloud console using the Ubuntu 14.04 x64 image.
2. Connect to the instance using the credentials from your mailbox.
3. Install the required packages:

```
dpkg --add-architecture i386 && apt-get update && apt-get -y upgrade &&
apt-get -y install libc6-i386 lib32stdc++6 lib32ncurses5
libkrb5support0:i386 libkeyutils1:i386 gsoap:i386 mc libglib2.0-0:i386
```

4. Upload the standalone load generator files. The files may be found in installation package, in the following locations:

```
\Linux\Web_Installers_64\InstData\Linux\VM
or
\Linux\Web_Installers_32\InstData\Linux\VM
```

5. Execute following command for uploaded files:
   chmod a+x inst64.bin installer.csh installer.sh unzip
6. Run the following:

```
bash installer.sh
```

7. Install the Load Generator, accepting all of the defaults. Skip the Certificate Installation step and the Load Generator startup.
8. Insert the following lines into **/etc/rc.local**:

   a. export LD_LIBRARY_PATH=/opt/MF/MF_LoadGenerator/bin
   b. export PRODUCT_DIR=/opt/MF/MF_LoadGenerator
   c. export M_LROOT=/opt/MF/MF_LoadGenerator
   d. export CLOUD=DigitalOcean
   e. /opt/MF/MF_LoadGenerator/bin/LgCloudAgent

9. Power off. Wait for the instance to be in the **Off** state in the Web UI.
10. Create a snapshot (Select droplet, and select the 'snapshots' menu item, fill in a name and click the button). The name should contain "LR LG" and version {12.60} or any other relevant version number. Wait while the snapshot is processed.
11. Copy the image to other regions. Go to https://cloud.digitalocean.com/images, select the new image, click **Edit region availability** and transfer it to other regions.
12. Make sure that all metadata and tags are copied to each of the images.

## Add Load Generators to a Scenario

The following steps describe how to add a load generator to a scenario. The procedure varies slightly depending on whether you are adding an on-premise or cloud-based load generator.

**To add load generators to your scenario:**

1. Open the Load Generators dialog box.

   In the **Design** tab, click the **Load Generators** button . For details, see "Load Generators Dialog Box" on page 1019.

2. Add a load generator machine

   **Add an on-premise load generator**

   a. Click **Add**, and enter the details of the load generator. For details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007.
   b. Click **OK**. When you add an on-premise load generator, it remains in the **Down** state until it is connected.

   **Provision new cloud-based load generators**

   The process of creating a new load generator in a cloud account is known as *provisioning* the load

generator.

Controller installs a default system configuration, or image, on the cloud. Alternatively you can apply a custom image. For more information, see "Create Custom Image for Load Generators on the Cloud" on page 992.

a. **Prerequisite:** Set up your cloud account and Controller for provisioning load generators on the cloud. For details, see "Load Generators on the Cloud" on page 987.

b. Click **Add From Cloud > Provision New LG**. The Create Cloud Load Generator dialog box opens.

c. From the list of available cloud accounts, select the cloud account which will host the new load generator.

d. Specify the number of load generators to provision.

> **Tip:** Increasing the number of load generators allows you to provision more than one machine with the same parameters.

e. To automatically terminate the machines after the scenario run and after all the data has been collated, select **Terminate the load generator when load test is finished**.

f. Click **More** to specify provider-specific parameters and images. Initially, default parameters are selected. These parameters are unique for each cloud provider. For details, see "Create Cloud Load Generator Dialog Box" on page 1024.

> **Tip:** In subsequent provisioning, LoadRunner uses the connection settings that you last applied to a load generator, not necessarily a cloud-based machine.

- In the **Machine Settings** section, select the relevant settings. These settings differ for each of the supported cloud providers. The default values are the most popular and commonly used settings.
- In the **Connections** section, accept the default network profile, or select another one from the drop-down list. Click the button to open the Network Profile Manager to create a new profile. For details, see "Network Profile Manager Dialog Box" on page 1028. All of the connection settings will be applied to the new load generators that will be provisioned. For example, if you specify **Port 6080**, the load generator will be created on port 6080 and listen on the same port.

g. Verify the provisioning parameters (amount, termination policy, and machine parameters), and click **Provision** to provision the specified load generators

During provisioning, the load generator status is **In Progress**. This status indicates provisioning of the test machine on the cloud and launching the OS and load generator on this machine. This process can take a few minutes.

Once the load generator status changes to **Down**, it can be used for running a scenario. You can verify its availability by clicking the **Connect** button.

> **Caution:** If you want to delete a cloud load generator from the scenario, make sure you select the correct deletion option. For details, see "Manage Load Generators on the Cloud" on page 990.

**Add an existing cloud-based load generator**

You can use running instances of cloud load generators in your scenario, either provisioned through the Controller, or manually created.

a. **Prerequisite:** Ensure that you have provisioned load generators in your cloud account. For details, see "Provision new cloud-based load generators" on page 1002.
b. Click **Add From Cloud > Use Existing LG**. The Use Cloud Load Generator dialog box opens. For details, see "Use Cloud Load Generator Dialog Box" on page 1027.
c. From the list of cloud accounts, select the cloud account that hosts the desired load generator machine. A list of available load generators is displayed.
d. Select the load generators that you want to add to the scenario in the **Available Machines** section.
e. Select a **Network Profile** if you defined them beforehand. The network profile indicates the connection information for the load generator, such as the connection mode, proxy, and ports. To create a new network profile, close this dialog box and choose **Tools > Network Profile Manager**. For details, see "Network Profile Manager Dialog Box" on page 1028. If the network profile that you chose indicates a different port for an existing load generator (cloud or on-premise), you also need to manually set it on the load generator machine.
f. Click **Add Machine**.

3. Check your machine for Shellshock vulnerability.

   For Linux load generators, check your machine for the Bash vulnerabilities. For details, see "Troubleshooting and Limitations for Controller" on page 1265.

## How to Modify Load Generator Settings

The following procedure describes how to modify the settings for a load generator.

1. Open the scenario . The scenario is displayed in the **Design** tab.
2. Click the **Load Generators** button on the Controller toolbar.
3. Select a load generator, and click **Details**.
4. Modify the details displayed in the configuration tabs at the bottom of the dialog box. For details, see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007.

## Load Balancing

Load balancing evenly distributes the load generated by Vusers among the requested load generators, ensuring an accurate load test.

When a Windows load generator's CPU usage becomes overloaded, Controller receives a message and stops loading Vusers on the overloaded load generator, and automatically distributes them among the other load generators taking part in the scenario. Only where there are no other available load generators in the scenario does Controller stop loading Vusers.

You can monitor the status of a machine's CPU usage using the icons in the Load Generators dialog box (see "Load Generators Dialog Box" on page 1019). When the CPU usage of a load generator becomes problematic, the icon to the left of the load generator name contains a yellow bar. When the machine becomes overloaded, the icon contains a red bar.

> **Note:** Load balancing is available only in manual scenarios in the percentage mode, and in goal-oriented scenarios.

## How to Connect/Disconnect a Load Generator

This task describes how to manually connect or disconnect a load generator.

1. On the **Controller** toolbar, click **Load Generators** .
2. Select a load generator in the list and click **Connect/Disconnect**.

## How to Connect to a Linux Load Generator Without Using RSH

This task describes how to connect to a Linux load generator without using RSH.

1. On the Linux load generator, make sure that the agent daemon (m_agent_daemon) is running. If not, launch it by running the following command from **<load generator installation directory>/bin**:

   > **Example:** `m_daemon_setup -install`

   If successful, you will receive the following message:

   ```
   m_agent_daemon <process ID>
   ```

   The agent now keeps running, even if the user is logged off. It only stops running if you run the command explained below, or if you reboot the machine.

   > **Note:** If you look at the log file **m_agent_daemon[xxx].log** in the temp folder, you may see communication errors, even if the installation succeeded. These messages appear because the LoadRunner agent always tries to open Port #443 (because any agent can be an MI

> Listener, and the MI Listener always listens to this port), and in Linux machines, this port cannot be opened by any user except for the root user. However, this does not interfere with using this agent for the load generator.

2. In the Controller, in the **Load Generator Information** dialog box **> Linux Environment** tab, select the **Don't use RSH** option. Connect as usual. For details, see "Load Generator Configuration > Linux Environment Tab" on page 1011.

### To stop the agent daemon:

Run the following command from the **<LoadRunner root>/bin** folder:
`m_daemon_setup -remove`

This stops the m_agent_daemon. If successful, you receive the following message:
`m_agent_daemon is down.`

## How to Change the Load Generator Temporary Folder [Linux]

When you run a scenario from Controller, and the scenario includes a load generator on a Linux machine, by default Vuser logs are written under "/tmp/brr_..."

This task describes how to change the load generator temporary folder on Linux machines.

### On the Linux load generator:

1. Close the agent.

2. Verify that the new location [folder] exists.

3. Run "setenv MERC_TEMP_DIR <new location>" where <new location> is the new location of the temporary folder.

4. Restart the agent.

## Linux Environment Variables

To work with a load generator in a Linux environment, your Linux startup configuration file needs to include specific environment variables. To set the environment variables, users need to add the **env.csh** script to their startup configuration file.

| Linux Users | Startup Configuration File |
|---|---|
| C shell | .cshrc |
| Bourne and Korn shell | .profile |

Add the following line in the startup configuration file:

```
source <load generator installation directory>/env.csh
```

For example:

```
source /opt/MF/MF_LoadGenerator/env.csh
```

## Add New Load Generator/Load Generator Information Dialog Box

The Add New Load Generator dialog box enables you to add a new local load generator to the scenario. The Load Generator Information dialog box enables you to view and edit information about the selected load generator.

| | |
|---|---|
| **To access** | Controller toolbar **>** .<br>• **Add New Load Generator dialog box:** Click **Add**.<br>• **Load Generator Information dialog box:** Select a load generator and click **Details**. |
| **Important information** | After adding a load generator, it appears in the Load Generators list with a **Down** status. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | "Load Generators" on page 983 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| <br>(Add New Load Generator dialog box only) | **More/Less.** Shows/hides the tabs where you configure the load generator's details. For details, see below. |

| UI Element | Description |
|---|---|
| **Enable load generator to take part in the scenario** | When selected, enables the load generator to participate in the scenario.<br>When cleared, the load generator is disabled, and therefore does not participate in the scenario. This is useful in the following cases:<br>• If a load generator is unavailable for a particular scenario run, you can exclude it temporarily instead of removing it entirely from the list of load generators that are available for the scenario.<br>• You can disable load generators to isolate a specific machine to test its performance. |
| **Name** | The name of the load generator (limited to 128 characters).<br>**Note:** For Dockerized load generators, then name must be in the format `<host machine name>_dkr_<port_number>`. For details, see "Configure Dockerized Load Generators" on page 985. |
| **Platform** | The platform on which the load generator is installed, either Windows or Linux. |
| **Temporary directory** | The location, on the load generator, where the Controller can store temporary files.<br>**Default:** If left empty, during a scenario run LoadRunner stores the temporary files on the load generator in a temporary folder specified by the load generator's TEMP or TMP environment variables. |

## Load Generator Configuration Tabs

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Connection Log tab** (Load Generator Information dialog box only) | Displays the standard output and standard errors generated as the Controller connects to the selected Linux load generator. You can change the command that the Controller sends to the remote bridge in order to connect to the load generator. See "Load Generator Configuration > Connection Log Tab" on page 1010.<br>**Available:** In Expert mode only. |
| **Runtime File Storage tab** | Enables you to specify the result folder for the performance data that LoadRunner gathers from each load generator during a scenario. See "Load Generator Configuration > Runtime File Storage Tab" on page 1014. |

| UI Element | Description |
|---|---|
| **Runtime Quota tab** | Enables you to specify the maximum number of Vusers that the load generator can initialize or stop simultaneously. See "Load Generator Configuration > Runtime Quota Tab" on page 1015. |
| **Connection tab** | Enables you to manage network profiles and run Vusers over a firewall. See "Load Generator Configuration > Connection Tab" below. |
| **Status tab** | Displays details about the status of the load generator. See "Load Generator Configuration > Status Tab" on page 1016. |
| **Terminal Services tab** | The Terminal Services Manager which enables you to distribute Vusers running in your load testing scenario on terminal servers. See "Load Generator Configuration > Terminal Services Tab" on page 1016. |
| **Linux Environment tab** | Enables you to configure the login parameters and shell type for each Linux load generator. See "Load Generator Configuration > Linux Environment Tab" on page 1011. |
| **Vuser Limits tab** (Add New Load Generator dialog box only) | Enables you to modify the maximum number of GUI, RTE, and other Vusers that the load generator can run. See "Load Generator Configuration > Vuser Limits Tab" on page 1017. |
| **Vuser Status tab** (Load Generator Information dialog box only) | Displays the status of all the Vusers connected to the load generator. See "Load Generator Configuration > Vuser Status Tab" on page 1018. |
| **Network Virtualization tab** | Allows you to apply network virtualization to your scenario. See "Load Generator Configuration > Network Virtualization Tab" on page 1013. |

## Load Generator Configuration > Connection Tab

This tab enables monitoring or running Vusers over a firewall. It also allows you to select a connection profile, for customizing the ports used by the load generator and MI listener.

| | |
|---|---|
| **To access** | Controller toolbar **>** > **Add** or **Details** |

| | |
|---|---|
| **Important information** | • If the load generator is connected, you cannot change values in this tab. To disconnect a load generator, select it in the Load Generators dialog box and click **Disconnect**. The load generator status changes to **Down**, and you can change the settings.<br>• If the load generator is **localhost**, this tab is disabled. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | • "Network Profile Manager Dialog Box" on page 1028 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Choose Network Profile** | A drop-down list of network profiles defining the connection properties of the load generator. Each profile defines whether to connect directly or via an MI Listener. You can also provide information to connect to the load generator via a proxy. |
| ... | **Browse.** Opens the Network Profile Manager. For details, see "Network Profile Manager Dialog Box" on page 1028. |
| **Enable Monitoring over Firewall** | Enables monitoring over firewall for this load generator. |

## Load Generator Configuration > Connection Log Tab

This tab displays the standard output and standard errors generated as the Controller connects to the selected Linux load generator and enables you change the command that the Controller sends to the remote bridge in order to connect to the load generator.

| | |
|---|---|
| **To access** | **Controller** toolbar **>** > **Details** |
| **Important information** | This tab is displayed only when the Controller is in Expert mode. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | • "Load Generators" on page 983<br>• "Expert Mode" on page 1068 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Bridge cmd** | The command sent by the Controller to the remote bridge in order to connect the Linux load generator.<br><br>**Note:** This command replaces the default bridge command sent by the Controller. |
| **Rsh standard errors** | Displays rsh standard errors as the Controller connects to the selected Linux load generator. |
| **Rsh standard output** | Displays rsh standard output as the Controller connects to the selected Linux load generator. |

## Load Generator Configuration > Linux Environment Tab

This tab enables you to configure the login parameters and shell type for each Linux load generator.

| | |
|---|---|
| **To access** | **Controller toolbar >** **> Add** or **Details** |
| **Important information** | • This tab is not available for a load generator running Vusers or monitoring over a firewall.<br>• Editable only when the load generator is on a Linux platform. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004<br>• "How to Connect to a Linux Load Generator Without Using RSH" on page 1005 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Login as** | The user's credentials for logging in to the Linux environment:<br>• **Name.** If the load generator is Linux-based, set the login information for the load generator.<br>**Default:** LoadRunner uses your Windows user name for the Linux login. That is, if your Windows login is **lrunner**, the Controller logs in to the load generator as lrunner. To log in to a Linux-based load generator using a different login name, select **Name** and specify the desired Linux login name.<br>Using this option, you can log in to the Windows Controller as **bill** and connect to the Linux load generator as **mike**. However, you should make sure that mike allows bill to log in using his name. This can be done by adding the line **"+ bill"** at the beginning of mike's .rhosts file.<br>• **Use lower case for login names.** Instructs LoadRunner to use lower case names during login to avoid case-sensitive issues with the Linux operation system.<br>• **Local User.** (Expert mode only) Linux load generators that use the rsh shell to establish a connection as the current Windows user (due to security considerations). To "mislead" rsh and log in as a user other than the current Windows login, select **Local user** and specify the desired Linux login name. Because modifying the local user name is a security breach for rsh, this option should be used only when you encounter a problem connecting to the remote machine. |

| UI Element | Description |
|---|---|
| **Shell Settings** | The Linux shell settings for the remote Linux load generator.<br>**Default:** The Controller connects remotely to the Linux load generator without using rsh (remote shell).<br>• **Don't use RSH.** Connects to the remote load generator without using RSH. In this case, you need to activate the agent daemon on the load generator manually.<br>**Note:** If you do want to connect using RSH, clear this check box, make sure that RSH is enabled on the load generator, and make sure that the agent daemon is not already running on the load generator. If the agent daemon is running, stop it by running the following command from the **<LoadRunner root>/bin** folder:<br>`m_daemon_setup -remove`<br>• **Default shell.** The default shell on the Linux load generator: **csh** (C Shell—the default), **bsh** (Bourne Shell), or **ksh** (Korn Shell).<br>To work with the load generator, your Linux startup configuration file needs to include specific environment variables. For details, see "Linux Environment Variables" on page 1006.<br>• **Initialization command.** Command line options for LoadRunner to use when logging in to a Linux system. This initialization command runs as soon as the shell opens.<br>**Example:** You could select **ksh** and use the following initialization command: `source .profile;` |

## Load Generator Configuration > Network Virtualization Tab

This tab shows the network virtualization location for your scenario.

| | |
|---|---|
| **To access** | **Controller toolbar >** **> Add and click More** or **Details** |
| **Important information** | • To enable network virtualization, see "Virtual Locations Settings Dialog Box" on page 1243. |
| **Relevant tasks** | • "How to Run a Scenario with Network Virtualization" on page 1240<br>• "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | "Network Virtualization Integration" on page 1237 |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Network Virtualization Settings button** | Opens the "Virtual Locations Settings Dialog Box" on page 1243. You use this dialog box to define new Network Virtualization locations or change the working mode (**Per Group** or **Per Load Generator**). It also provides a link to the Network Virtualization screens, allowing you to configure the virtualization properties. |
| **Default Virtual Location** | The default virtual location for the Load Generator. (Available only when Network Virtualization is enabled and **Per Load Generator** mode selected.)<br>For details, see " Network Virtualization Locations" on page 1238. |
| **Description** | The description of the virtual location as entered in the"Virtual Locations Settings Dialog Box" on page 1243. |

## Load Generator Configuration > Runtime File Storage Tab

This tab enables you to specify the results folder for the performance data that LoadRunner gathers from this load generator during a scenario run.

| | |
|---|---|
| **To access** | **Controller toolbar >** **> Add** or **Details** |
| **Important information** | The folder specified here stores the result files gathered on the selected load generator. You can specify a global results folder using the Options dialog box. For details, see "Options > Runtime File Storage Tab" on page 1081.<br>The following guideline apply:<br>• If the settings specified here differ to the global load generator settings, the settings specified here take preference.<br>• If the load generator is **localhost**, LoadRunner stores the scripts and results on a shared network drive and the options on this tab are all disabled.<br>• If you are monitoring over a firewall, the settings in this tab are not relevant. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | • "Runtime File Storage Locations" on page 1068<br>• "Options > Runtime File Storage Tab" on page 1081 |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Scripts and results stored** | The location in which to store the results of the scenario run and/or Vuser scripts gathered from the selected load generator during a scenario run:<br>• **As defined in Tools > Options > Runtime File Storage.** Stores the results as specified in the global settings.<br>• **In temporary directory on <load generator name>.** Instructs LoadRunner to save the runtime files on a hard drive of the load generator machine. For information on setting the temporary directory, see "Options > Runtime File Storage Tab" on page 1081.<br>• **On a shared network drive.** Instructs LoadRunner to store the results on a shared network drive. To use this option, you save the Vuser scripts at a shared network location, and define a shared network location for saving the scenario results from the load generator. For details, see "Runtime File Storage Locations" on page 1068. |

## Load Generator Configuration > Runtime Quota Tab

This tab enables you to specify a maximum number of Vuser types that the load generator should initialize or stop simultaneously, so as to reduce load on the load generator.

| | |
|---|---|
| **To access** | Controller toolbar **>** > **Add** or **Details** |
| **Important Information** | The settings specified here are relevant for the selected load generator.<br>You can set runtime quotas for all load generators in a scenario from the **Tools > Options > Runtime Settings** tab. For details on setting global runtime quotas, see "Options > Runtime Settings Tab" on page 1082.<br>**Note:** If the settings specified here differ to the global load generator settings, the settings specified here take preference for this particular load generator. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | • "Options > Runtime Settings Tab" on page 1082 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Defaults | Resets values to their defaults. |

| | |
|---|---|
| **Vuser Quota** | • **Number of Vusers that may be initialized at one time - <current load generator>.** The maximum number of Vusers that the current load generator can initialize simultaneously.<br>**Default:** 50<br>**Maximum value:** 999<br>• **Limit the number of users that may be stopped at one time to.** The maximum number of Vusers that the current load generator can stop simultaneously.<br>**Default:** 50 |

## Load Generator Configuration > Status Tab

This tab displays details about the status of the load generator.

| | |
|---|---|
| **To access** | Controller toolbar **>** > **Details** |
| **Relevant tasks** | "How to Modify Load Generator Settings" on page 1004 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Details** | Error and other runtime information about the selected load generator. |
| **Load Generator Status** | The status of the load generator. |

## Load Generator Configuration > Terminal Services Tab

This tab displays the Terminal Services Manager which enables you to distribute Vusers running in your load testing scenario on terminal servers.

| | |
|---|---|
| **To access** | **Controller toolbar >** **> Add** or **Details** |
| **Important information** | This feature is not supported if the load generator is located over a firewall. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | "LoadRunner Terminal Services Manager" on page 1051 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| Defaults | Resets values to their defaults. |

| | |
|---|---|
| **<Connection Options>** | • **Connect to existing Terminal Services Sessions.** Enables connection to existing (open) terminal sessions.<br>If you select this option, you must open a terminal client session manually for each terminal that you want to run Vusers on during the scenario.<br>• **Create new Terminal Services Sessions.** Enables the Controller to open and close terminal sessions automatically. When choosing this option you also specify the following:<br>  • **User name**, **Password**, and **Domain.** The credentials required for automatic login. |
| **Enable Terminal Services Manager** | Enables the Controller to manage load automatically using terminal sessions on the load generator.<br>**Note:** When enabled, you can only see the load generator's name, without adding any extra references.<br>If you need to use three load generator sessions, **my_machine**, **my_machine:1**, and **my_machine:2**, then in the load generator list, you need only insert the load generator, **my_machine**, and enable the Terminal Services Manager for three terminals (**Number of terminals = 3** - see below).<br>Example:<br>If not enabled, you need to insert each of the three terminals as separate load generators: **my_machine**, **my_machine:1**, and **my_machine:2**. |
| **Maximum number of Vusers per terminal** | The maximum number of Vusers that you want to run in a terminal session. This depends on the Vuser type used in the script.<br>**Default:** 50<br>**Example:** For GUI Vusers, the maximum is **one** Vuser for each terminal session. |
| **Number of terminals** | The number of terminals you want to use in your scenario. You must open a terminal client session for each terminal on which you want to run Vusers during the scenario.<br>**Default:** 2 |

## Load Generator Configuration > Vuser Limits Tab

This tab enables you to modify the maximum number of GUI, RTE, and other Vusers that the load generator can run.

| | |
|---|---|
| **To access** | Controller toolbar > > **Add** or **Details** |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Defaults | Resets values to their defaults. |
| **Available Types** | The types of Vusers the load generator should run.<br>The types of Vusers are:<br>• **GUI**<br>• **RTE**<br>• **Other Vusers** |
| **Maximum Active** | The maximum number of each type of Vuser that the load generator can run.<br>**Defaults:**<br>• GUI: 1<br>• RTE: 1000<br>• Other Vusers: 5000<br>**Note:** The maximum number of active Vusers that you specify must not exceed the number of Vusers that you are licensed to run. You can check your Vuser licensing limitation using the LoadRunner License Utility: from the Windows Start menu, go to **Micro Focus > LoadRunner > License > LoadRunner License Utility**. |

## Load Generator Configuration > Vuser Status Tab

This tab enables you to view the status of all the Vusers connected to the load generator.

| | |
|---|---|
| **To access** | **Controller toolbar >** **> Add** or **Details** |
| **Important information** | This tab is visible only when the load generator is connected. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | "Network Virtualization Integration" on page 1237 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|

| | |
|---|---|
| **<Vuser Status>** | The status of the Vusers:<br>• **Pending.** Vusers are waiting to initialize.<br>• **Initializing.** Vusers are in the initialization state.<br>• **Active.** Vusers are actively running in the scenario. |
| **GUI** | The number of GUI Vusers that are in the **Pending**, **Initializing**, and **Active** states. |
| **Other Vusers** | The number of Vusers—other than GUI and RTE Vusers—that are in the **Pending**, **Initializing**, and **Active** states. |
| **RTE** | The number of RTE Vusers that are in the **Pending**, **Initializing**, and **Active** states. |
| **Totals** | The total number of Vusers that are in the **Pending**, **Initializing**, and **Active** states. |

## Load Generators Dialog Box

This dialog box enables you to manage the load generators defined for the scenario.

| | |
|---|---|
| **To access** | • Open a scenario and click on the Controller toolbar<br>• Select **Scenario > Load Generators** |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | • "Load Generators" on page 983<br>• "Manage Cloud Accounts Dialog Box" on page 1022 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Connect / Disconnect** | • **Connect.** Instructs the Controller to connect the selected load generator. When connected, the status of the load generator changes from **Down** to **Ready**.<br>• **Disconnect.** When the load generator is connected, the button automatically changes to **Disconnect**. When clicked, the status of the load generator changes to **Down**. |

| | |
|---|---|
| **Add** | Opens the Add New Load Generator dialog box where you specify and add a new local load generator for the scenario.<br>See "Add New Load Generator/Load Generator Information Dialog Box" on page 1007.<br><br>**Note:** When you add a local load generator, the status of the load generator is set to **Down** until it is connected. |
| **Add From Cloud** | Enables you to add a cloud-based load generator to the scenario. The load generator can be either a new load generator or a load generator that already exists in the cloud.<br><br>**Note:** When you add a local load generator, the status of the load generator is set to **Down** until it is connected. |
| **Delete** | Removes the load generator from the list of load generators that are available to run Vuser scripts during the scenario.<br><br>**Note:** If the selected load generator is cloud-based, you can select to delete the load generator from the cloud account as well.<br><br>A load generator can be removed only when it is disconnected. |
| **Reset** | Attempts to reset a failed load generator to the **Down** status. |
| **Details** | Opens the Load Generator Information dialog box where you can view and modify information about the selected load generator. See "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |
| **Disable/Enable** | Instructs the Controller to disable or enable the selected load generator. When a load generator is disabled, the details of the load generator are displayed in gray. |
| **<icons>** | Display the status of the load generator's CPU usage:<br> The load generator is available to run additional Vuser scripts.<br> There is a problem with the CPU usage of the load generator.<br> The load generator is overloaded.<br> The status of the load generator is unknown. This applies when the specified load generator name or IP address cannot be resolved.<br><br>**Note:** These icons are only displayed for Windows-based load generators. |

<table>
<tr>
<td><Load Generator table></td>
<td>
Displays the following information:
<ul>
<li>Name. The name or IP address of the load generator. You can rename load generators that you already provisioned. Renaming is only supported for Amazon AWS providers.
<br>Note:
<ul>
<li>You can rename cloud-based load generator machines that you already provisioned. The renaming affects the provider side too, so that the new name will appear in the list of available machines in the "Use Cloud Load Generator Dialog Box" on page 1027.</li>
<li>Renaming cloud-based load generator machines will not affect the IP address that was provisioned for that machine. For physical load generators, however, this name is used to establish a connection with the machine, and should not be modified.</li>
</ul>
</li>
<li>Status. The current status of the load generator
<table>
<tr><td>Ready</td><td>The load generator is connected, and is ready to run Vuser scripts. If the connection is secure, an icon indicates a secure connection.</td></tr>
<tr><td>Connecting</td><td>The Controller is in the process of connecting to the load generator.</td></tr>
<tr><td>Active</td><td>The load generator is running Vusers. If the connection is secure, an icon  indicates a secure connection.</td></tr>
<tr><td>Down</td><td>The load generator is not connected.</td></tr>
<tr><td>Failed</td><td>A connection with the load generator could not be established.</td></tr>
<tr><td>In Progress</td><td>The load generator is being provisioned. This status only applies to cloud-based load generators.</td></tr>
<tr><td>Terminating</td><td>The load generator is being terminated by the provider. This status only applies to cloud-based load generators.</td></tr>
</table>
</li>
<li>Platform. The type of platform on which the load generator is running. The platform indicates whether the load generator is Windows-based, Linux-based, or cloud-based. The LoadRunner version of the load generator must match the version of the Controller. You can determine the version of the load generator from the Image string. To determine the version of the Controller, select Help > About.</li>
</ul>
</td>
</tr>
</table>

| | |
|---|---|
| | **Note:** This field may initially display **Windows-based**, even for machines provisioned for Linux. The field will show **Linux-based** after the provider receives confirmation of the need for a Linux machine.<br>• **Type.** Indicates whether the load generator is local or cloud-based. For cloud-based load generators, **Type** displays the name of the cloud provider.<br>• **Network Profile.** The network profile assigned to this load generator. The network profile bundles all of the network connectivity settings, including proxies and ports, into a single entity. For details, see "Network Profile Manager Dialog Box" on page 1028.<br>• **Virtual Location.** The virtual location to emulate (only visible when Network Virtualization is installed). Click the cell to select a location from the drop-down list. To clear the value, select **None**,<br>• **Details.** If the connection between the Controller and the load generator fails, displays details about why the connection failed. |
| **<Right-click menu>** | • **Save List As Default.** Saves the current list of load generators as the default list.<br>• **Load Default List.** Loads the default list of load generators.<br>• **Filter Hosts.** Enables you to filter the load generator list by status: **Active** , **Ready**, **Down**, or **Failed**.<br>• **Sort Hosts.** Enables you to sort the Load Generator table by name, status, or type. To sort the table in ascending/descending order, click the relevant table heading. |

## Manage Cloud Accounts Dialog Box

This dialog box enables you to manage the cloud accounts that can host load generators. By default, the dialog box displays a list of all those cloud accounts that can host load generators.

| | |
|---|---|
| **To access** | **Controller > Tools > Manage Cloud Accounts** |
| **Important information** | The default proxy is used for validating the cloud account. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002 |
| **See also** | • "Load Generators on the Cloud" on page 987<br>• "Use Cloud Load Generator Dialog Box" on page 1027<br>• "Create Certificates for Azure Cloud" on page 991 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|

| | |
|---|---|
| | Adds a cloud account to the list. |
| | Removes a cloud account from the list. |
| | Edit the details of the cloud account (visible in the expanded details of each cloud account). |
| **<Account Name>** | The name that you will use to identify the account in the Controller. You can modify the name if and when required. |
| **Username / Password** | Placeholder identifiers before an account is selected. |
| **Access Key ID/ Secret Access Key** | AWS (Amazon Web Service) identifiers that are used to manage IAM (Identity and Access Management) in order to manage users and their security credentials across Amazon's EC2 (Elastic Compute Cloud).<br>For details, see http://docs.aws.amazon.com/AWSSecurityCredentials/1.0/AboutAWSCredentials.html#AccessCredentials. |
| **Subscription ID/ Path to certificate** | Identifiers that are used to manage cloud subscriptions for Microsoft Azure. For details, see https://manage.windowsazure.com.<br>For details, see "Create Certificates for Azure Cloud" on page 991. |
| **Wait ... hours before terminating the load generator after a test run or collation that did not succeed** | Lets you specify how long to wait after the post-run collation failed, before terminating the cloud-based load generator. This delay gives you time to access information on the load generator and determine why the collation failed. When a load generator is removed from the cloud, all data saved as part of the load generator is deleted.<br>You can enter a value from 0 through 168, and specify two digits to the right of the decimal point. For example, a value of 10.50 waits ten and a half hours.<br>**Note:** This setting is only visible if your cloud provider supports this feature. |
| **Provider Site** | A direct hyperlink to the cloud provider's website. |

| | |
|---|---|
| **Choose network profile** | The network profile defines the connection between the Controller and the cloud provider.<br><br>**Note:** A network profile is used to define the connection properties for the Controller with cloud providers and the connection with Load Generators.<br><br>You can select the default connection ("LoadRunner Default), where the settings are taken from the Internet Explorer proxy settings. Otherwise, fill in the proxy settings section of the Network Profile Manager to define the connection with the cloud provider.<br><br>For details of how to create a network profile, see "Network Profile Manager Dialog Box" on page 1028. |

## Create Cloud Load Generator Dialog Box

The Create Cloud Load Generator dialog box enables you to deploy a new cloud-based load generator.

| | |
|---|---|
| **To access** | 1. Click on the Controller toolbar to open the **Load Generator dialog box**.<br>2. Click **Add from Cloud**.<br>3. Click **Provision New LG**. |
| **Relevant tasks** | "Add Load Generators to a Scenario" on page 1002 |
| **See also** | "Use Cloud Load Generator Dialog Box" on page 1027 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Provision** | Provision the load generator(s) for the scenario according to the last configuration used for the selected account. This button is only enabled after you select a cloud account.<br>If this is the first time you are provisioning load generators, the default configuration is used. |
| **<number of load generators>** | The number of cloud-based load generators to provision. |
| **Terminate the Load Generator when load test is finished** | Terminates the current deployment when the load test is complete. |

| | |
|---|---|
| | **More/Less.** Shows/hides the tabs where you configure the cloud load generator parameters and the network profile to be used by the load generator. LoadRunner saves your selections for future provisioning. For details, see below. |

## Create Cloud Load Generator Sections

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Machine Settings** (Amazon EC2) | The cloud machine parameters. You can configure the following parameters:<br>• **Region.** The region (geographic location) and the availability-zone (isolated location within the region).For example **us-west-1**.<br>• **Availability Zone.** The zone within the region in which the cloud machine is accessible.<br>• **AMI.** (Amazon Machine Image) A template that contains a software configuration for your server such as the operating system. In the case of LoadRunner, the image will also contain the load generator software. For information on creating custom cloud images, see "Create Custom Image for Load Generators on the Cloud" on page 992.<br>• **Security Group.** The security group is a firewall-like mechanism that controls traffic to or from its associated instances. If you omit this property, the cloud provider's default security group is used. The default security group does not allow the Controller to connect to load generators using its default port. Therefore, you must create your own security group and enable it for all of the relevant communication ports. This includes the LoadRunner port, default port (54345), and any port set by the communication channel infrastructure. (See "Network and Security Manager - Command Line Tool" on page 1163). In addition, if you are going to access your load generators manually, you need to enable the security group for RDP access. You can select multiple values if allowed by the vendor.<br>• **Key Pair.** (Optional - only if you want to connect remotely.) The key pair encrypts and decrypts your login information.<br>• **Elastic IP.** (EIP) A static IP address that lets you mask the failure of an instance or software by remapping the address to another instance, if your current instance fails. For more information, see the Amazon documentation.<br>• **Security Group.** The security group is a firewall-like mechanism that controls traffic to or from its associated instances. If you omit this property, the cloud provider's default security group is used. The default security group does not allow the Controller to connect to load generators using its default port. Therefore, you must create your own security group and enable it for all of the relevant communication ports. This includes the LoadRunner port, default port (54345), and any port set by the communication channel infrastructure. (See "Network and Security Manager - Command Line Tool" on page 1163). In addition, if you are going to access your load generators manually, you need to enable the security group for RDP access. You can select multiple values if allowed by the vendor.<br>• **Instance Type.** The hardware configuration of the host computer that will be provisioned. This will impact parameters such as storage and memory. For example, **m1.xlarge**.<br>• **Shutdown Behavior.** (For AWS cloud service only) How to handle the cloud machine when the scenario ends:<br>  • **Terminate.** Terminate the machine completely and remove it from the list of load |

| UI Element | Description |
|---|---|
| | generators at the end of the scenario run.<br>• **Stop.** Place the machine in *Stopped* mode and remove it from the list of load generators at the end of the scenario run. If you add this machine at a later time using the "Use Cloud Load Generator Dialog Box" below, it will be automatically started again.<br>Refer to the cloud provider's website for further information about these parameters. |
| **Machine Settings** (Microsoft Azure) | The cloud machine parameters. You can configure the following parameters:<br>• **Location.** The geographic location.<br>• **Image.** A template that contains a software configuration for your server such as the operating system. In the case of LoadRunner, the image should also contain the load generator software. For information on creating an image for Azure, see "Create Custom Image for Load Generators on the Cloud" on page 992.<br>• **Storage Account.** A specific storage area in your account.<br>• **Size.** The hardware configuration of the machine to be provisioned. This will impact the storage and memory.<br>• **User/Password.** Credentials for accessing the machine directly, for example, through Remote Desktop.<br>• **Additional Ports.** (Optional) A list of ports, separated by spaces, to be opened in the machine. For example, to connect to the machine using Remote Desktop, specify port 3389.<br>Refer to the cloud provider's website for further information about these parameters. |
| **Connections tab** | Shows the details of the network profile assigned to the load generator, such as the proxy information, connection mode, and ports.<br>• To use a network profile other than the default, select one from the drop-down list.<br>• To create a new profile, click the ... button to open the Network Profile Manager. For details, see "Network Profile Manager Dialog Box" on the next page. |

## Use Cloud Load Generator Dialog Box

The Use Cloud Load Generator dialog box enables you to add an existing cloud-based load generator to your scenario.

| | |
|---|---|
| **To access** | 1. Click on the Controller toolbar to open the **Load Generator dialog box**.<br>2. Click **Add from Cloud**.<br>3. Click **Use Existing LG**. |

| | |
|---|---|
| **Relevant tasks** | "Add Load Generators to a Scenario" on page 1002 |
| **See also** | • "Load Generators on the Cloud" on page 987<br>• "Create Cloud Load Generator Dialog Box" on page 1024 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Account** | A drop-down list of your cloud accounts. |
| **Show** | Indicates which machines to display in the list below: **All machines** associated with the selected account, or **Only LG machine**, those machines upon which a load generator is installed. |
| **Only include machines using the current SSL certificate** | Show only machines whose SSL certificate is compatible with the Controller certificate.<br><br>**Tip:** The cloud machine adds the certificate's hash value to its meta data when it is provisioned through the Controller. If you manually install the certificate on the load generator machine, it will not be shown. |
| **Available Machines** | A list of cloud-based machines available for the current account. The grid contains the following columns: **Name**, **Region/Availability Zone**, **Launch Time**, and **Image**. Click a column name to sort it by that criteria.<br><br>**Note:** When working with Azure cloud provider, the list may display only a partial list of the load generators that are actually available in the Azure console (even with **All machines** selected). |
| **Network Profile** | A drop-down list of the network profiles that were defined for the active scenario. This allows you to configure the network settings only once for all machines included in the scenario. For details, see "Network Profile Manager Dialog Box" below. |

## Network Profile Manager Dialog Box

This dialog box enables you to set up network profiles for your load generator. For each profile, you can allow communication over a firewall. You can also customize the ports used by the load generator and MI listener.

| | |
|---|---|
| **To access** | Do one of the following:<br>• **Load Generator dialog box**<br>a. Controller toolbar **>** Load Generators .<br>b. In the Load Generators dialog box, click **Add** or **Details**.<br>c. Select the **Connections** tab.<br>d. Click **Network Profile Manager**.<br>• **Tools > Network Profile Manager** |
| **Important information** | • If the load generator is connected, you cannot change values in its profile. To disconnect a load generator, select the load generator click **Disconnect**. The load generator status changes to **Down**, and you can change the settings.<br>• Network Profile names may only have English characters. |
| **Relevant tasks** | • "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Profiles Toolbar** | Allows you to manage the profiles in the list.<br> . Add a new profile.<br> . Delete the selected profile.<br> . Clone the selected profile.<br> . Set the selected profile as default.<br> . Import a previously saved profile.<br> . Export the selected profile to a file for future use. |
| **Network Profile Name** | A list of all of the defined profiles. |

| | |
|---|---|
| **Proxy Settings > Use proxy** | Allows you to access a load generator through a proxy server.<br>**Use the default HTTP proxy settings.**<br>• When running on Windows, the Internet Explorer proxy settings are used.<br>To use this option, your default browser cannot be Firefox. If it is, uninstall Firefox and then select another default browser.<br>• When running on Linux, the proxy values in the HTTP_PROXY or HTTPS_PROXY environment variables are used.<br>You must manually create the environment variables on the Linux computer where your script runs.<br>**Use custom proxy server.** Specifies the URL, port, and credentials for the proxy server.<br>**Note:** LoadRunner currently supports basic and NTLM proxy authentication. |
| **Connection Mode** | Indicates how to connect to the LoadRunner agent.<br>**Connect directly to Load Generator Agent.** (default) Connects to the load generator directly via the specified **Port**. The default port is 54345.<br>**Connect via MI Listener.** Connects to the load generator through the MI Listener. This is useful when the load generator is behind a firewall. You specify the **Name** and **Port** of the MI Listener that the load generator uses to communicate over the firewall. The default port is 50500. |
| **Enable SSL** | Enables a secure connection between the Controller and load generator. This is enabled automatically when using a proxy. |

## LoadRunner Agent Runtime Settings Dialog Box

By default, you need to manually log on to a machine before it can run Vusers. This dialog box allows to set up an automatic login so that it is not necessary to manually log on each time.

| | |
|---|---|
| **To access** | Do one of the following on the LoadRunner machine:<br>• From the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > Agent Runtime Settings Configuration**.<br>• Run **<LoadRunner root>\launch_service\bin\Magentconfig.exe**. |
| **Important information** | • When created, the Load Testing Agent Service starts with the **LocalSystem**.<br>• After configuring the automatic login, you must reboot and log on to the system at least once after the LoadRunner installation. |
| **See also** | The *LoadRunner Installation Guide* included on the LoadRunner installation media. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Allow virtual users to run on this machine without user login** | Allows an automatic login to the network from the agent machine, so that Vusers can run without any manual intervention.<br>**Domain.** The network domain upon which the user machine resides.<br>**User.** The user name required to run the Vusers.<br>**Password.** The password for the specified user. |
| **Manual log in to this machine** | Instructs LoadRunner to prompt you for a manual login for each Vuser run. |

# SSL Utility

The SSL Utility enables you to perform common **OpenSSL** functions from a user interface.

> **Note:** To open the **SSL Utility**, on the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > SSL Utility**.

## Converting Certificates

The **Convert Certificate** tab of the **SSL Utility** enables you to convert certificates from PKCS #12 and X.509 formats to PEM format.

**To convert a certificate to PEM format:**

1. Open the **SSL Utility**, and click the **Convert Certificate** tab.
2. In the **Source certificate file** field, select the file that contains the certificate that you want to convert. The following file types are supported:

| | |
|---|---|
| PKCS #12 files | .p12, .pxf |
| X.509 files | .cer, .crt, .der |

3. In the **Import password** field, enter the password for decrypting the certificate file.
4. In the **PEM file** field, enter the name and location of the .pem file that will be created.
5. In the **PEM pass phrase** field, enter a password for the new certificate. If this is not specified, the certificate will not be password-protected.
6. From the **Source format** list, select the format of the original certificate that will be converted. This is automatically determined unless a file with a non-standard extension is selected.
7. Click **Convert**.

> **Note:** This tab has the same functionality as the “x509” and “pkcs12” OpenSSL commands.

### Test Connection

The **Test Connection** tab of the **SSL Utility** enables you to capture the connection data from an SSL server and save it to a file <server name>_<port number>_ConnectionData.txt in the %temp% directory. Additionally, the file is displayed in a Notepad file when the test is completed.

To test a connection, enter the following on the **Test Connection** tab of the **SSL Utility**:

- **Host/Port.** Specifies the host and port to connect to. If not specified, an attempt is made to connect to the local host on port 4433.
- **Client certificate file.**The certificate to use if requested by the server. By default, a certificate is not required.
- **Client key file.** The private key to use. If not specified the certificate file will be used.
- **Password.** The client machine's password.

> **Note:** This tab has the same functionality as the "s_client" OpenSSL command.

### Remove Certificate Encryption

This tab of the **SSL Utility** enables you to remove encryption from a private key certificate.

To remove certificate encryption, enter the following information:

- **Source PEM.** Name of the file containing an encrypted private key and password.
- **Target PEM.** Name of the file that will be unencrypted.

> **Note:** This tab has the same functionality as the "pkcs12 -export" OpenSSL command.

## Troubleshooting and Limitations - Load Generators

This section describes troubleshooting and limitations for running Vusers on load generators machines.

### Load Generator not included with "All Generators"

**Problem:** I am adding a Vuser group to a scenario in percentage mode. When I click **All Generators** to add all the load generators to the scenario, the load generator called "localhost" is not added.

**Cause:** By default, the load generator called "localhost" is defined as a **Low Priority LG**. Any load generator defined as a **Low Priority LG** is not included when clicking the **All Generators** button.

**Solution:**

Do one of the following:

- Manually add the host that was not included
- Give "localhost" a different name, for example **127.0.0.1**

- Shut down the Controller and edit **<Installation directory>\Config\wlrun7.ini** file. Under**[General]**, add the following line:

```
HostWithLowPriority=
```

## Overloading of CPU

If you receive a warning indicating that the load generator machine has reached 80% of the CPU resources, make sure that your load generator machine conforms to the recommended system requirements as stated in the Readme file.

## Custom Cipher

For communication between the various LoadRunner components (such as Controller and Load Generator, MI Listener etc.) you may need to specify your desired cipher. To do so, add a **[SSLCipher]** section to the **config/m_agent_attribs.cfg** file. Add the cipher name attribute using this syntax: "SSLCipherList=<cipher_string>". For details, see https://www.openssl.org/docs/man1.0.2/apps/ciphers.html.

## Virtual Machines

LoadRunner supports the latest versions of VMware ESX. However, running Vusers on virtual machines may adversely affect performance due to the sharing of physical resources.

## VB and C# Projects

You cannot run VB and C# projects created with the Visual Studio add-ins on load generators with limited user permissions.

## Limited Users

You cannot run LoadRunner projects created in Visual Studio (VB and C#) on remote load generators if they were launched by a limited, non-admin user.

## ODBC scripts on Linux machines

When running ODBC scripts on RHEL 6, you may encounter the following replay error:

```
lrdb_open_connection: "SQLConnect", return-code=-1, native-error-code=0, SQLState=01000,
SQLError=[unixODBC][Driver Manager]Can't open cursor lib 'libodbccr' : file not found"
```

**Workaround:** Create a libodbccr.so.1 soft link to libodbccr.so.2.0.0 using the following syntax:

```
n -s libodbccr.so.2.0.0 libodbccr.so.1
```

For details, see **https://bugzilla.redhat.com/show_bug.cgi?id=719595**.

### GUI Vusers

Windows Load Generator machines can only run one GUI Vuser at a time. To run multiple GUI Vusers in a load test, you need to open a terminal server session for each GUI Vuser.

### Non-English Locales

Scripts containing Japanese symbols may generate errors during replay or fail during replay, when running on Linux load generators set to a Japanese locale.

# Scheduling Manual Scenarios

## Scheduling Manual Scenarios Overview

An important factor in the creation of a manual scenario is developing a test that accurately portrays user load behavior—the type of load and the timing of the load.

After you create a scenario, you schedule the scenario to start running at a specified time. You can limit the execution duration of the scenario or of a Vuser group within the scenario.

You can also stipulate how many Vusers to start and stop running within a certain time frame. You can specify whether LoadRunner should start or stop running all Vusers in a scenario simultaneously, or only a certain number of Vusers within a specified amount of time.

> **Note:** Rendezvous points in a Vuser script interfere with a scheduled scenario run. If your script contains rendezvous points, your scenario will not run as scheduled. For details on rendezvous points, see "Rendezvous Points" on page 1090.

## Scheduling by Scenario or Group

After you have designed a manual scenario, you can schedule the participating Vuser groups/scripts to run as part of a scenario schedule.

You can schedule all the groups/scripts to run together on one schedule, or you can define a separate schedule for each Vuser group.

### Scheduling By Scenario

When you schedule **by scenario**, LoadRunner runs all the Vuser groups participating in the scenario simultaneously. That is, the schedule defined for running the scenario is applied to all the Vuser groups concurrently, and LoadRunner applies each action proportionately to all the Vusers groups.

For example, take a scenario that includes 3 participating Vuser groups as follows:

| Group Name | Number of Vusers |
|---|---|

| | |
|---|---|
| Group1 | 10 |
| Group2 | 20 |
| Group3 | 30 |
| **Total** | **60** |

When scheduling by scenario, if the schedule instructs LoadRunner to load 30 Vusers when it starts running, LoadRunner loads a proportional number of Vusers from each group as follows:

| Group Name | Number of Vusers |
|---|---|
| Group1 | 5 |
| Group2 | 10 |
| Group3 | 15 |
| **Total** | **30** |

> **Note:** The same principle applies when viewing the scenario in percentage mode.

### Scheduling By Vuser Group

> **Note:** For scenarios in Vuser group mode only.

When you schedule **by Vuser group**, each Vuser group participating in the scenario runs on its own separate schedule. That is, for each Vuser group, you can specify when to start running the Vuser group, how many Vusers in the group to start and stop running within specified time intervals, and how long the group should continue running.

### See also:

- "Manual Scenarios" on page 942

## Schedule Run Modes

You can schedule a scenario to run according to the runtime settings defined in the Vuser groups, or you can let the groups run over and over again until the scenario schedule instructs them to stop running.

You can schedule a scenario to run in one of the following modes:

- **Real-world schedule.** (Default) The scenario runs according to a user-defined group of actions that simulate a real-world schedule of events. Vuser groups run according to the iterations defined in their runtime settings, but you can define how many Vusers to run at a time, how long Vusers should

continue to run, and how many Vusers to stop running at a time.

- **Basic schedule.** All enabled Vuser groups run together on one schedule, each according to its own runtime settings. You can schedule how many Vusers to start running at a time, and how long they should run before stopping.

> **Note:** You can change the default run mode in the **Tools > Options > Execution** tab.

The following table illustrates how the given schedule types run in real-world versus basic run mode:

| Schedule by | Run mode | |
|---|---|---|
| | **Real-world** | **Basic** |
| **Scenario** | All participating Vuser groups run together on one schedule. The scenario runs according to a user-defined group of actions that emulate a true-to-life schedule of events. You can schedule how many Vusers to start running at a time, how long to run the Vusers, and how many Vusers to stop running at a time. | All participating Vuser groups run together on one schedule, each according to its own runtime settings. You can schedule the Vusers to start and stop running simultaneously or gradually, and you can specify how long they should run before stopping. |
| **Group**<br>(Not applicable when viewing scenario in Percentage mode) | Each participating Vuser group runs according to its own defined schedule that emulates a true-to-life schedule of events for that Vuser group. You can schedule when to start running the Vuser group, how many Vusers to run at a time, how long to run the Vusers, and how many Vusers to stop running at a time. | Each participating Vuser group runs according to its own schedule, each according to its own runtime settings. For each Vuser group, you can schedule how many Vusers in the group to start and stop running simultaneously or gradually, and you can specify how long they should run before stopping. |

## Schedule Actions

A scenario schedule contains a series of actions that instruct the scenario when to start running a Vuser group, how to initialize Vusers, when to start and stop running Vusers, and how long to run an action.

You set these actions from the Edit Action dialog box. For details, see "Edit Action Dialog Box" on page 1043.

The following sections describe the available schedule actions.

### Start Group

The Start Group action defines when to start running a Vuser group.

| Options | Description |
|---|---|

| | |
|---|---|
| **Start immediately after the scenario begins** (Default) | LoadRunner starts running the Vuser group as soon as the scenario starts running. |
| **Start** <00:00:00> **(HH:MM:SS) after the scenario begins** | After the scenario starts running, LoadRunner waits the specified time (in hours, minutes, and seconds) before it starts running the Vuser group. |
| **Start when group <group name> finishes** | LoadRunner starts running the Vuser group immediately after the Vuser group specified in this option has finished running. |

**Note:**

- The Start Group action is available for **group schedules only**, and always appears as each group's first schedule action.
- The Start Group action is always followed by the **Initialize** action.
- The Start Group action cannot be deleted.

## Initialize

The Initialize action instructs LoadRunner to prepare the Vusers so that they are in the Ready state and can run.

| Options | Description |
|---|---|
| **Initialize all Vusers simultaneously** | LoadRunner initializes all the Vusers together before running them.<br>**Note:** Selecting this option loads all of the Vuser scripts before beginning to run them. If your scripts are very large and you expect the loading time to be lengthy, select this option. |
| **Initialize** XX **Vusers every** <00:00:00> **(HH:MM:SS)** | LoadRunner initializes the specified number of Vusers gradually, according to the specified time interval (in hours, minutes, and seconds), before running them. |
| Initialize each Vuser just before it runs (Default) | LoadRunner initializes each Vuser just before it starts running.<br>**Note:** This option is not available for group schedules when the **Wait for all groups to initialize** option is selected. For details, see "Scenario Schedule Pane" on page 1043. |

**Note:**

- The Initialize action appears in the Actions grid for all schedule types.
- The Initialize action cannot be deleted.

## Start Vusers

The Start Vusers action instructs LoadRunner to start running Vusers.

| Options | Description |
|---|---|
| **Start** XX **Vusers: Simultaneously** (Default) | LoadRunner runs the specified number of Vusers simultaneously. |
| **Start** XX **Vusers:** YY **Vusers every** <00:00:00> **(HH:MM:SS)** | LoadRunner runs the specified number of Vusers (XX) gradually. That is, LoadRunner runs YY Vusers, and waits the specified time (in hours, minutes, and seconds) before running another YY Vusers. |

**Note:**

- LoadRunner starts running Vusers only when they have reached the Ready state.
- In a basic schedule, LoadRunner always runs **all** the Vusers, whether simultaneously or gradually. In a real-world schedule, you can select how many Vusers to start running.
- While a scenario is running, you can add Vuser groups/scripts to participate in the scenario. When starting the Vusers gradually, if you add a Vuser group to the scenario after all the original Vusers have already started running, the new group starts running immediately.

## Duration

The Duration action instructs LoadRunner to continue running the scenario in the current state, for the specified amount of time.

| Options | Description |
|---|---|
| **Run until completion** | The scenario runs until all the Vusers have finished running.<br><br>**Note:** In real-world schedules, this option is available after the **first** Start Vusers action only, and if selected, causes all subsequent actions to be deleted. |
| **Run for** XX **days and** <00:00:00> **(HH:MM:SS)** | The scenario runs in its current state for the specified amount of time (in days, hours, minutes, and seconds) before continuing with the next action.<br><br>**Default:** 5 minutes.<br><br>**Note:** In a real-world schedule, if you select this option, and this Duration action is not followed by any other action, the scenario continues to run indefinitely. |

| | |
|---|---|
| **Run indefinitely** (Basic schedule only) | The scenario runs indefinitely. |

### Stop Vusers

The Stop Vusers action instructs LoadRunner to stop running Vusers.

| Options | Description |
|---|---|
| **Stop** XX **Vusers: Simultaneously** (Default) | LoadRunner stops running the specified number of Vusers at once. |
| **Stop** XX **Vuser:** YY **Vusers every** <00:00:00> **(HH:MM:SS)** | LoadRunner stops running the specified number of Vusers gradually. That is, LoadRunner stops YY Vusers, and waits the specified time (in hours, minutes, and seconds) before stopping another YY Vusers, until all XX Vusers have stopped running. |

## Define a Schedule for the Scenario

This task describes how to define a schedule for a scenario.

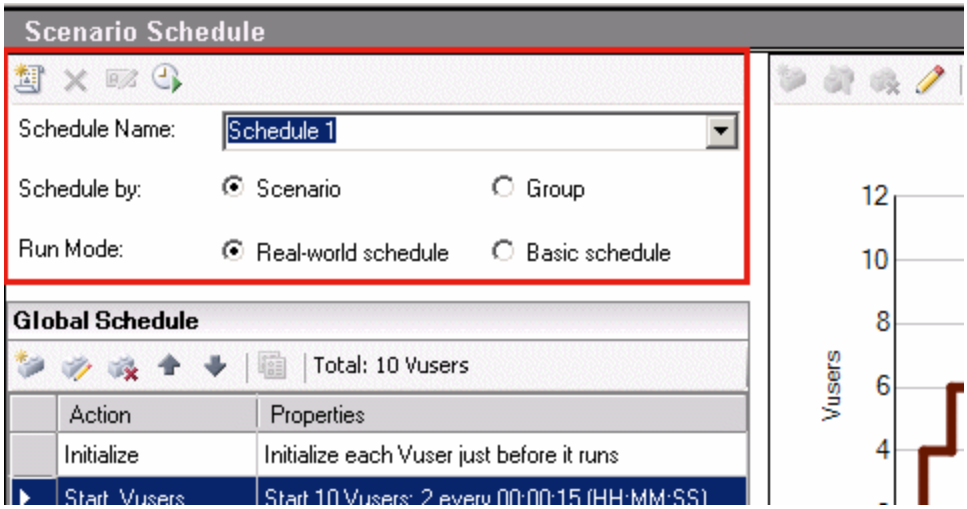
1. **Prerequisite**

   Make sure that a scenario is open, or create a new one, and that scripts have been selected for the scenario.

   For more details, see "How to Design a Manual Scenario" on page 953.

2. **Define the schedule**

   In the Scenario Schedule pane, select a schedule from the list, or define a new schedule by clicking **New Schedule**



Define the schedule in the definition area as follows:

a. (Optional) To rename the schedule, type a new name in the **Schedule Name** box and click **Save New Name** .

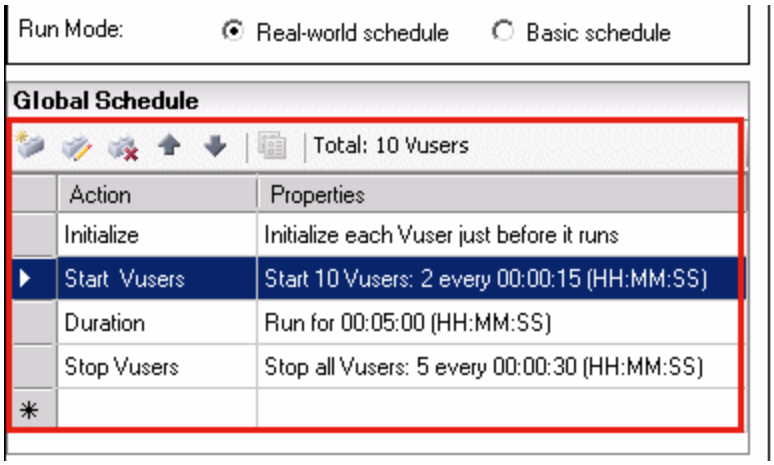
b. Select the type of schedule: **Scenario** or **Group**. For details, see "Scheduling by Scenario or Group" on page 1034.

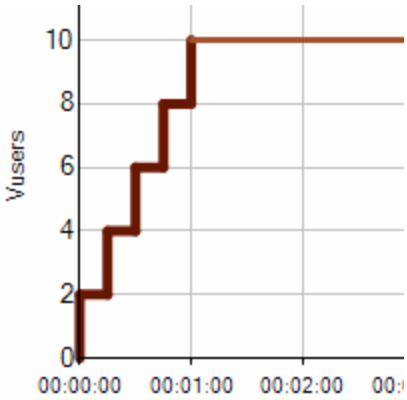
c. Select a run mode: **Real-world** or **Basic**. For details, see "Schedule Run Modes" on page 1035.

> **Note:** The default run mode for all schedules is **Real-world**. You can change the default to **Basic** in the **Tools > Options > Execution** tab.

3. **Define actions for the schedule**

The "Scenario Schedule Pane" on page 1043 displays the default actions that correspond to the type of schedule you selected above.




For schedule action details, see "Schedule Actions" on page 1036.

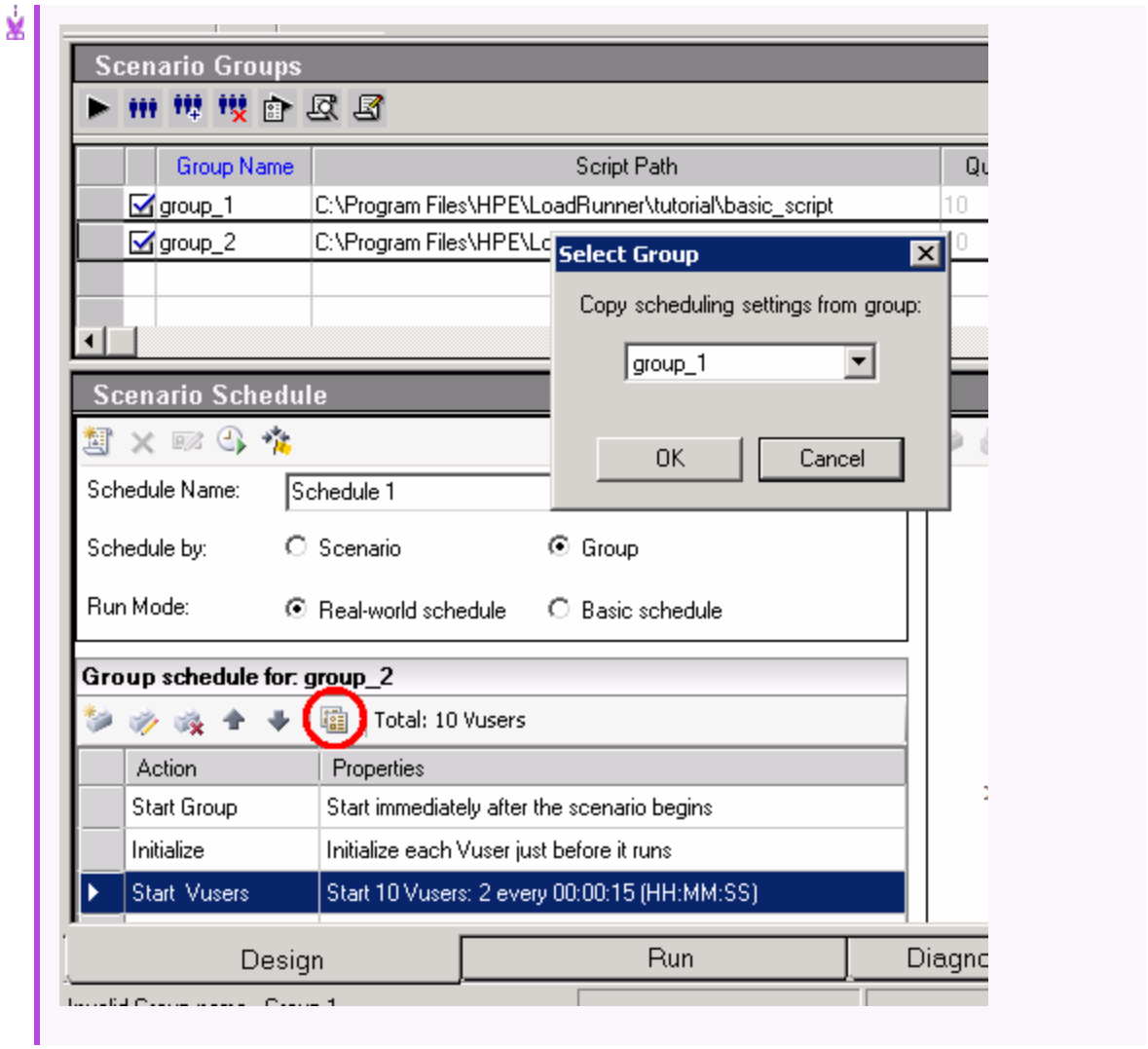
For details on adding and editing schedule actions, see "Add and Edit Schedule Actions" on the next page.

4. **Vuser group schedules only: Copy a group's schedule settings to other groups - optional**

When scheduling by Vuser group, you can copy a Vuser group's schedule settings to other Vuser groups.

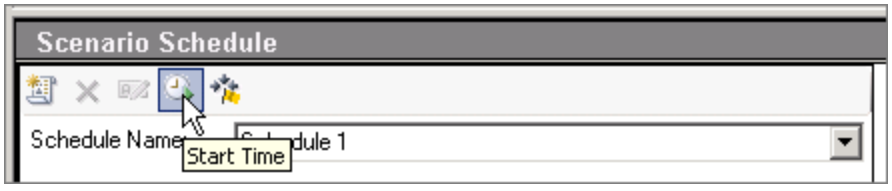
> **Note:** Schedule settings copied include the schedule run mode (basic or real-world) and the set of schedule actions.

> **Example:**
>
> To copy group_1's schedule settings to group_2, select **group_2** in the Scenario Groups pane, click **Copy Schedule Settings From** , and select **group_1**.




5. **Schedule a start time for the scenario - optional**

   In the Schedule Definition area, click the **Start Time** button and select when to start running the scenario.




## Add and Edit Schedule Actions

This task describes how to add and edit schedule actions.

## Add a schedule action

> **Note:** You can add actions to a **real-world schedule** only.

**To add an action to the schedule:**

1. In the Action grid do one of the following:
   - To insert an action after a specific action, select the action and click **Add Action After** .
   - To add an action after the last action, double-click the last row in the Actions grid—marked by an asterisk (*).
2. In the Add Action dialog box, define the new action. For schedule action details, see "Schedule Actions" on page 1036.
3. Click **Apply**.
4. To add another action while in the Add Action dialog box, click **Add Another Action** and repeat steps 2 through 3.

## Edit a schedule action

**To edit a schedule action:**

Double-click the action (or select it), click **Edit Action** , and edit the action as desired.

You can also edit other actions before closing the Edit Action dialog box. Click **Previous** or **Next** to navigate between the actions.

# Add Action Dialog Box

This dialog box enables you to add actions to a real-world scenario schedule to simulate a more true-to-life schedule by which to run your scenario.

| | |
|---|---|
| **To access** | In the **Design** tab > **Scenario Schedule** pane > **Actions** grid, do one of the following:<br>• Double-click the last row in the Actions grid—marked by an asterisk (*).<br>• In the Actions grid, select the action after which you want to add a new action, and click **Add Action After** . |
| **Important information** | • Available for **real-world schedules only**.<br>• You can add Start Vusers, Duration, and Stop Vusers actions only. |
| **Relevant tasks** | "Define a Schedule for the Scenario" on page 1039 |
| **See also** | • "Scenario Schedule Pane" on the next page<br>• "Schedule Actions" on page 1036 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| Add Another Action | Resets the Add Action dialog box so that you can add another action.<br>Available only after clicking **Apply** when adding an action. |
| Apply | Adds the defined action to the Actions grid. Leaves the Add Action dialog box open, **in edit mode**, so that you can make changes to the action you added or add another action. |
| **<Action details area>** | The details of the new action.<br>For details about the schedule actions, see "Schedule Actions" on page 1036. |
| **Action type** | The type of action to add. |

## Edit Action Dialog Box

This dialog box enables you to edit schedule actions.

| | |
|---|---|
| **To access** | Double click an action in the "Scenario Schedule Pane" below |
| **Relevant tasks** | • "Define a Schedule for the Scenario" on page 1039<br>• "Add and Edit Schedule Actions" on page 1041 |
| **See also** | • "Scenario Schedule Pane" below<br>• "Schedule Actions" on page 1036 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| < Previous<br>Next > | Displays the details of the previous/next action in the Actions grid. |
| **Action type** | Displays the type of action selected: **Initialize**, **Start Vusers**, or **Duration**. |
| **<Action details area>** | Displays the current details of the selected action. For details about each action, see Schedule Actions. |

## Scenario Schedule Pane

This pane enables you to define a schedule for running your scenario.

| | |
|---|---|
| **To access** | Manual scenario > **Design** tab |
| **Relevant tasks** | "Define a Schedule for the Scenario" on page 1039 |
| **See also** | • "Scheduling Manual Scenarios Overview" on page 1034<br>• "Scheduling by Scenario or Group" on page 1034<br>• "Schedule Run Modes" on page 1035 |

## Schedule definition area

Displays the selected schedule's details.

| UI Element | Description |
|---|---|
| | **New Schedule.** Creates a new schedule. |
| | **Delete Schedule.** Deletes the selected schedule. |
| | **Save New Name.** Saves a new name given to the schedule.<br>**Note:** Enabled when you start typing the new name. |
| | **Start Time.** Opens the Scenario Start Time dialog box. You can schedule the scenario to start running:<br>• **Without delay.** As soon as the Start Scenario command is issued.<br>• **With a delay of HH:MM:SS.** The specified time after the Start Scenario command was issued.<br>• **At HH:MM:SS on <date>.** At a specified time on the specified date. |
| <br>(Group schedule only) | **Wait for all groups to finish initialization.** When this option is selected, all of the Vusers in all of the Vuser groups finish initializing before any of them start running.<br>**Note:** When this option is selected, **Initialize each Vuser just before it runs** is not available. For details about the **Initialize** action, see "Schedule Actions" on page 1036. |
| **Run Mode** | The mode according to which the schedule will run:<br>• **Real-world schedule.** A schedule designed according to a true-to-life series of events.<br>• **Basic schedule.** A schedule according to which all the Vusers start running, run for a given duration, and then all stop running.<br>For details, see "Schedule Run Modes" on page 1035.<br>**Default value:** Real-world.<br>**Note:** To change the default, select **Tools > Options > Execution** tab. |

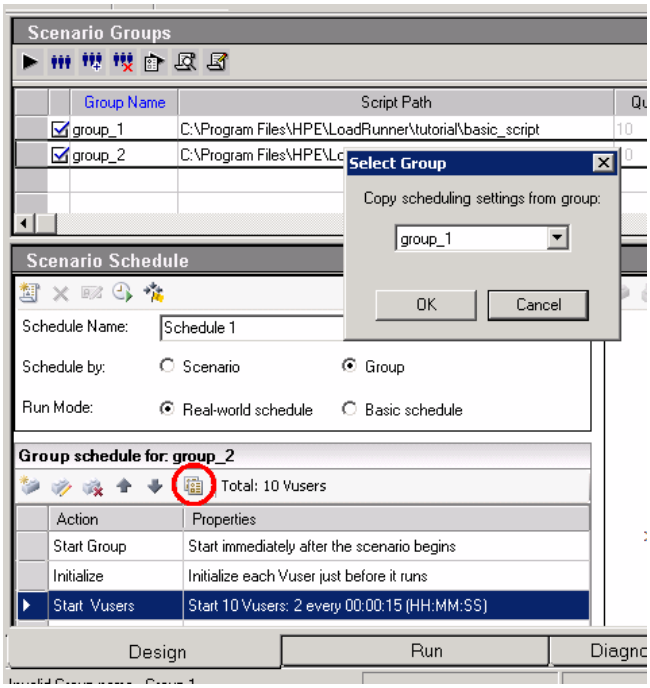
| | |
|---|---|
| **Schedule by** | The schedule type:<br>• **Scenario.** Runs all participating Vuser Groups together on the same schedule.<br>• **Group.** Each Vuser group runs on its own schedule.<br>For details, see "Scheduling by Scenario or Group" on page 1034. |
| **Schedule Name** | The name given to the schedule. |

## Action grid

You define the actions for a schedule in this area. You can add, modify, and delete actions. These actions include starting Vuser groups (in group schedules) and initializing, starting, and stopping Vusers. You can also define how long each action should continue.

When creating group schedules, you can copy group schedule settings from one Vuser group to another.

| UI Element | Description |
|---|---|
| <br>(Real-world schedule only) | **Add Action After.** Opens the Add Action dialog box where you can define new actions. For details, see "Add Action Dialog Box" on page 1042.<br>The new action is added **after** the action selected in the Actions grid. |
| | **Edit Action.** Opens the Edit Action dialog box where you can edit the schedule actions. For details, see "Edit Action Dialog Box" on page 1043. |
| | **Delete Action.** Deletes the selected action. |
| | **Move Action Up.** Moves the selected action up the grid. |
| | **Move Action Down.** Moves the selected action down the grid. |

| | |
|---|---|
| <br>(Vuser group schedules only) | **Copy Schedule Settings From.** Enables copying group schedule settings from one Vuser group to another in the Scenario Groups pane.<br>**Note:** Schedule settings copied include the schedule run mode (basic or real-world) and the set of schedule actions.<br>**Example:** To copy group_1's schedule settings to group_2, select **group_2** in the Scenario Groups pane. Then click this button, and select **group_1**.<br>  |
| Total: 20 Vusers | Displays the total number of Vusers scheduled to run in the scenario.<br>• This is editable in basic schedules, when the scenario is in percentage mode only.<br>• When the scenario is in Vuser group mode and you are defining a basic schedule, this value is updated when you modify the quantity of Vusers in the Vuser groups.<br>For details, see "Add Vusers Dialog Box" on page 961.<br>• In real-world schedules, the total number of Vusers is the sum of all the Vusers defined in the Start Vusers actions. |

| | |
|---|---|
| | **Apply.** When you modify the **Total: <#> Vusers** field, applies the specified total number of Vusers proportionately to the Vuser groups. (Percentages are visible in the % column of the Scenario Scripts pane.)<br><br>This appears only when **Total: <#> Vusers** is modified. |

## Schedule graph

Displays a graphical representation of the scenario schedule. The lines in the graph correspond to the actions defined in the Actions grid.

The graph reflects the scenario Vuser schedule and not the actual state of the Vusers during scenario execution.

| UI Element | Description |
|---|---|
| | **Pause.** Pauses the schedule during the scenario run. When the schedule is paused, the red vertical line that indicates the schedule's progress freezes.<br><br>**Note:** Available only when the scenario is running. |
| | **Resume.** Resumes a paused schedule. When the schedule resumes running, the red vertical line continues moving across the graph, indicating the schedule's progress.<br><br>**Note:** Available only when the scenario is running. |
| | **Show Selected Group.** Displays only the group selected in the Scenario Groups pane.<br><br>**Note:** Available for group schedules only. |
| | **Show All Groups.** Displays all Vuser groups participating in the scenario.<br><br>**Note:** Available for group schedules only. |
| | **Open Full View.** Opens the graph in its own window. |
| | **Zoom In.** Zooms into the x-axis of the graph, spreading the graph out to view shorter time intervals. |
| | **Zoom Out.** Zoom out of the x-axis of the graph, displaying longer time intervals. |
| | **Zoom Reset.** Reverts to the default time intervals displayed on the x-axis. |

| | |
|---|---|
| | **Hide/Show Legend.** Hides/Shows the graph legend. |
| **<Schedule progress indicator>** | A red vertical line that slides across the schedule graph during the scenario run.<br>**Note:** The schedule may run a few seconds ahead of the scenario run. |
| **<Time scroll bar>** | Appears if, when zooming into the graph, the graph spreads out wider than the graph area. |

# Service Level Agreements

Service level agreements (SLAs) are specific goals that you define for your load test scenario. After a scenario run, LoadRunner Analysis compares these goals against performance related data that was gathered and stored during the course of the run, and determines whether the SLA passed or failed.

To learn more about Service Level Agreements (SLAs), see "Service Level Agreements Overview" on page 1300.

For details about defining an SLA, see "Define Service Level Agreements" on page 1301.

# Multiple IP Addresses

## Multiple IP Addresses Overview

Application servers and network devices use IP addresses to identify clients. The application server often caches information about clients coming from the same machine. Network routers try to cache source and destination information to optimize throughput. If many users have the same IP address, both the server and the routers try to optimize. Since Vusers on the same load generator have the same IP address, server and router optimizations do not reflect real-life situations.

LoadRunner's Multiple IP Address feature enables Vusers running on a single load generator to be identified by many IP addresses (this is known as **IP spoofing**). The server and router recognize the Vusers as coming from different load generators and as a result, the testing environment is more realistic.

This feature can be implemented on Windows and Linux platforms with the following protocols: DNS, IMAP, Oracle NCA, Oracle-Web, POP3, RTE, SAP-Web, Siebel-Web, SMTP, Web - HTTP/HTML, Web Services, and Windows Sockets.

## How to Add IP Addresses to a Load Generator

The following steps describe how to add IP addresses to a load generator.

1. Run the IP Wizard on the load generator
   - **Windows:** LoadRunner includes an IP Wizard program that you run on Windows load

generators to create multiple IP addresses. You add new IP addresses to a machine once and use the addresses for all scenarios.

Run the IP Wizard on the load generator to add a specified number of IP addresses.

For details, see "IP Wizard" below.

- **Linux:** Manually configure the new IP addresses for Linux load generators.

2. Update the server's routing table with the new addresses

Once the client machine has new IP addresses, the server needs the addresses in its routing table, so that it can recognize the route back to the client. If the server and client share the same netmask, IP class, and network, the server's routing table does not require modification.

> **Note:** If there is a router between the client and server machines, the server needs to recognize the path via the router. Make sure to add the following to the server routing table: route from the Web server to the router, and routes from the router to all of the IP addresses on the load generator.

Update the Web server routing table as follows:

a. Edit the batch file that appears in the IP Wizard Summary screen. An example **.bat** file is shown below.

```
REM This is a bat file to add IP addresses to the routing table of a server
REM Replace [GATEWAY_IP] with the IP of the gateway that can route packets
REM to the Load Generator network
REM This script should be executed on the server machine

route ADD 192.168.1.1 MASK 255.255.255.255 [GATEWAY_IP] METRIC 1
route ADD 192.168.1.2 MASK 255.255.255.255 [GATEWAY_IP] METRIC 1
route ADD 192.168.1.3 MASK 255.255.255.255 [GATEWAY_IP] METRIC 1
route ADD 192.168.1.4 MASK 255.255.255.255 [GATEWAY_IP] METRIC 1
```

b. For each occurrence of [GATEWAY_IP], insert your IP address instead.

c. Run the batch file on the server machine.

3. Enable the Multiple IP Addresses feature from the Controller

Once you define multiple IP addresses, you set an option to tell the Controller to use this feature.

a. In the Controller Design view, select **Scenario > Enable IP Spoofer**.

> **Note:** You must select this option before connecting to a load generator.

b. In the Controller's **Tools > Options > General** tab (Expert mode only), specify how the Controller should allocate the IP addresses: per process or per thread. For details, see "Options > General Tab" on page 1077.

## IP Wizard

This wizard enables you to create and save new IP addresses on Windows machines.

| | |
|---|---|
| **To access** | On the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > IP Wizard** |
| **Important information** | This wizard resides on each load generator. It enables you to create and save new IP addresses on Windows machines. The new addresses can be a range of addresses defined by the Internet Assignment Numbers Authority. They are for internal use only, and cannot connect to the Internet. This range of addresses is the default used by the IP Wizard. |
| **Relevant tasks** | "How to Add IP Addresses to a Load Generator" on page 1048 |

User interface elements are described below:

| | |
|---|---|
| **Step 1 of 3 - Welcome** | |
| **Create new settings** | Enables you to define new IP settings on the load generator. |
| **Load previous settings from file** | Enables you to use an existing file with IP address settings. |
| **Restore Original Settings** | Restores original settings. |
| **Step 2 of 3 - Optional** | |
| **<Web Server Address box>** | If you have more than one network card, enables you to choose the card to use for IP addresses.<br>This step enables the IP Wizard to check the server's routing table to see if it requires updating after new IP addresses are added to the load generator. |
| **Step 3 of 3 - Optional** | |

| | |
|---|---|
| **Add button** | Opens the Add dialog box where you can add a new IP address.<br>The Add dialog box contains the following options:<br>• **Private Address Spaces.** Classes that represents the correct submask for the machine's IP addresses.<br>• **From IP.** Adds IP addresses starting with this number.<br>• **Submask.** IP addresses include two components, a **netid** and **hostid**. The submask determines where the netid portion of the address stops and where the hostid begins.<br>• **Number to add.** The number of IP addresses to add.<br>• **Verify that new IP addresses are not already in use.** Instructs the IP Wizard to check the new addresses. The IP Wizard adds only the addresses that are not already in use. |
| **Remove button** | Removes a selected IP Address. |
| **IP Address** | The IP addresses on the load generator machine. |
| **Subnet Mask** | The submasks of the IP addresses on the load generator machine. |
| **Number of IPs added** | The number of IP addresses added to the load generator machine. |
| **Finish button** | Click **Finish** to exit the IP Wizard. The IP Wizard will run a **netsh** batch file to update the NT device drivers with the new addresses. |
| **Summary** | |
| **<Summary area>** | Displays a summary of the operations performed by the IP Wizard.<br>Take note of the location of the **netsh** batch file (**.bat**). This is used to update the routing table, if necessary. See "How to Add IP Addresses to a Load Generator" on page 1048. |

# LoadRunner Terminal Services Manager

**Note:** This feature is not supported if Controller and the load generators connect over a firewall.

## Terminal Services Overview

Terminal services allows centralized management of computing resources for each client connected to the server, and provides each user with their own working environment. Using a terminal server client, you can operate in a server-based computing environment from a remote machine. The terminal server transmits applications over the network and displays them via terminal emulation software. Each user

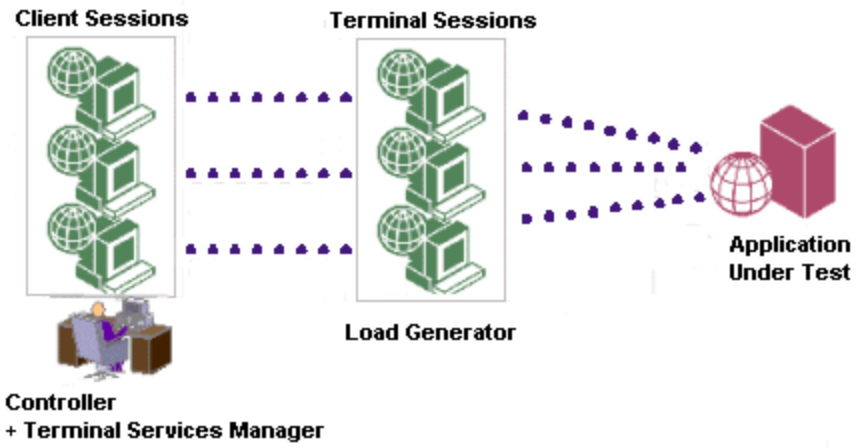
logs on and sees only his individual session, which is managed transparently by the server's operating system independent of any other client session.

You can use LoadRunner's Terminal Services Manager to remotely manage multiple load generators running in your load testing scenario on a terminal server. In addition, you can use a terminal server to overcome the limitation of being able to run only a single GUI Vuser on a Windows-based load generator. By opening a terminal server session for each GUI Vuser, you can run multiple GUI Vusers on the same application.

A terminal server client can have multiple terminal sessions running simultaneously. Using LoadRunner's Terminal Services Manager, you can select the number of terminals to be used in your scenario (provided that you have sufficient terminal sessions running), and the maximum number of Vusers that can be run per terminal. The Terminal Services Manager then evenly distributes the number of Vusers among the client sessions.

**Note:** Only users with administrator privileges can connect from Controller to a local load generator via a terminal services session.

The following diagram illustrates how the LoadRunner components work together during a terminal session.




## Using the Terminal Services Manager

This task describes how to set up and use LoadRunner's Terminal Services Manager.

1. Prerequisite

   Make sure that a load generator has been installed on the terminal services machine. For more information, see the *LoadRunner Installation Guide*.

2. Allow terminal services on the load generator

   Add *TSLauncher.exe* to the list of allowed remote applications on all load generator machines.

3. Set up the LoadRunner agent on the load generator

   Perform the following steps:

a. On the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Advanced Settings > Agent Configuration**, or run **<LoadRunner root>\launch_service\bin\AgentConfig.exe** to open the Agent Configuration dialog box.

b. Select **Enable Terminal Services**, and click **OK**.

c. Restart the LoadRunner Agent as a process by double-clicking the shortcut on the desktop, or from the Windows Start menu, go to **Micro Focus > LoadRunner > Advanced Settings > Agent Process**. You need to run the LoadRunner Agent as a process for each terminal session that you are running.

4. Launch a terminal client session

   Be aware of the following:

   - You must open a Terminal Client session for each terminal that you want to run Vusers on during the scenario.
   - If you are connecting to an existing Terminal Services session, you need to open a Terminal Client session, log in to the session, and run the LoadRunner Agent as a process.

5. Distribute Vusers on the terminal server

   In the Terminal Services tab of the Load Generator Information dialog box, select **Enable Terminal Services Manager** and enter information about the terminals and Vusers that you want to use. For more information, see "Load Generator Configuration > Terminal Services Tab" on page 1016.

# Running Load Test Scenarios

This section describes how to plan and create LoadRunner Controller scenarios.

### What do you want to do?

- Set up an online monitor
- Configure scenario options
- Prepare for a scenario run
- Run a scenario in Controller
- Run a scenario from the command line

### See also:

- Rendezvous points
- Using UFT tests
- Integration with ALM
- Continuous integration with Jenkins

# Online Monitor Graphs

# Online Monitor Graphs Overview

You can view the data collected by the online monitors using the online monitor graphs.

## About Online Monitor Graphs

Online monitor graphs display performance measurements for those resources being monitored during scenario run. Each measurement is represented on the graph by a colored line. Information about the measurements is listed in the legend below the graph. The legend displays the measurements for the selected graph only.

For details about selecting monitor graphs and customizing the graph display area, see "How to Display Online Monitor Graphs" below.

For details about customizing graph layout and measurements, see "How to Customize Online Graph and Measurement Settings" on the next page.

## Viewing Monitor Data Offline

After monitoring resources during a scenario run, you can view a graph of the data that was gathered using LoadRunner Analysis. Analysis processes the data from the scenario run results files, and generates a graph for each measurement that was monitored.

For details about working with Analysis at the end of the scenario run, see the *LoadRunner Analysis User Guide*.

# How to Display Online Monitor Graphs

This task describes how to open other monitor graphs and customize the graph display area.

1. **Prerequisites.** To see data in the online monitor graphs, the relevant monitoring environments must be configured. For details, see "How to Set Up a Monitoring Environment" on page 1171.
2. Open a monitor graph

   By default, LoadRunner displays the following graphs in the graph display area:

   a. **Running Vusers**

   b. **Transaction Response Time**

   c. **Hits per Second**

   d. **Windows Resources**

   You can open other graphs one at a time, as follows:

   - **Method 1**

     i. Select **Monitors > Online Graphs > Open a New Graph**, or right-click a graph and select **Open a New Graph**.

     ii. In the Open a New Graph dialog box, click the "**+**" in the left pane to expand the category nodes, and select a graph. You can view a description of the selected graph in the **Graph**

**Description** box.

iii. Click **Open Graph**, or drag the selected graph into the right pane of the Run view.

- **Method 2**

  In the graph tree on the left of the Run tab, click the "**+**" to expand the category nodes. Double-click a graph or, alternatively, select it and drag it to the graph display area on the right.

> **Note:** If the graph tree is not displayed, select **View > Show Available Graphs**. To hide the graph tree view, select **View > Hide Available Graphs**.

3. Customize the graph display area. Optional .

   By default, LoadRunner displays four graphs in the graph display area.

   To change the number of graphs displayed, right-click a graph in the graph display area and select **View Graphs** (or select **View > View Graphs**).

   Do one of the following:

   - Select the number of graphs to display from the options given
   - Select **Custom Number** and enter a number of graphs to display.
   - To display only one graph, double-click the graph displayed in the graph display area. To return to the previous view, double-click the graph again.

## How to Customize Online Graph and Measurement Settings

This task describes ways to customize online graphs and graph measurements.

### Configure the graph settings

You can customize:

- The type of graph display
- What to display on the x-axis and y-axis.

You can apply these settings to all graphs, or to a specific graph only.

To configure the graph settings, in the Run tab, select **Monitors > Online Graphs > Configure**, (or right-click a graph and select **Configure**).

For user interface details, see "Graph Configuration Dialog Box" on page 1058.

### Configure the measurement settings

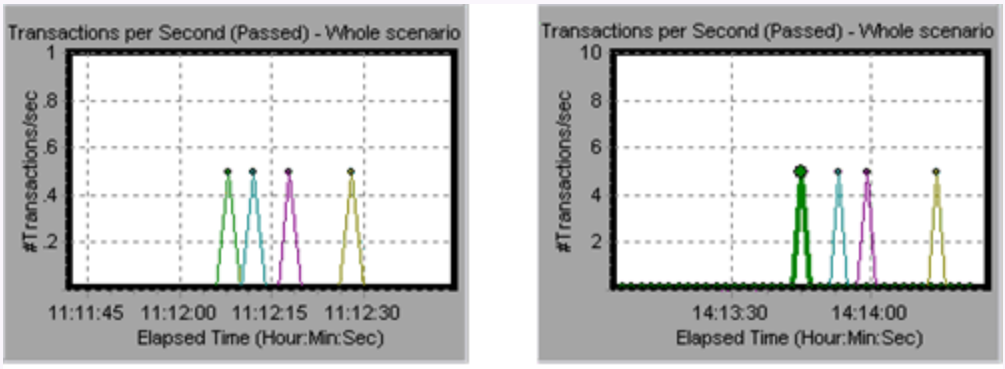
You can customize the appearance and scale of measurements in a graph, and select whether they should be displayed in the graph or hidden from the graph.

In the Run tab, right-click a measurement in the graph or legend, and select **Configure**.

For user interface details, see "Measurement Configuration Dialog Box" on page 1060.

**Example: Measurement Scale**

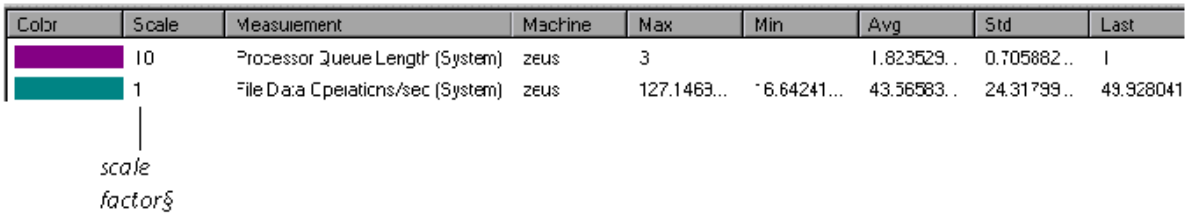
In the following example, the same graph is displayed with a scale of 1 and 10:




*Scale =1* *Scale = 10*

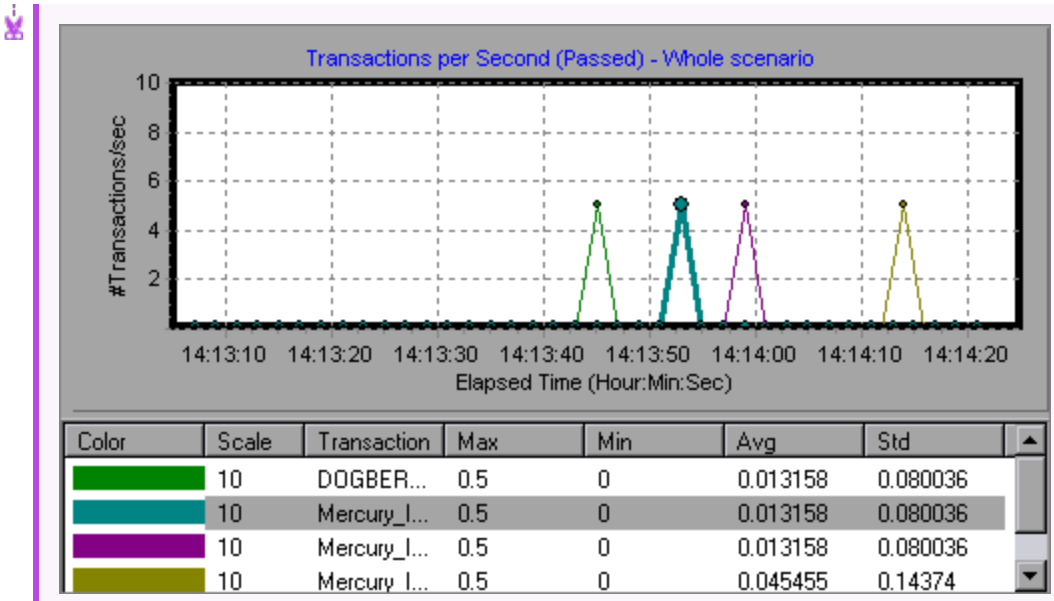
The actual graph values range from 0-1, as shown in the left graph. You can view the information more accurately using a larger scale for the display, as shown in the right graph. However, to obtain the actual values, you need to divide the displayed value by the scale. In the example above, the highest value shown in the graph is 5. Since the scale is 10, the actual value is 0.5.

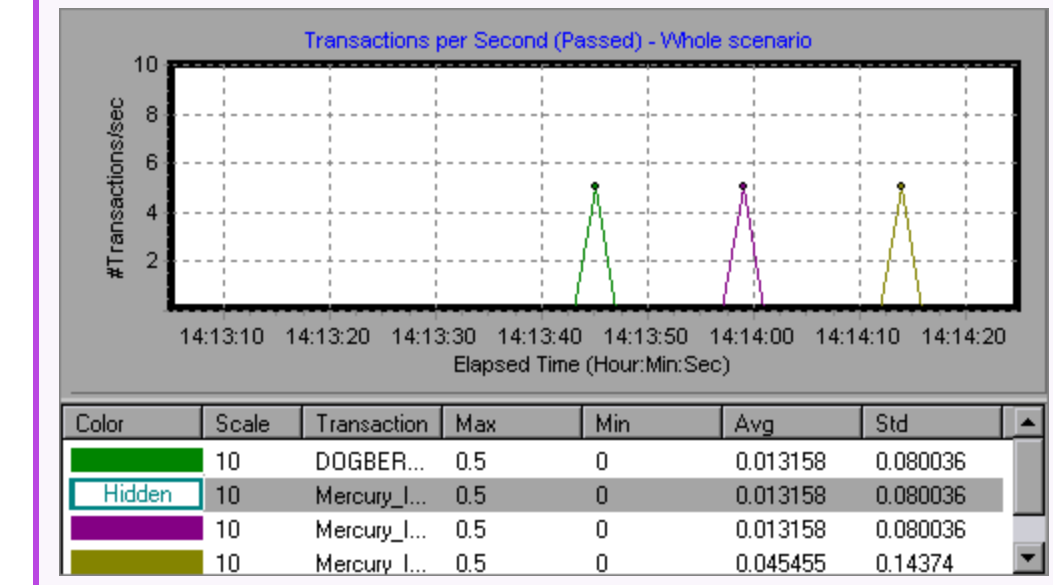
The legend below the graph indicates the scale factor.




**Example: Shown/Hidden Measurements**

In the following example, the first image displays a line for each of the four measurements. In the second image, the second measurement listed in the legend is hidden in the graph:




## Manage Online Graphs

The following sections describe ways to work with the online monitor graphs.

### Freeze graphs

You can pause a specific graph during a scenario run. Select the graph and select **Monitors > OnlineGraph** > **Freeze,** or right-click the graph and select **Freeze**. To resume, repeat the above action. When resumed, the graph displays the data for the paused period as well.

### Overlay graphs

You can merge or overlay the results of two graphs from the same scenario into a single graph. This enables you to compare several different measurements at once.

In the Run tab, right-click one of the online graphs you want to overlay, and select Overlay Graphs.

> **Note:** The x-axis of both graphs must be the same measurement.

For details, see "Overlay Graphs Dialog Box" on page 1062.

### Export graphs to HTML

You can export graphs displayed on the Run tab to HTML format for offline viewing at a later stage. When you export to HTML, the legend is also displayed with the graph.

You can export a single graph or all graphs in the online monitor display.

### Export a single graph

1. Right-click the graph and select **Export to HTML.**
2. Specify a path and filename for the exported graph/report.

## Graph Configuration Dialog Box

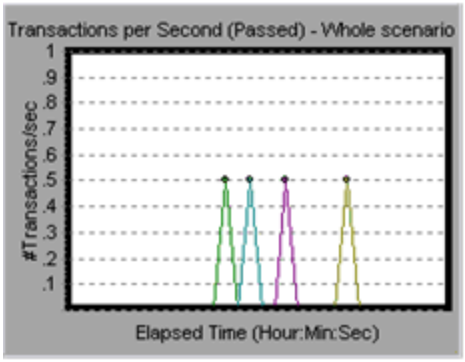
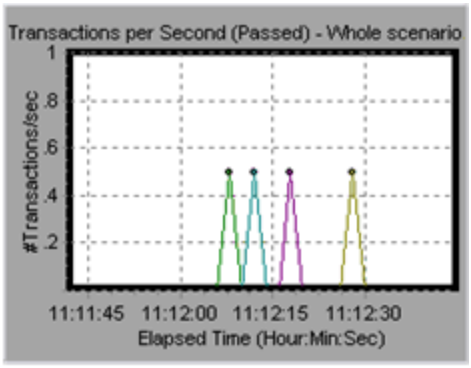
This dialog box enables you to customize the online graph settings.

| | |
|---|---|
| **To access** | Use one of the following:<br>• Select **Monitors > Online Graphs > Configure**.<br>• Right-click a graph and select **Configure**. |
| **Important information** | You can apply these settings to all graphs, or to a specific graph only. |
| **Relevant tasks** | "How to Customize Online Graph and Measurement Settings" on page 1055 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Apply to all graphs** | Applies the dialog box settings to all graphs. |

| | |
|---|---|
| **Apply to selected graph** | Applies the dialog box settings to the selected graph. |
| **Bar Values Type** | If the **Bar** display type is selected, determines the type of value that will be displayed in the bar graph: **Average**, **Last Value**, **Minimum**, **Maximum**. |
| **Display Type** | The type of graph displayed: line graph or bar graph.<br>By default, each graph is displayed as a line graph.<br><br>**Note:** For the Network Delay graph, if you right-click the graph and select **View Segments**, you can view the network segments of the graph as an area graph or a pie graph. |
| **Graph Time** | Indicates the scale for a graph's x-axis when it is time-based. A graph can show 60 to 3600 seconds of activity. To see the graph in greater detail, decrease the graph time. To view the performance over a longer period of time, increase the graph time. The available graph times are: **Whole Scenario**, **60**, **180**, **600**, and **3600** seconds. |
| **Network Delay View** | Available for the Network Delay Time graph only:<br>• **SubPaths**. Displays the delay measurements from the source machine to each of the nodes along the network path.<br>• **DNS name.** Displays the DNS names of the measurements displayed in the legend. |
| **Refresh Rate** | The interval at which the graph is refreshed with new data. By default, the graph is refreshed every five seconds. If you increase the refresh rate, the data is refreshed less frequently.<br><br>**Note:** In a large load test, it is recommended to use a refresh rate of three to five seconds. This enables you to avoid problems with CPU resource usage. |

| | |
|---|---|
| **Time** | Specifies how the graph displays the time (in seconds) on the x-axis:<br>• **Don't Show.** Instructs LoadRunner not to display values for the x-axis.<br>• **Clock Time.** Displays the absolute time, based on the system clock.<br>• **Relative to Scenario Start.** Displays the time relative to the beginning of the scenario.<br>**Note:** If no step is running, clock time is displayed.<br>**Example:** In the left image below the time is not displayed on the x-axis. In the right image, the time is displayed.<br>  <br>*Don't Show*<br>  <br>*Clock Time* |
| **Y-Axis Scale** | Displays graphs using the selected y-axis scale:<br>• **Automatic.** Displays the default y-axis values.<br>• **Maximum Y-Axis Value.** The maximum value for the y-axis.<br>• **Minimum Y-Axis Value.** The minimum value for the y-axis. |

## Measurement Configuration Dialog Box

This dialog box enables you to configure settings for measurements in a graph. You can:

- Change line colors
- Configure a measurement's scaling
- Show/hide measurements
- View descriptions of the measurements

| | |
|---|---|
| **To access** | In the Run tab, right-click a measurement in the graph or legend, and select **Configure.** |
| **Relevant tasks** | "How to Customize Online Graph and Measurement Settings" on page 1055 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Configuration tab** | • **Color.** The color assigned to the selected measurement.<br>• **Scale.** The relationship between the y-axis and the graph's actual value.<br>  • **Autoscale.** Automatically scales the measurement by calculating the best ratio for displaying the graph. For some graphs, this option is not available.<br>  **Default value:** Autoscale<br>  **Example:** A scale of **1** indicates that the measurement's value is the value of the y-axis. If you select a scale of **10**, you must multiply the y-axis value by 10 to obtain the true value of the measurement.<br>• **Show / Hide.** The resource selected in the legend is shown/hidden in the graph.<br>By default, all resource measurements are shown in the graph. To show only a selected measurement, right-click the measurement, and select **Show Only Selected**.<br>**Note:** Alternatively right-click a measurement in the graph legend and select **Show/Hide**. |
| **Description tab** | Information about the measurement:<br>• **Machine.** Displays the name of the machine whose resources are being monitored.<br>**Note:** Displayed only when a machine's resources are being monitored.<br>• **Description.** Displays a description of the selected measurement.<br>**Note:** Also accessible by right-clicking a measurement in the legend and selecting **Description**. |
| **Machine** | The name of the machine whose resources are being monitored.<br>**Note:** Displayed only when a machine's resources are being monitored. |
| **Measurement** | The name of the selected measurement. |
| **Network Type** | Appears only when monitoring a network path. |

## Open a New Graph Dialog Box

This dialog box enables you to open a new graph.

| | |
|---|---|
| **To access** | Use one of the following:<br>• **Run** tab > **Monitors** > **Online Graphs** > **Open a New Graph**<br>• Right-click a graph and select **Open a New Graph**. |
| **Important information** | The graph selected in the graph display area will be replaced by the added graph. |
| **Relevant tasks** | "How to Display Online Monitor Graphs" on page 1054 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Open Graph | Opens the selected graph and displays it in the graph tree view. |
| **Display only graphs containing data** | Select this option to view only those graphs that contain data. To view the entire list of online monitor graphs (even those that do not contain data), clear this option. |
| **Graph Description** | Displays a description of the selected graph |
| **Select Graph box** | Lists the online monitor graphs by category. To expand a category node, click the "+".<br>**Tip:** Graph names displayed in blue contain data.<br>**Note:** You can select only one graph at a time. |

## Overlay Graphs Dialog Box

This dialog box enables you to merge or overlay the results of two graphs from the same scenario into a single graph. The merging enables you to compare several different measurements at once.

For example, you can make an overlaid graph that displays the Web Throughput and Hits per Second as a function of the elapsed time.

| | |
|---|---|
| **To access** | In the Run tab, right-click one of the online graphs you want to overlay, and select **Overlay Graphs**. |
| **Important information** | • In order to overlay graphs, the x-axis of both graphs must be the same measurement.<br>• When you overlay the contents of two graphs that share a common x-axis, the left y-axis on the overlaid graph shows the current graph's values. The right y-axis shows the values of the graph that was overlaid. |

| | |
|---|---|
| **Relevant tasks** | "Manage Online Graphs" on page 1057 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Current Graph** | The name of the current graph. |
| **Select graph to overlay with** | The name of the graph to be merged with the current graph.<br><br>**Note:** The drop-down list displays only the active graphs that have a common x-axis with the current graph. |
| **Title of overlaid graph** | The title given to the overlaid graph. |

## Available Graphs Tree

The Available Graphs Tree displays the online monitor graphs.

**Tip:** Graph names that are displayed in blue in the list of **Available Graphs**, contain data.

To select measurements to monitor in a particular graph, see the monitor configuration instructions for each specific monitor. For details, see "How to Set Up a Monitoring Environment" on page 1171.

| Graph | Description |
|---|---|
| **Running Vusers** | Provides information about the status of the Vusers running in the current scenario on all load generators. The graph shows the number of running Vusers, while the information in the legend indicates the number of Vusers in each state. |
| **User-Defined Data Points** | Displays the real-time values of user-defined data points. You define a data point in your Vuser script by inserting an **lr_user_data_point** function at the appropriate place (**user_data_point** for GUI Vusers and **lr.user_data_point** for Java Vusers). |
| **Error Statistics** | Provides details about the number of errors that occurred during each second of the scenario run. The errors are grouped by error source—for example, the location in the script or the load generator name. |
| **Vusers with Errors** | Provides details about the number of Vusers that generate errors during scenario execution. The errors are grouped by error source. |
| **Transaction Response Time** | Shows the average response time of transactions in seconds (y-axis) as a function of the elapsed time in the scenario (x-axis). |

| Graph | Description |
|---|---|
| **Transaction Response Time by Location** | Shows the average transaction response time (y-axis) for each of the virtual locations with which the transactions were performed (x-axis). |
| **Transaction Per Second (Passed)** | Shows the number of successful transactions performed per second (y-axis) as a function of the elapsed scenario time (x-axis). |
| **Transaction Per Second (Passed) by Location** | Shows the number of successful transactions performed per second (y-axis) as a function of the elapsed time in the scenario (x-axis), broken down by location. |
| **Transaction Per Second (Failed, Stopped)** | Shows the number of failed and stopped transactions per second (y-axis) as a function of the elapsed scenario time (x-axis). |
| **Transaction Per Second (Failed, Stopped) by Location** | Shows the number of failed and stopped transactions per second (y-axis) as a function of the elapsed scenario time (x-axis), broken down by location. |
| **Total Transactions Per Second (Passed)** | Shows the total number of completed, successful transactions per second (y-axis) as a function of the elapsed scenario time (x-axis). |
| **Hits Per Second** | Shows the number of hits (HTTP requests) to the Web server (y-axis) as a function of the elapsed time in the scenario (x-axis). You can compare this graph to the Transaction Response Time graph to see how the number of hits affects transaction performance. |
| **Throughput** | Shows the amount of throughput on the Web server (y-axis) during each second of the scenario run (x-axis). Throughput is measured in bytes and represents the amount of data that the Vusers received from the server at any given second. You can compare this graph to the Transaction Response Time graph to see how the throughput affects transaction performance. |
| **HTTP Responses per Second** | Shows the number of HTTP status codes—which indicate the status of HTTP requests, for example, "the request was successful," "the page was not found"—(y-axis) returned from the Web server during each second of the scenario run (x-axis), grouped by status code. You can group the results shown in this graph by script (using the "Group By" function) to locate scripts which generated error codes. |

| Graph | Description |
|---|---|
| **WebSocket Statistics** | Provides statistics for the WebSocket data during the scenario run, such as byte rate, connection status, and the number of messages. |
| **Pages Downloaded per Second** | Shows the number of Web pages (y-axis) downloaded from the server during each second of the scenario run (x-axis). This graph helps you evaluate the amount of load Vusers generate, in terms of the number of pages downloaded. |
| **Retries per Second** | Shows the number of attempted Web server connections (y-axis) as a function of the elapsed time in the scenario x-axis). A server connection is retried when the initial connection was unauthorized, when proxy authentication is required, when the initial connection was closed by the server, when the initial connection to the server could not be made, or when the server was initially unable to resolve the load generator's IP address. |
| **Connections** | Shows the number of open TCP/IP connections (y-axis) at each point in time of the scenario (x-axis). One HTML page may cause the browser to open several connections, when links on the page go to different Web addresses. Two connections are opened for each Web server. |
| **Connections per Second** | Shows the number of new TCP/IP connections (y-axis) opened and the number of connections that are shut down each second of the scenario (x-axis). |
| **SSLs per Second** | Shows the number of new and reused SSL Connections (y-axis) opened in each second of the scenario (x-axis). An SSL connection is opened by the browser after a TCP/IP connection has been opened to a secure server. |
| **Windows Resources** | Shows the NT and Windows 2000 resources measured during the scenario. The NT and Windows 2000 measurements correspond to the built-in counters available from the Windows Performance Monitor. |
| **UNIX Resources** | Shows the UNIX resources measured during the scenario. The UNIX measurements include: average load, collision rate, context switch rate, CPU utilization, incoming packets error rate, incoming packets rate, interrupt rate, outgoing packets error rate, outgoing packets rate, page-in rate, page-out rate, paging rate, swap-in rate, swap-out rate, system mode CPU utilization, and user mode CPU utilization. |
| **SNMP** | Shows statistics for machines running an SNMP agent, using the Simple Network Management Protocol (SNMP). The x-axis represents the elapsed time. The y-axis represents the resource usage. |
| **Network Delay Time** | Shows the delays for the complete path between the source and destination machines (for example, the database server and Vuser load generator). The graph maps the delay as a function of the elapsed scenario time. |

| Graph | Description |
|---|---|
| **Apache** | Displays statistics about the resource usage on the Apache server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage. |
| **Microsoft IIS** | Shows server statistics as a function of the elapsed scenario time. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage. |
| **Microsoft Active Server Pages** | Displays statistics about the resource usage on the ASP server during the scenario run. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage. |
| **Oracle** | Displays information from Oracle V$ tables: Session statistics, V$SESSTAT, system statistics, V$SYSSTAT, and other table counters defined by the user in the custom query. |
| **SQL Server** | Shows the standard Windows resources on the SQL server machine. The x-axis represents the time that has elapsed since the start of the scenario run. The y-axis represents the resource usage. |
| **JMeter Graphs** | Provides information on throughput, hits, active threads, and more for JMeter tests. For details, see "JMeter Monitoring Graphs" on page 1191. |
| **MQTT Statistics Graphs** | Provide statistics about MQTT message flow and throughput between the MQTT broker and the client during the scenario run. For details, see "MQTT Statistics Monitoring" on page 1198. |
| **Streaming State Statistics Graphs** | Provide statistics about the streaming videos in the scenario run. For details, see "Streaming State Statistics Monitoring" on page 1221. |
| **TruClient Native Mobile** | Provide statistics about CPU utilization memory on a mobile device during a scenario run. For details, see "TruClient - Native Mobile Monitors" on page 1227. |
| **Network Virtualization Graphs** | Graphs that show information about the Network Virtualization: Average Latency, Packet Loss, Average Throughput, Average Bandwidth Utilization, and Total Throughput. For details, see "Network Virtualization Monitoring" on page 1209. |
| **Service Virtualization** | Shows statistics of the service virtualization, showing the **Operations** and **Services** graphs. For details, see "Service Virtualization Integration" on page 1247. |

| Graph | Description |
|---|---|
| **SiteScope** | Displays statistics about the resource usage on the SiteScope machine during the scenario run. The x-axis represents the elapsed time. The y-axis represents the resource usage.<br><br>These monitors can also be used to monitor Siebel and other ERP/CRM systems. For more information, see SiteScope Technical Specifications. |
| **Dynatrace** | Provides information on the availability and performance of software applications and the impact on user experience measured by Dynatrace. For details, see "Dynatrace Monitor" on page 1189. |
| **Flex** | Measures statistics related to Flex RTMP connections and throughput, as well as buffering time. For details, see "Flex Monitoring Graphs" on page 1190. |
| **Citrix Server** | Displays statistics about resource usage on the Citrix server during the scenario run. |
| **IBM WebSphere MQ** | Shows the resource usage of IBM WebSphere MQ Server channel and queue performance counters as a function of the elapsed scenario time. |
| **Network Client** | Shows statistics for FTP, POP3, SMTP, IMAP, and DNS Vusers on the network client machine as a function of the elapsed scenario time. |

# Configuring Scenario Options

## Configuring Scenario Options Overview

Before you run a scenario, you can configure both the load generator and Vuser behavior for the scenario. Although the default settings correspond to most environments, LoadRunner allows you to modify the settings to customize the scenario behavior. The settings apply to all future scenario runs and generally need to be set only once.

You configure these settings from the **Tools > Options** dialog box. Settings related to load generator behavior apply to all the load generators in a scenario.

> **Note:** You can configure settings for an individual load generator that override the global settings for that particular load generator. For details, see "How to Modify Load Generator Settings" on page 1004.

Global scenario configuration settings are unrelated to the Vuser runtime settings. Runtime settings apply to individual Vusers or scripts and contain information about logging, think time, and the network, the number of iterations, and the browser. For information on setting runtime settings, see "Runtime Settings" on page 301.

# Expert Mode

Expert mode is intended to provide support personnel with access to system information. When you work in the Expert mode, Controller's dialog boxes contain additional options for fine tuning the Controller operation.

To activate the Expert mode, select **Tools** > **Expert Mode.**

To deactivate Expert mode, select **Tools** and clear the **Expert Mode** option.

# Runtime File Storage Locations

When you run a scenario, by default the runtime files are stored **locally on each load generator** (the machine running the Vuser script). The default location of the files is in the temporary folder specified by the load generator's environment variables (on Windows, **TEMP** or **TMP**, and on Linux, **$TMPDIR** or **$TMP**). If no environment variable is defined, the files are saved to the **/tmp** folder.

Alternatively, you can store the runtime files on a "Shared Network" below.

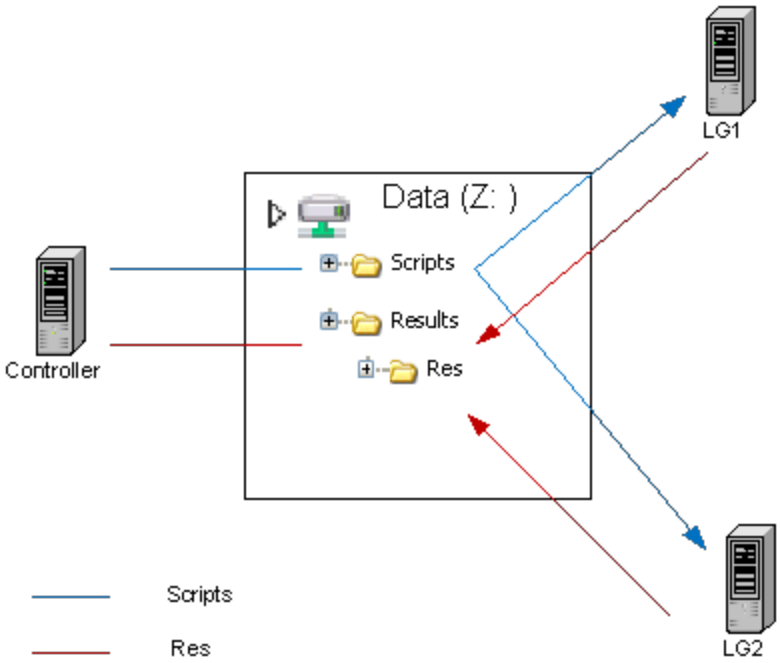
You select where to store runtime files in the **Tools > Options > Runtime File Storage** tab. For details, see "Options > Runtime File Storage Tab" on page 1081.

## Shared Network

A shared network location is a folder or drive to which Controller and all the load generators in the scenario have read and write permission. If all Vusers access their Vuser scripts directly at some shared location, no transfer of script files occurs at runtime. This method may be useful in either of the following situations:

- The file transfer facility does not work.
- The Vuser script files are large and therefore take a long time to transfer.

> **Note:** Keep in mind that Vuser script files are transferred only once during a scenario, and using a shared network drive will increase network traffic.




Using a shared network location may necessitate path translation. Path translation ensures that all participating machines recognize the same network drive. For examples, see "Path Translation" on the next page.

Where possible, we recommend using the same shared network drive mapping for Controller and all load generator machines, to avoid the need to define path translation.

**How to work with a shared network location:**

1. Make sure the scenario scripts are all stored in a shared network location.
2. Create a results folder at a shared network location. This can be at the same shared location as the scripts, or in a different shared location. Define the path in the "Set Results Directory Dialog Box" on page 1088.
3. If path translation is required for the shared locations, enter the necessary information in the "Path Translation Table" on page 1071.
4. To configure global use of the shared network location for the scenario, open the**Tools > Options > Runtime File Storage** tab and select **Shared Network Drive**. Alternatively, define this individually for each load generator running the scenario, in the load generator configuration Runtime File Storage tab.

The primary runtime files are as follows:

| Runtime File Type | Description |
|---|---|

| | |
|---|---|
| **Vuser Script files** | • When you run a Vuser, Controller sends a copy of the associated Vuser script to the load generator. The script is stored in the load generator's temporary runtime folder.<br>• If you use a shared network location, no transfer of script files occurs at runtime. |
| **Result files** | • While you run a scenario, the participating Vusers write their results to the temporary runtime file folder. After scenario execution, these result files are collated or consolidated—results from all of the load generators are transferred to the results folder. After collating the results, the temporary runtime folder is deleted.<br>• If you use a shared network location, all results from all load generators are written to the shared results folder. |

# Path Translation

**Path translation** might be required when storing scripts and runtime data results from a scenario on a shared network drive (**Tools > Options > Runtime File Storage** tab).

## What is path translation?

Path translation is a mechanism used by LoadRunner to convert a remote path name for Controller so that all participating machines recognize the same network drive.

**Example:**

**Example 1.** The scenario runs on a Windows-based machine and includes multiple Vusers running on both Windows-based and Linux load generators. One remote load generator may map the network drive as F, while another load generator maps the same drive as H. In a complex situation such as this, you need to ensure that all participating load generators recognize the same network drive.

**Example 2.** The Scenario Groups/Scripts pane in the Design view contains a list of all the Vuser scripts associated with a scenario, and their locations. A script's location (path) is always based on the Controller machine's mapping of that location. If a load generator maps to the script's path using a different name, path translation is required.

For example, assume that the scenario is running on a Windows-based machine named **pc2**, and that a Vuser script is located on a network drive. The Controller machine maps the network drive as **m:\lr_tests**. If the remote load generator hosting the Vusers also maps the path as **m:\lr_tests**, no translation is necessary. However, if the remote machine maps the path as another drive or path, for example **r:\lr_tests**, you must translate the path to enable the load generator to recognize the script location.

> **Note:** If the Controller and load generator machines are all Windows machines, consider using the **Universal Naming Convention** method instead of manually adding path translation information. On Windows machines, you can tell Controller to convert all paths to UNC, in which case all the machines are able to recognize the path without requiring path translation. An example of UNC format is `\\machine_a\results`.

You add path translation entries in the "Path Translation Table" below.

## Path Translation Table

To translate a path from one Windows-based computer to another, or between Windows-based and Linux machines, you create an entry in the **Path Translation table**. This table contains a list of paths translated into formats that can be recognized by different machines.

Each line of the Path Translation table has the following format:

```
<controller_host> <controller_path> <remote_path> [<remote_host>]
```

| | |
|---|---|
| **controller_host** | Name or type of the machine that is running the Controller.<br>The value of **controller_host** can be:<br>• **<hostname>.** The name of the machine running the Controller, for example, `LOADPC1`<br>• **win.** The Controller is running on a Windows-based computer |
| **controller_path** | Path of a specific folder—as recognized by the Controller.<br>For example, if the folder **scripts** is located on the network drive **r**—as mapped by the Controller—type the path `r:\scripts` in the **controller_path** field. |
| **remote_path** | Path of a specific folder—as recognized by the remote machine.<br>For example, if the folder **scripts** is located on the network drive **n**—as mapped by the remote load generator—type the path `n:\scripts` in the **remote_path** field.<br>If a Vuser on the remote Linux load generator recognizes the above path as `/m/tests`, you would type this path in the **remote_path** field. |
| **remote_host** | Name or type of the remote load generator.<br>For example, if all the remote machines are Linux workstations, you could type `Linux` in the **remote_host** field. The options for the **remote_host** field are the same as the options for the **controller_host** field, listed above. The **remote_host** parameter is optional. |

### Examples

The examples below show the use of the Path Translation table for a Windows-based Controller called `Merlin`.

- **Example 1:** Vusers are running on a Windows 2003 machine, `Oasis`. Merlin maps the network drive as `f:`, while `Oasis` maps it as `g:\loadtest`.

```
merlin f:\ g:\loadtest\ Oasis
```

- **Example 2:** Vusers are running on a Linux machine, `Ultra`. `Ultra` maps the networks drive as `/u/tests/load`.

```
merlin f:\ /u/tests/load/ Ultra
```

- **Example 3:** The mapping of the network drive by the remote load generator `Jaguar`, is identical to the Controller's mapping, so no translation is required. This line can be excluded from the Path Translation table.

```
merlin n:\ n:\ Jaguar
```

- **Example 4:** All Windows-based Vuser load generators map the network drive as `m:\loadtest`.

```
merlin l:\mnt\ m:\loadtest\ win
```

## How to Configure Scenario Options

The following sections describe how to configure options that will be relevant for all your scenarios. You configure these options in the Options dialog box (**Tools > Options**).

### Configure timeout options

Select **Tools > Options > Timeout** tab and specify timeout values for commands related to the load generator. For user interface details, see "Options > Timeout Tab" on page 1083.

If the command is not executed successfully within the timeout period, the load generator status changes to **Error**.

### Configure Vuser runtime settings

Select **Tools > Options > Runtime Settings** tab. You can specify:

- The Vuser quota for a scenario
- How to stop running Vusers
- Whether to use a seed number for random sequencing

For user interface details, see "Options > Runtime Settings Tab" on page 1082.

## Configure general scenario options for Expert mode

Select **Tools > Options > General** tab to specify the following general scenario settings that apply when in Expert mode:

- Specify the folder for data table storage
- Disable collation of log files after a scenario run. For details, see "How to Collate Scenario Run Results" on page 1131.
- Enable multiple IP address allocation. For details, see "Multiple IP Addresses" on page 1048.

For user interface details, see "Options > General Tab" on page 1077.

## Configure the default schedule run mode

Select **Tools > Options > Execution** tab. Under **Default Scheduler** select a default run mode. For user interface details, see "Options > Execution Tab" on page 1076.

For details about schedule run modes, see "Schedule Run Modes" on page 1035.

## Define a command to run after scenario results are collated

Select **Tools > Options > Execution** tab. Under **Post Collate Command**, enter a command to run after collating scenario results. For user interface details, see "Options > Execution Tab" on page 1076.

For more details about collating run results, see "How to Collate Scenario Run Results" on page 1131.

## Configure global runtime file storage options

Select **Tools > Options > Runtime File Storage** tab, and specify where LoadRunner should save and store scenario runtime files collected on remote load generators:

- On the load generator.
- On a shared network drive. For information on this option, see "Runtime File Storage Locations" on page 1068.

For user interface details, see "Options > Runtime File Storage Tab" on page 1081.

> **Note:** LoadRunner applies these settings to all the load generators participating in a scenario. You can change the settings for individual load generators as described in "How to Modify Load Generator Settings" on page 1004.

## Configure monitoring options

Select **Tools > Options > Monitors** tab, and configure the online monitoring settings. For user interface details, see "Options > Monitors Tab" on page 1078.

### Configure debug information options (Expert mode only)

Select **Tools > Options > Debug Information** tab, and determine the extent of the trace to be performed during a scenario run. For user interface details, see "Options > Debug Information Tab" on the next page.

### Configure output display options (Expert mode only)

Select **Tools > Options > Output** tab, and configure how to display running Vusers on the Controller machine. For user interface details, see "Options > Output Tab" on page 1080.

## Options Dialog Box

This dialog box enables you to configure scenario options.

| | |
|---|---|
| **To access** | **Tools > Options** |
| **Important information** | The settings configured in this dialog box:<br>• Generally need to be set only once<br>• Apply to all future scenarios<br>• Apply globally to all the load generators in a scenario.<br>**Note:** You can change the settings for individual load generators (see "How to Modify Load Generator Settings" on page 1004). Individual load generator settings override global scenario settings. |
| **Relevant tasks** | "How to Configure Scenario Options" on page 1072 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Debug Information tab** (Expert mode only) | Enables you to determine the extent of the trace to be performed during a scenario run.<br>For details, see "Options > Debug Information Tab" on the next page. |
| **Execution tab** | Enables you to configure the following miscellaneous scenario settings:<br>• The default schedule run mode for a new scenario<br>• The command to run after collating scenario results<br>For details, see "Options > Execution Tab" on page 1076. |

| | |
|---|---|
| **General tab** (Expert mode only) | Enables you to specify global settings for data table storage, log file collation, and multiple IP address allocation.<br>For details, see "Options > General Tab" on page 1077. |
| **Monitors tab** | Enables you to configure the online monitoring settings.<br>For details, see "Options > Monitors Tab" on page 1078. |
| **Output tab** (Expert mode only) | Enables you to configure how running Vusers are displayed on the Controller machine.<br>For details, see "Options > Output Tab" on page 1080. |
| **Path Translation Table tab** | Enables you to perform path translation when storing result and script files stored on a shared network drive.<br>For details, see "Options > Path Translation Tab" on page 1080. |
| **Runtime File Storage tab** | Enables you to specify where LoadRunner should save and store the runtime files.<br>**Default value:** On the current Vuser machine.<br>For details, see "Options > Runtime File Storage Tab" on page 1081. |
| **runtime settings tab** | Enables you to specify scenario runtime settings.<br>For details, see "Options > Runtime Settings Tab" on page 1082. |
| **Timeout tab** | Enables you to specify timeout values for certain commands related to the load generator.<br>For details, see "Options > Timeout Tab" on page 1083. |

## Options > Debug Information Tab

This tab enables you to configure the settings to determine the extent of the trace to be performed during a scenario run.

| | |
|---|---|
| **To access** | **Tools > Options > Debug Information** tab |
| **Important information** | • This tab is available only when the Controller is operating in the Expert mode.<br>• The debug information is written to the Output window. |
| **Relevant tasks** | "How to Configure Scenario Options" on page 1072 |
| **See also** | "Output Window" on page 1102 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Defaults | Sets the default debug information settings. |

| | |
|---|---|
| **Keep temporary files** | The LoadRunner Agent and Controller create temporary files that collect information such as the parameter file sent to the Vuser, the output compilation file, and the configuration file. The Agent files are saved in **brr** folders in the **TMP** or **TEMP** folder of the Agent machine. The Controller files are saved in **lrr** folders in the **TMP** or **TEMP** folder of the Controller machine. At the end of the scenario, all these files are automatically deleted.<br>When selected, this option instructs the Agent and Controller not to delete these files so that you can use them for debugging. |
| **Trace Flags** | For debugging purposes, you can configure the type of trace performed by LoadRunner during the scenario run. Select the appropriate options to enable the detailed trace. The available trace flags are:<br>• **General**<br>• **File Transfer**<br>• **Incoming Communication**<br>• **Outgoing Communication**<br>The trace information appears in the log file located in the specified Agent log folder.<br>**Note:** Select only the flags relating to your problem. For example, if you encounter specific problems with the transfer of files, select the **File Transfer** flag. |

## Options > Execution Tab

This tab enables you to configure miscellaneous scenario execution settings.

| | |
|---|---|
| **To access** | **Tools > Options > Execution** tab |
| **Relevant tasks** | "How to Configure Scenario Options" on page 1072 |
| **See also** | "Schedule Run Modes" on page 1035 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Default Scheduler** | Enables you to set the default schedule run mode for new scenarios.<br>• **Real-world schedule.** Runs the scenario according to a real-world set of events<br>• **Basic schedule.** Runs a basic schedule, starting the Vuser, running the for a given amount of time, and stopping them.<br>For more details, see "Schedule Run Modes" on page 1035. |

| | |
|---|---|
| **Post Collate Command** | Enables you to define a command that the Controller will run directly after it collates the results of a scenario run.<br>**Example:** You can define a command to run a customer application that runs the Analysis API to extract data.<br><br>**Note:** In the command, you can use the keyword, **%ResultDir%**, to refer to the scenario's results folder. (This keyword is not case sensitive.) |

## Options > General Tab

This tab enables you to specify global settings for data table storage, log file collation, and multiple IP address allocation.

| | |
|---|---|
| **To access** | **Tools > Options > General** tab |
| **Important Information** | This tab is available only when the Controller is operating in Expert mode. |
| **Relevant tasks** | • "How to Configure Scenario Options" on page 1072<br>• "How to Add IP Addresses to a Load Generator" on page 1048 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Multiple IP address mode** | Allocates IP addresses when the multiple IP address option is enabled (**Scenario > Enable IP Spoofer**). The Controller can allocate an IP address per process or per thread. Allocation per thread results in a more varied range of IP addresses in a scenario.<br><br>**Note:** If the IP Spoofer is not enabled, this option is not available. |
| **Data tables global directory** | The network location for data tables used as a source for parameter values.<br><br>**Note:** This setting is only required for scripts created with earlier versions of LoadRunner. |
| **Do not collate log files** | Instructs LoadRunner to collates only the result files, and not the log files. |
| **Override RDP file** | Instructs LoadRunner to override the settings in the RDP file and use the terminal server settings.This applies to RDP Vusers running on load generators for which the Terminal Services Manager is enabled. This option requires you to have Remote Desktop access to the load generator. |

# Options > Monitors Tab

In this tab you can enable the Transaction monitor, configure the behavior of the transaction data, and set the data sampling rate, error handling, debugging, and frequency settings for the online monitors.

| | |
|---|---|
| **To access** | **Tools > Options > Monitors** tab |
| **Relevant tasks** | "How to Configure Scenario Options" on page 1072 |
| **See also** | "How to Set Up a Monitoring Environment" on page 1171 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Defaults | Sets the default timeout values. |
| **Debug** | If **Display debug messages** is selected, debug-related messages are sent to the Output window.<br>For the Network monitor the messages are sent according to the specified debug level (1 - 9). |
| **Error Handling** | Controls the way in which LoadRunner issues error messages:<br>• **Send errors to the Output window.**<br>• **Pop-up an error message box.** |
| **Send** (Expert mode only) | • **Summary.** Sends a summary of the collected data back to the Controller. Use this option if the speed at which the data is transferred is significant to you.<br>• **Raw Data.** Sends all of the data in raw form back to the Controller. Sending the data in raw form saves time because the data does not need to be processed. However, since all of the data is being transferred to the Controller, it may cause more network traffic. |

| | |
|---|---|
| **Transaction Data** | Configures the behavior of data for the Transaction, Data Point, and Web Resource online graphs.<br>• **Enable Transaction Monitor.** Enables the online Vuser Transaction monitor to start monitoring transactions at the start of a scenario.<br>• **Frequency.** The frequency, in seconds, at which the online monitor samples the data to produce the Transaction, Data Point, and Web Resource online graphs. For a small scenario, use a lower frequency, for example, **1**. For a large scenario, use a higher frequency, for example, **3 - 5**. The higher the frequency, the less network traffic there will be. The data is averaged for the frequency period defined, and only one value is sent to the Controller.<br>**Default value:** 5 seconds<br><br>For information on enabling and disabling the Transaction monitor and Web Page Diagnostics, see "Runtime and Transaction Monitoring" on page 1217.<br>**Note:**<br>• Disabling this option conserves resources.<br>• You cannot modify these settings during scenario execution; you must stop the scenario before disabling the monitor or changing its frequency. |
| **Resource Monitoring** | Indicates the sampling rate for resource monitors.<br>**Data Sampling Rate.** The sampling rate is the period of time (in seconds) between consecutive samples. Enter the rate at which LoadRunner samples the scenario for monitoring data. If you increase the sampling rate, the data is monitored less frequently. The default rate is 3 seconds.<br>**Notes:**<br>• This data sampling rate is applied to all server monitors that are subsequently activated. It is not applied to server monitors that have already been activated. To apply the new data sampling rate to activated server monitors, save your scenario and reopen it.<br>• While many monitors use the value provided in this field, some monitors may use a different sampling rate than provided here. For example:<br>  • **Minimum rates.** Some monitors limit the minimum sampling period. For example, the **Windows Resource** monitor, limits the minimum sampling period to 3 seconds. If you set value to less than 3, the monitor will still use a 3 second interval.<br>  • **Custom settings.** Other monitors ignore the Controller's settings and use a sampling rate defined in their own configuration files. For example, the minimum sampling rate for the Oracle monitor is 10 seconds. If the sampling rate is set here at less than 10 seconds, the Oracle Monitor will continue to monitor data at 10 second intervals. |

## Options > Output Tab

This tab enables you to configure how to display running Vusers on the Controller machine.

| | |
|---|---|
| **To access** | **Tools > Options > Output** tab |
| **Important information** | This tab is available only when the Controller is operating in Expert mode. |
| **Relevant tasks** | "How to Configure Scenario Options" on page 1072 |
| **See also** | "Output Window" on page 1102 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Defaults | Sets the default output options. |
| **Configuration of the "Show Vuser" operation** | Specifies how to handle the Vuser logs:<br>• **Max. simultaneously displayed.** The maximum number of Vuser logs that may be displayed simultaneously.<br>**Default value:** 10<br>• **Refresh timeout (milliseconds).** How often the Vuser log should be refreshed.<br>**Default value:** Every 1000 milliseconds |
| **Delete Output window messages upon Reset** | When selected, clears all messages in the Output window when you reset a scenario. |

## Options > Path Translation Tab

This tab enables you to perform path translation when storing result and script files on a shared network drive.

| | |
|---|---|
| **To access** | **Tools > Options > Path Translation** tab |
| **Relevant tasks** | "How to Configure Scenario Options" on page 1072 |
| **See also** | • "Runtime File Storage Locations" on page 1068<br>• "Path Translation" on page 1070 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|

| | |
|---|---|
| **<Path Translation table>** | Displays a list of paths translated into formats that can be recognized by different machines.<br>You can insert comments by typing the **#** symbol at the start of a line in the table.<br>For details, see "Path Translation Table" on page 1071. |
| **Convert to UNC** | When selected, LoadRunner ignores the path translation table and converts all paths to the Universal Naming Convention.<br><br>**Note:** This option can be used only when the Controller and load generator machines are all Windows-based machines. |
| **Mode** | The read/write permissions for the **ppath.mnt** file which contains the path translation table. |
| **Path** | The path to the **ppath.mnt** file which contains the path translation table. |

## Options > Runtime File Storage Tab

This tab enables you to specify where LoadRunner should save runtime files.

| | |
|---|---|
| **To access** | **Tools > Options > Runtime File Storage** tab |
| **Important information** | The runtime file storage options described below apply to all the load generators in a scenario. To change the settings for an individual load generators see "Load Generator Configuration > Runtime File Storage Tab" on page 1014. |
| **Relevant tasks** | • "How to Configure Scenario Options" on page 1072<br>• "How to Prepare a Scenario to Run" on page 1085 |
| **See also** | "Runtime File Storage Locations" on page 1068 |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| **Scripts and results stored** | The default location for storing scripts and results:<br>• **On the current Vuser machine.** LoadRunner saves the runtime files on the load generator that is running the Vuser script.<br>**Note:** If you select this option, you must collate the results before you can perform any analysis. You can wait for LoadRunner to collate the results when you launch LoadRunner Analysis, or you can collate results by choosing **Results** > **Collate Results**. Alternatively, select **Results** > **Auto Collate Results** to automatically collate the results at the end of each scenario run.<br>• **On a shared network drive.** To use this option, you save the Vuser scripts at a shared network location, and define a shared network location where LoadRunner saves the scenario results. For details, see "Runtime File Storage Locations" on page 1068.<br>**Note:** If you select this option, you may need to perform path translation. Path translation ensures that the specified results folder is recognized by the remote load generator. For information about path translation, see "Path Translation" on page 1070. |
| **Default remote host temporary directory** | The default temporary folder (User Temp) for all load generator machines used during run time by LoadRunner and Performance Center. |
| **Do not transfer the parameter files to remote hosts** | Instructs the Controller not to send **.dat** files to the load generator machine. This is useful if you know that the load generator machines already have access to the **.dat** files, or if the **.dat** file size is exceptionally large.<br>This only applies to scripts and groups added from this point onward—it will not affect scripts or groups that have already been added to the scenario. To exclude **.dat** files for these scripts, open the **Files** tab in the "Script Information Dialog Box" on page 978 dialog box, and manually deselect the check boxes adjacent to these files.<br>If you want to include a specific **.dat** file, you can do so by manually selecting the file in the **Files** tab of the "Script Information Dialog Box" on page 978 dialog box. Manually selecting a file will override this global setting. |

## Options > Runtime Settings Tab

This tab enables you to specify scenario runtime settings relating to Vuser quotas, stopping Vusers, and the seed for random sequences.

| To access | **Tools > Options > Runtime Settings** tab |
|---|---|
| **Relevant tasks** | • "How to Configure Scenario Options" on page 1072<br>• "How to Prepare a Scenario to Run" on page 1085 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Defaults | Sets the default runtime setting values. |
| **Use random sequence with seed** | Allows LoadRunner to use a seed number for random sequencing. Each seed value represents one sequence of random values used for test execution. Whenever you use this seed value, the same sequence of values is assigned to the Vusers in the scenario. This setting applies to parameterized Vuser scripts using the **Random** method for assigning values from a data file. It also affects the random percentage of recorded think time (see "Runtime Settings" on page 301).<br>Enable this option if you discover a problem when running the test and want to repeat the test using the same sequence of random values.<br>**Default:** 0 |
| **Vuser Quota** | To prevent your system from overloading, you can set quotas for Vuser activity. The Vuser quotas apply to Vusers on all load generators.<br>• **Number of Vusers that may be initialized at one time - all load generators.** The maximum number of Vusers the load generator can initialize at a time (when an Initialize command is sent).<br>**Default:** 999 |
| **When stopping Vusers** | Controls how Vusers stop running when the **Stop** button is clicked.<br>• **Wait for the current iteration to end before stopping.** (Default) The Vuser completes the iteration it is running before stopping. The Vusers move to the **Gradual Exiting** status and exit the scenario gradually.<br>• **Wait for the current action to end before stopping.** The Vuser completes the action it is running before stopping. The Vusers move to the **Gradual Exiting**status and exit the scenario gradually.<br>• **Stop immediately.** The Vusers stop running immediately. The Vusers move to the **Exiting** status and exit the scenario immediately. |

## Options > Timeout Tab

This tab enables you to specify timeout values for certain commands related to the load generator.

| To access | **Tools > Options > Timeout** tab |
|---|---|

| | |
|---|---|
| **Important information** | LoadRunner enables you to set the timeout interval for commands and Vuser elapsed time.<br>The command timeouts are the maximum time limits for various LoadRunner commands. When a command is issued by the Controller, you set a maximum time for the load generator or Vuser to execute the command. If it does not complete the command within the timeout interval, the Controller issues an error message.<br><br>**Note:** LoadRunner recognizes the fact that the number of active Vusers influences the timeout values. For example, 1000 Vusers trying to initialize will take much longer than 10 Vusers. LoadRunner adds an internal value, based on the number of active Vusers, to the specified timeout value. |
| **Relevant tasks** | "How to Configure Scenario Options" on page 1072 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Defaults | Sets the default timeout values. |
| **Command Timeout (seconds)** | If **Enable timeout checks** is selected, this area defines how LoadRunner should monitor the status of load generators and Vusers after a command is issued by the Controller. If the load generator or Vuser does not complete the command within the timeout interval you specified, the Controller issues an error message.<br>If **Enable timeout checks** is not selected, LoadRunner waits an unlimited time for the load generators to connect and disconnect, and for the Initialize, Run, Pause, and Stop commands to be executed. |
| **Command Timeout: Load Generator** | • **Connect.** The time limit that LoadRunner waits to connect to any load generator. If a connection is not successful within this time, the status of the load generator changes to **Failed**.<br>**Default value:** 120 seconds.<br>• **Disconnect.** The time limit that LoadRunner waits to disconnect from any load generator. If a disconnection is not successful within this time, the status of the load generator changes to **Failed**.<br>**Default value:** 120 seconds. |

| | |
|---|---|
| **Command Timeout: Vuser** | • **Init.** The timeout value for the Initialize command.<br>**Default value:** 180 seconds.<br>• **Run.** The timeout value for the Run command.<br>**Default value:** 120 seconds.<br>• **Pause.** The timeout value for the Pause command.<br>**Default value:** 120 seconds.<br>• **Stop.** The timeout value for the Stop command.<br>**Default value:** 120 seconds. |
| **Update Vuser elapsed time every** | The frequency at which LoadRunner updates the value displayed in the **Elapsed Time** column in the Vusers dialog box.<br>**Default value:** 4 seconds.<br>**Example:** If you select a Vuser and click the **Initialize** button, LoadRunner checks whether the Vuser reaches the **READY** state within 180 seconds (the default **Init** timeout period); if it does not, the Controller issues a message indicating that the **Init** command timed out. |

# Before Running Your Scenario

## How to Prepare a Scenario to Run

This task describes steps to take before you start running your scenario.

### Specify result file name and location

Select **Results** > **Results Settings**.

1. Enter a descriptive name for the result file.

   This is especially useful for cross results analysis, in which LoadRunner superimposes the results of several scenario runs in a single graph and lets you compare the results of multiple scenario runs. Giving each run a descriptive name enables you to distinguish between the results of the multiple runs displayed later in the analysis graph.
2. Enter the full path to the folder where the result file will be stored. This depends on the runtime file storage options configured.

   For details on setting **global** runtime file storage options, see "Options > Runtime File Storage Tab" on page 1081. To set runtime file storage options for an **individual** load generator, see "Load Generator Configuration > Runtime File Storage Tab" on page 1014.

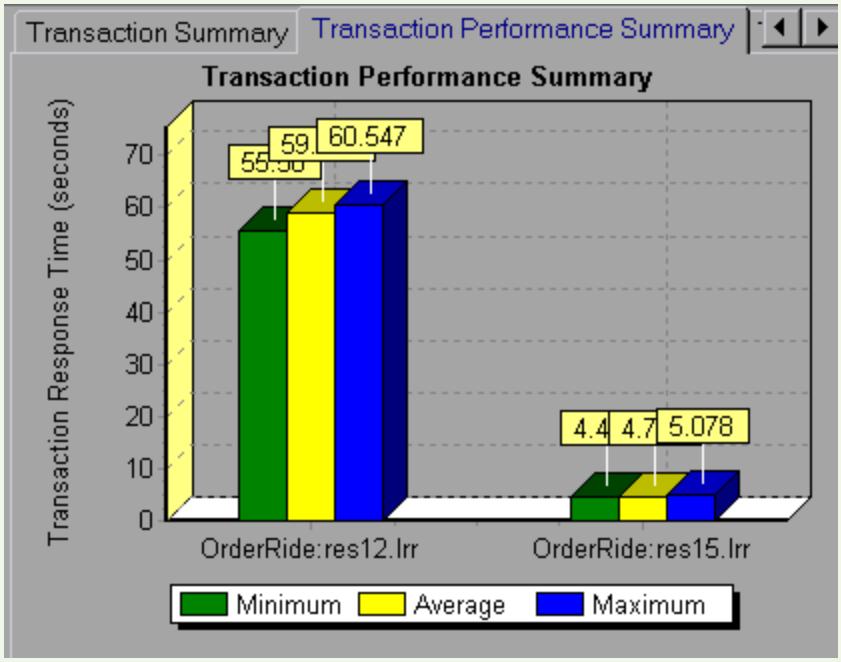
   - If you are using the default file storage setting (local machine), specify a folder in which to store all of the collated results after the scenario run.
   - If you specified a shared network drive as the file storage method, specify the folder to which Vuser groups should write during scenario execution.

3. Select the appropriate options for subsequent scenario runs.

**Note:**

- When comparing the results of scenario runs in Analysis, the graph displays all the result sets by name. For example, the image below displays the superimposed results sets of two scenario runs, **res12**, and **res15**.

  When naming the results files, avoid using the same name when saving to different directories.




- You can use Micro Focus's Web-based test management program, Application Lifecycle Management, to store results to a project. For information, see "Managing Scenarios Using Application Lifecycle Management" on page 1126.

## Specify Scenario Runtime Settings

You can instruct LoadRunner to allow an individual Vuser or the Vusers in a group to complete the iterations they are running before stopping, to complete the actions they are running before stopping, or to stop running immediately. For details, see "Options > Runtime Settings Tab" on page 1082.

## Set up the monitoring environment - optional

LoadRunner enables you to view data generated during the scenario run using the online monitors. Before the run, specify the server machines that the Controller should monitor during the scenario run.

For details, see "How to Set Up a Monitoring Environment" on page 1171.

## Enable Automatic Result Collation - optional

You can enable auto-collation before running the scenario if you are using the local machine file storage setting (default setting). If Automatic Result Collation is enabled, LoadRunner automatically collates the results from all the load generators when the screnario run finishes.

> **Note:** Alternatively, you can collate the results manually after the scenario run is complete. For details, see "How to Collate Scenario Run Results" on page 1131.

To enable automatic collation, select **Results > Auto Collate Results**. When this feature is enabled, **Auto Collate Results** is displayed in the status bar.

If you are working in Expert mode, you can disable the collation of the log files. Select **Tools > Options > General tab > Do not collate log files.**

To set a post-collation command, select **Tools > Options > Execution** tab, and enter the command in the **Post Collate Command** box. For details, see "Options > Execution Tab" on page 1076.

## Enable Auto Load Analysis- optional

To invoke LoadRunner Analysis as soon as the scenario is finished running, select **Results > Auto Load Analysis**. When this is enabled, **Auto Load Analysis** is displayed in the status bar.

## Schedule scenario - Optional

Define a schedule for the scenario. See "Define a Schedule for the Scenario" on page 1039.

## Provide scenario summary information - optional

Select **Scenario > Summary Information**, and enter the scenario's summary information.

## Set up the scenario to run MMS (Media Player)- optional

> **Note:** This protocol is currently supported for replay only. All support will be removed in a future version.

- When running a scenario in controller for an MMS script the scenario may fail if the runtime setting **Network >Speed Simulation** is not set to **Use maximum bandwidth**, and **Miscellaneous >Multithreading** setting is not "Run Vuser as a process".
- If Media Player scripts fail with the error: *Error -86801:Host access denied*, <hostname> not available or missing, add the **mms_disable_host_check** function to the script.

### Set up the scenario to run GUI Vusers - optional

If you have integrated a Unified Functional Testing (UFT) script into the scenario:

- Set up runtime settings for the script in UFT. Runtime settings from LoadRunner are not applied to UFT scripts.
- Ensure that UFT is closed before running the scenario.

> **Note:** You can run only one GUI Vuser concurrently per machine.

### See also:

- "Designing Scenarios" on page 942.
- "Define a Schedule for the Scenario" on page 1039.

## Set Results Directory Dialog Box

This dialog box enables you to set the location in which the Controller saves scenario run results.

| | |
|---|---|
| **To access** | **Results > Results Settings** |
| **Important information** | If you have an open connection to an ALM project, the Controller saves the results to a test set. You can also save the results directly to disk using the standard file system. |
| **Relevant tasks** | "How to Prepare a Scenario to Run" on page 1085 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Automatically create a results directory for each scenario execution** | Instructs LoadRunner to create a unique results folder for each scenario run. By default, the result names are **res1**, **res2**, **res3**, and so on. |
| **Automatically overwrite existing results directory without prompting for confirmation** | Instructs LoadRunner to automatically overwrite previous result sets, without prompting the user. |

| | |
|---|---|
| **Directory** | A location in the file system under the Results folder, to which the Controller should save the results.<br><br>**Note:** Do not specify an admin/administrator folder, as non-admin users without sufficient permissions may be unable to write to such a folder. |
| **File System** | Displays the default LoadRunner folder path. |
| **ALM** (only when connected to ALM) | Enables you to save the results to an Application Lifecycle Management test set. |
| **Results Name** | Specify a name for the run results.<br>LoadRunner allows you to give descriptive names to each result set. This is especially useful for cross results analysis, in which LoadRunner superimposes the results of several scenario runs in a single Analysis graph and lets you compare the results of multiple scenario runs. The descriptive graph names enable you to distinguish between the results of the multiple runs. |
| **Results Path** | Displays the location for the results as specified in **Results Name** and **Directory**. Avoid using the same name with different paths. Only the names appear on the Analysis graphs. If the result names are identical it will be difficult to distinguish between the runs. |

## Summary Information Dialog Box

This dialog box enables you to provide a detailed description of the scenario.

| | |
|---|---|
| **To access** | **Scenario > Summary Information** |
| **Relevant tasks** | "How to Prepare a Scenario to Run" on page 1085 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Author** | The name of the scenario's author |
| **Description** | A description of the scenario |
| **Scenario Path** | The name and location of the scenario definition file (.lrs) |
| **Subject** | A subject name or short title for the scenario |

# Rendezvous Points

## Rendezvous Points Overview

During a scenario run, you can instruct multiple Vusers to perform tasks simultaneously by using **rendezvous points**. A rendezvous point creates intense user load on the server and enables LoadRunner to measure server performance under load.

Suppose you want to measure how a Web-based banking system performs when ten Vusers simultaneously check account information. To emulate the required user load on the server, you instruct all the Vusers to check account information at exactly the same time.

You ensure that multiple Vusers act simultaneously by creating a rendezvous point. When a Vuser arrives at the rendezvous point, it is held there by Controller. You then set a **rendezvous policy** according to which Controller releases the Vusers from the rendezvous point either when the required number of Vusers arrives, or when a specified amount of time has passed.

You define rendezvous points in the Vuser script. For information about inserting rendezvous points into Vuser scripts, see "Rendezvous Points" on page 343.

Using Controller, you can influence the level of server load by selecting:

- Which of the rendezvous points will be active during the scenario
- How many Vusers will take part in each rendezvous

For example, to test a bank server, you could create a scenario that contains two rendezvous points. The first rendezvous ensures that 1000 Vusers simultaneously deposit cash. The second rendezvous ensures that another 1000 Vusers simultaneously withdraw cash. If you want to measure how the server performs when only 500 Vusers deposit cash, you can deactivate the **withdraw** rendezvous, and instruct 500 Vusers to participate in the **deposit** rendezvous only.

## How to Set Up a Rendezvous in a Scenario

This task describes set up rendezvous points and policies in a scenario.

1. Prerequisites

   Add Vuser scripts that contain rendezvous points. For details, see "Rendezvous Points" on page 343.

   Rendezvous points are only effective for group mode—not percentage mode. For details, see "How to Change the Scenario Mode (Manual Scenario)" on page 954.

   When you add a Vuser group or script to the scenario, LoadRunner scans the included scripts for rendezvous points and adds them to a central list of rendezvous points. Select **Scenario > Rendezvous** to view this list. For user interface details, see "Rendezvous Information Dialog Box" on the next page.

> **Note:** In goal-oriented scenarios, a script's rendezvous points are disabled.

2. Set the level of emulated user load

   Select the rendezvous points to take part in the scenario, and the number of Vusers to participate in each rendezvous.

   You can temporarily disable a rendezvous and exclude it from the scenario. You can disable a rendezvous point for all Vusers in a scenario, or you can temporarily disable specific Vusers from participating in the rendezvous.

   By disabling and enabling a rendezvous, you influence the level of server load.

3. Set the attributes for the rendezvous policy - Optional

   In the Rendezvous Information dialog box, for each rendezvous:

   a. Select the rendezvous, and click the **Policy** button.

   b. In the Policy dialog box, set the **policy** attributes as follows:

   - **Release.** The number of Vusers to be released from a rendezvous at one time.
   - **Timeout.** The time the Controller waits before releasing Vusers from a rendezvous.

   For user interface details, see "Rendezvous Information Dialog Box" below.

## Rendezvous Information Dialog Box

This dialog box enables you to view and modify the attributes of each rendezvous point in the scenario. It displays general information about the rendezvous point: which script is associated with the rendezvous, and release history.

| | |
|---|---|
| **To access** | Manual scenario > **Design** tab > **Scenario** > **Rendezvous** |
| **Important information** | Available only if one of the Vuser scripts participating in the scenario contains a rendezvous point. |
| **Relevant tasks** | "How to Set Up a Rendezvous in a Scenario" on the previous page |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Disables the rendezvous, excluding it from the scenario, thereby influencing the level of server load. |
| | Enables a disabled rendezvous point. |

| | |
|---|---|
| Policy... | Opens the Policy dialog box, where you can set the number Vusers to be released from a rendezvous at a time, as well as the amount of time the Controller waits before releasing Vusers from a rendezvous.<br>• **Release when X% of all Vusers arrive at the rendezvous.** Releases the Vusers only when the specified percentage of all Vusers arrives at the rendezvous point.<br>**Note:** This option interferes with the scheduling of your scenario. If you select this option, your scenario will not run as scheduled.<br>• **Release when X% of all running Vusers arrive at the rendezvous.** Releases the Vusers only when the specified percentage of all Vusers running in the scenario arrives at the rendezvous point.<br>• **Release when X Vusers arrive at the rendezvous.** Releases the Vusers only when the specified number arrives at the rendezvous point.<br>• **Timeout between Vusers.** The timeout value (in seconds). After each Vuser arrives at the rendezvous point, LoadRunner waits up to the maximum timeout period specified for the next Vuser to arrive. If the next Vuser does not arrive within the timeout period, the Controller releases all the Vusers from the rendezvous. Each time a new Vuser arrives, the timer is reset to zero. You set a timeout for each rendezvous point.<br>**Default value:** 30 seconds |
| Disable VUser | Disables a Vuser from taking part in the rendezvous. |
| Enable VUser | Enables a Vuser to take part in the rendezvous. |
| Reset | Resets the Status Information, removing the information currently displayed. |
| Release | While a scenario is running, enables you to manually release Vusers from a rendezvous before the Controller releases them. Use this option if you want the scenario to continue running even though all the Vusers did not reach the rendezvous. |
| **Rendezvous** | The names of the rendezvous points in the scenario. |
| **Scripts** | The Vuser scripts associated with the rendezvous points. |

| | |
|---|---|
| **Status Information** | During and after a scenario, displays:<br>• **Current Status.** The number of Vusers that arrived at the rendezvous point out of the total number of Vusers assigned to the rendezvous.<br>• **Time.** The time at which the Vusers at the rendezvous point were released.<br>• **Reason.** The reason the Vusers at the rendezvous point were released. The possible reasons are **Timeout** or **Arrived**. |
| **Vusers** | The Vusers associated with the rendezvous points. |

# Running Scenarios in Controller

## Running Scenarios Overview

After planning, designing and scheduling your scenario, you run it to create load on your application and to test its performance.

### Before run

When running a manual scenario in Percentage mode, you can change the way LoadRunner loads the Vusers. By default, when running a group assigned to use **All Load Generators**, Controller calculates the number of Vusers per load generator, and ramps up all Vusers, one load generator after another. If the initialization section uses a lot of resources, this may affect the performance on the load generators. LoadRunner allows you to distribute the Vusers in a round-robin fashion.

- For Vusers in groups configured to use **All Load Generators** LoadRunner uses a round-robin method to ramp up the groups between all of the load generators.
- For all Vusers in a group not configured to run to use **All Load Generators**, LoadRunner uses a round-robin method to ramp up the Vusers between all of the group's load generators.

For details on how to enable round-robin ramp up, see "How to Run a Scenario" on the next page.

### Start of run

When you instruct LoadRunner to begin the scenario run, Controller checks the scenario configuration information, invokes the applications that were selected to run with the scenario, and then distributes each Vuser script to its designated load generator. When the Vusers are ready, they start running their scripts.

As the scenario starts, in the Scenario Groups pane you can watch Vusers gradually start running.

### During run

During the scenario run, you can see a synopsis of the running scenario in the Scenario Status pane. You can also drill down to see which Vuser actions are causing the application problems.

Controller's online graphs display performance data collected by the monitors. You use this information to isolate potential problem areas in your system.

### End of run

The scenario ends when all the Vusers have completed their scripts, when the duration runs out, or when you terminate it.

At the conclusion of the test run, the Scenario Status pane shows the **Down** status. This indicates that the Vusers have stopped running.

## How to Run a Scenario

This task describes how to run a scenario.

1. Prerequisite

   Open an existing scenario, or design a new one.

   - To design a manual scenario, see "How to Design a Manual Scenario" on page 953.
   - To design a goal-oriented scenario, see "How to Design a Goal-Oriented Scenario" on page 951.
2. Prepare to run the scenario

   Before you run the scenario, specify a location for the scenario results and other runtime related settings. For details, see "How to Prepare a Scenario to Run" on page 1085.
3. Set the ramp up method to round-robin - optional

   To enable the round-robin ramp up, search for the **wlrun7.ini** file, located in the %windir%. In the **[ScenarioDesign]** section, set the **DitributionByHostsInPrecentageMode** flag to 1. For details, see "Running Scenarios Overview" on the previous page.
4. Run the scenario

   In the Run tab, click the **Start Scenario** button to begin running the scenario. The scenario runs according to its defined schedule.

   > **Note:** When the Controller starts the scenario, it checks the versions of each VuGen script and its designated load generator machine. The Controller detects a mismatch if the version of the script is later than that of the load generator, or if either the script or load generator version is undefined (for example in earlier load generator versions).
   >
   > If there is a mismatch, the Controller log shows a message that the versions do not match and the script may not run properly. You can stop the scenario and try to resolve the mismatch by upgrading your load generator, for example. Alternatively, you can let the scenario continue running and determine whether there were any other errors.
5. Manually control the behavior, addition, and stopping of Vusers during the scenario run - optional

   You can do the following during the scenario run:

- **Control the behavior of Vuser groups.** You can initialize, run, and stop Vuser groups during the scenario run.

  To initialize, run, or stop an entire Vuser group, select the group in the Scenario Groups pane, and click the desired button on the main Controller toolbar:

  - **Initialize Vusers.**
  - **Run Vusers.**
  - **Stop Vusers immediately.**
  - **Stop Vusers gradually.**

- **Run or stop individual Vusers.** You can run or stop specific Vusers within a Vuser group. For user interface details, see "Vusers Dialog Box" on page 981.
- **Initialize/Run additional Vusers, or stop currently running Vusers.** You can manually control the addition of new Vusers to a running scenario, as well as stop running Vusers. For user interface details, see "Run/Stop Vusers Dialog Box" on page 1108.

6. View a log containing runtime information about each running Vuser - Optional

   For user interface details, see "Vuser Script Log" on page 1112.

7. Release Vusers from a rendezvous before the Controller releases them - Optional

   For more information, see "Rendezvous Points" on page 1090.

8. Log execution notes during the scenario run - Optional

   The Controller provides you with a dialog box in which you can log comments while a scenario is running. To open the dialog box select **Scenario > Execution Notes**. The notes are automatically saved by clicking **OK** to close the dialog box.

9. Monitor the scenario - optional

   During the scenario run, you can view data collected by the online monitors using the online monitor graphs. If you did not set up the monitors before you started the run, you can do so during the run. The data collected by the monitors can be viewed using the LoadRunner online graphs.

   - For details about setting up the online monitors, see "How to Set Up a Monitoring Environment" on page 1171.
   - For details about viewing the monitor graphs, see "Online Monitor Graphs" on page 1053.

10. Collate run results

    If you are using the default file storage setting—local machine, when the scenario run is complete, the run results must be collated or consolidated in preparation for result analysis. If LoadRunner is not set up to collate the results automatically upon completion of the run, you need to collate the results manually after the run.

    Select **Results > Collate Results > Collate Results**. For details, see "How to Collate Scenario Run Results" on page 1131.

    For details about result collation, see "Collating Run Data" on page 1131.

# Use-Case Scenario - Initialize/Run Additional Vusers or Stop Running Vusers

This use-case scenario describes how Peter can manipulate the behavior of Vusers during a scenario run, irrespective of their defined schedules. The examples will show how he can initialize or run specified numbers of additional Vusers, or stop specified numbers of running Vusers.

> **Note:** The examples presented in this section demonstrate options in the **Run/Stop Vusers** dialog box. Not all information relevant for working with this dialog box necessarily appears here. For full information about working with the **Run/Stop Vusers** dialog box, see "Run/Stop Vusers Dialog Box" on page 1108.

## Initialize/Run additional Vusers in Vuser group mode

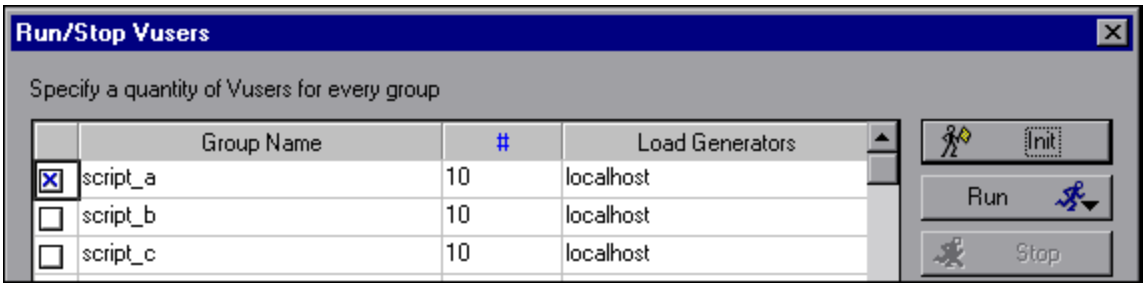
The following procedure shows how Peter can initialize and run additional Vusers when he is working in Vuser Group mode:

> **Note:** The options to initialize or run additional Vusers can be done as two separate actions, with no connection to each other. They are being shown here together as a single workflow for demonstrative purposes only.

1. Initialize additional Vusers

   If he wants to initialize **ten** Vusers in **Script_A** immediately, and not wait for them to initialize as per their defined schedules, in the Run tab, he clicks **Run/Stop Vusers** to open the Run/Stop Vusers dialog box.

   In the dialog box, he makes sure that only the check box by **script_a** is selected, and he enters **10** in the **#** (number) column.



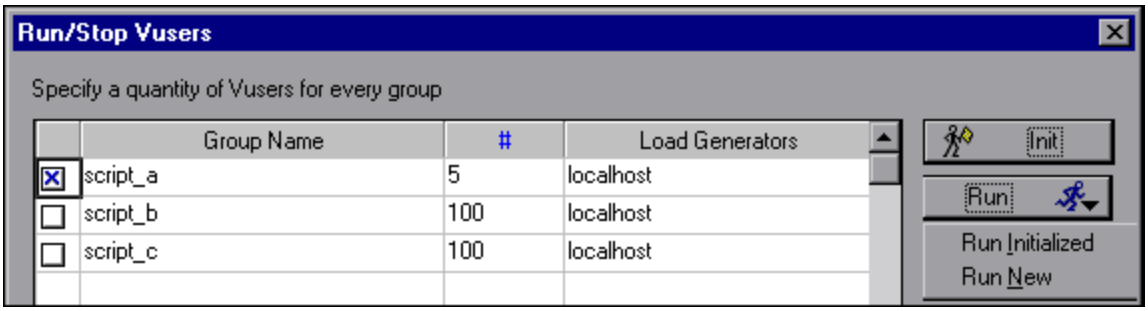

   To initialize these Vusers, he clicks **Initialize**. Ten Vusers are immediately initialized and move to the **Ready** state. From there, they run according to their defined schedules.

| Group Name | Down | Pending | Init | Ready | Run | Rendez | Passed | Failed | Error | Gradual Exiting | Exiting | Stopped |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 10 | 0 | 0 | 10 | 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Script_A | | | | 10 | 5 | | | | | | | |
| Script_B | 5 | | | | 5 | | | | | | | |
| Script_C | 5 | | | | 5 | | | | | | | |

> **Note:** The additional initialized Vusers are taken from the Vusers that are in the **Down** state. If Peter initializes a greater number of Vusers than there are in the **Down** state, then all of them will be initialized. In the example above, there were five Vusers in the **Down** state. All of them have been initialized, while an additional five have been created.

2. Run additional Vusers

If Peter then wants to run five additional Vusers in **Script_A** immediately, and not wait for them to run as per their defined schedules, in the Run/Stop Vusers dialog box, he makes sure that only the check box by **script_a** is selected, and he enters **5** in the **#** (number) column.




He then has the following two options for how to run these additional Vusers:

- **Run initialized Vusers.** He can run five Vusers from those he initialized in the previous step. To do this, he clicks the arrow on the **Run** button and selects **Run Initialized**. Five Vusers immediately move from the **Ready** state to the **Run** State.

| Group Name | Down | Pending | Init | Ready | Run | Rendez | Passed | Failed | Error | Gradual Exiting | Exiting | Stop |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 10 | 0 | 0 | 5 | 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Script_A | | | | 5 | 10 | | | | | | | |
| Script_B | 5 | | | | 5 | | | | | | | |
| Script_C | 5 | | | | 5 | | | | | | | |

- **Run new Vusers.** He can create and run five new Vusers, with no effect on those he initialized in the previous step. To do this, he clicks the arrow on the **Run** button and selects **Run New**. Five Vusers are immediately created and move directly to the **Run** state.

| Group Name | Down | Pending | Init | Ready | Run | Rendez | Passed | Failed | Error | Gradual Exiting | Exiting | Stop |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 10 | 0 | 0 | 10 | 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Script_A | | | | 10 | 10 | | | | | | | |
| Script_B | 5 | | | | 5 | | | | | | | |
| Script_C | 5 | | | | 5 | | | | | | | |

> **Note:** If there were still Vusers in the **Down** state, the new Vusers would be taken from them.

## Initialize/Run additional Vusers in Percentage mode

The following procedure shows how Peter can initialize and run additional Vusers when he is working in Percentage mode:

> **Note:** The options to initialize or run additional Vusers can be done as two separate actions, with no connection to each other. They are being shown here together as a single workflow for demonstrative purposes only.

1. Initialize new Vusers

   If he wants to initialize ten Vusers in **Script_A** immediately, and not wait for them to initialize as per their defined schedules, in the Run tab, he clicks **Run/Stop Vusers** to open the Run/Stop Vusers dialog box.

   In the dialog box, he makes sure that only the check box by **script_a** is selected, and he enters **10** in the **Distribute X Vusers among all the scripts** box. It is also important that the percentage values for **Script_B** and **Script_C** are set to **0%**. See the note below for a detailed explanation why.

   > **Note:** When a check box is deselected, no Vusers are distributed to that script. However, the amount of Vusers that *would* have been assigned to it are not redistributed to the scripts that remain selected, unless you specify **0%** in the percentage column.

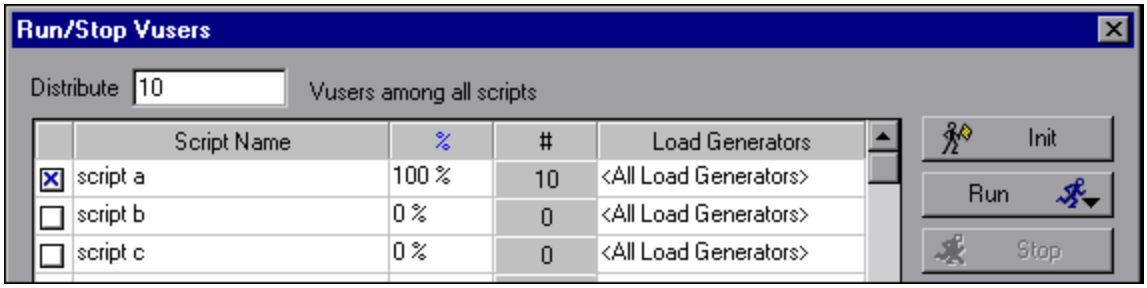
   For example, in our use-case scenario, when Peter enters **10** in the **Distribute X Vusers among all the scripts** box, LoadRunner automatically distributes these Vusers as equally as possible among the available scripts, that is:

   - **script_a**. 4 Vusers
   - **script_b**. 3 Vusers
   - **script_c**. 3 Vusers

   However, should Peter wish to distribute all **10** Vusers to **script_a**, it is not sufficient to simply deselect **script_b** and **script_c**. This only ensures that no Vusers are added to these scripts, but it *does not* change the original Vuser distribution.

In other words, should Peter complete the step now, the four Vusers that are assigned to **script_a** will be added, while the three each assigned to **script_b** and **script_c** will not, though they will still appear under the **#** (number) column. To distribute these six Vusers to **script_a** instead, Peter must first change the **%** (percentage) columns for these scripts to **0%**.

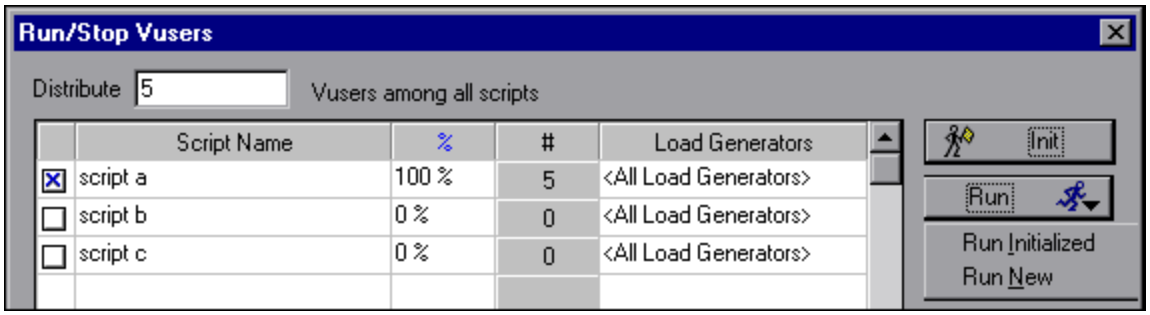
He then clicks **Initialize**. Ten Vusers are immediately initialized and move to the **Ready** state. From there, they run according to their defined Scheduler settings.




> **Note:** The additional initialized Vusers are taken from the Vusers that are in the **Down** state. If you initialize a greater number of Vusers than there are in the **Down** state, then all of them will be initialized. In the example above, there were five Vusers in the **Down** state. All of them have been initialized, while an additional five have been created.

- Run additional Vusers

If Peter then wants to run five additional Vusers in **Script_A** immediately, and not wait for them to run as per their defined schedules, in the Run/Stop Vusers dialog box, he makes sure that only the check box by **script_a** is selected, and he enters **5** in the **Distribute X Vusers among all the scripts** box at the top of the dialog box. It is also important that the percentage values for **Script_B** and **Script_C** are set to **0%**. See the note in the step above for a detailed explanation why.




He then has the following two options for how to actually run these additional Vusers:

- **Run initialized Vusers.** He can run five Vusers from those he initialized in the previous step. To do this, he clicks the arrow on the **Run** button and selects **Run Initialized**. Five Vusers immediately move from the **Ready** state to the **Run** State.

| Group Name | Down | Pending | Init | Ready | Run | Rendez | Passed | Failed | Error | Gradual Exiting | Exiting | Stop |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 10 | 0 | 0 | 5 | 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Script_A | | | | 5 | 10 | | | | | | | |
| Script_B | 5 | | | | 5 | | | | | | | |
| Script_C | 5 | | | | 5 | | | | | | | |

- **Run new Vusers.** He can create and run five new Vusers, with no effect on those he initialized in the previous step. To do this, he clicks the arrow on the **Run** button and selects **Run New**. Five Vusers are immediately created and move directly to the **Run** state.

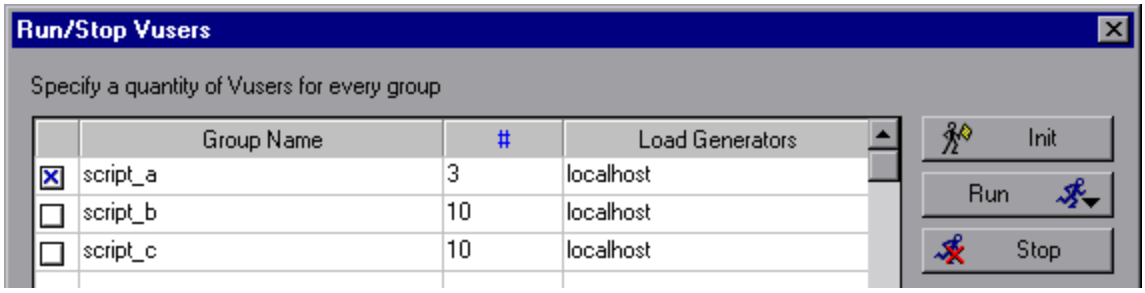
| Group Name | Down | Pending | Init | Ready | Run | Rendez | Passed | Failed | Error | Gradual Exiting | Exiting | Stop |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 10 | 0 | 0 | 10 | 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Script_A | | | | 10 | 10 | | | | | | | |
| Script_B | 5 | | | | 5 | | | | | | | |
| Script_C | 5 | | | | 5 | | | | | | | |

> **Note:** If there were still Vusers in the **Down** state, the new Vusers would be taken from them.

## Stop running Vusers in Vuser group mode

If Peter wants to stop **three** of the **five** running Vusers in **Script_A**, and not wait for them to stop as per their defined schedules, in the Run tab, he clicks **Run/Stop Vusers** to open the Run/Stop Vusers dialog box.

In the dialog box, he makes sure that only the check box by **script_a** is selected, and he enters **3** in the **#** (number) column.



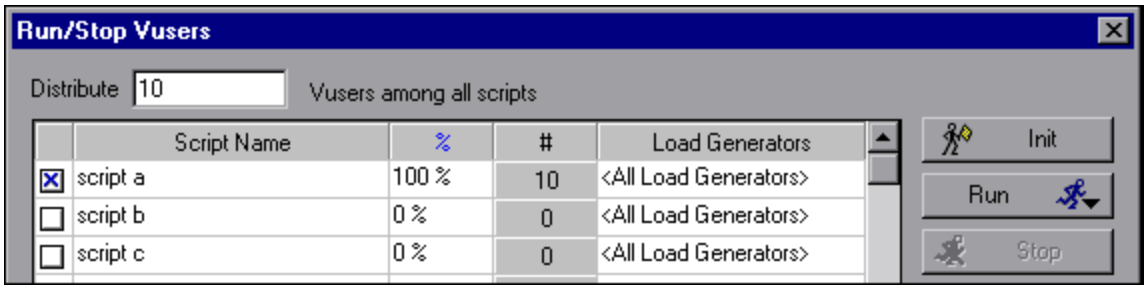

He then clicks **Stop**. Three of the running Vusers in **Script_A** move from the **Run** state to the **Gradual Exiting** state.

| Group Name | Down | Pending | Init | Ready | Run | Rendez | Passed | Failed | Error | Gradual Exiting | Exiting | Stopped |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 | 12 | 0 | 0 | 0 | 0 | 6 | 0 | 12 |
| Script_A | | | | | 2 | | | | | 4 | | 4 |
| Script_B | | | | | 5 | | | | | 1 | | 4 |
| Script_C | | | | | 5 | | | | | 1 | | 4 |

## Stop running Vusers in Percentage mode

If Peter wants to stop **three** of the **five** running Vusers in **Script_A**, and not wait for them to stop as per their defined schedules, in the Run tab, he clicks **Run/Stop Vusers** to open the Run/Stop Vusers dialog box.

In the dialog box, he makes sure that only the check box by **script_a** is selected, and he enters **3** in the **Distribute X Vusers among all the scripts** box. It is also important that the percentage values for **Script_B** and **Script_C** are set to **0%**. See the note below for a detailed explanation why.




**Note:** When a check box is deselected, no Vusers are distributed to that script. However, the amount of Vusers that *would* have been assigned to it are not redistributed to the scripts that remain selected, unless you specify **0%** in the percentage column.

For example, in our use-case scenario, when Peter enters **3** in the **Distribute X Vusers among all the scripts** box, LoadRunner automatically distributes these Vusers as equally as possible among the available scripts, that is:

- **script_a**. 1Vuser
- **script_b**. 1Vuser
- **script_c**. 1Vuser

However, should Peter wish to distribute all **3** Vusers to **script_a**, it is not sufficient to simply deselect **script_b** and **script_c**.This only ensures that no Vusers are added to these scripts, but it *does not* change the original Vuser distribution.

In other words, should Peter complete the step now, the single Vuser that is assigned to **script_a** will be stopped, while the one each assigned to **script_b** and **script_c** will not, though they will still appear under the **#** (number) column. To distribute these two Vusers to **script_a** instead, Peter must first change the **%** (percentage) columns for these scripts to **0%**.

He then clicks **Stop**. Three of the running Vusers in **Script_A** move from the **Run** state to the **Gradual Exiting** state.

| Group Name<br>3 | Down<br>0 | Pending<br>0 | Init<br>0 | Ready<br>0 | Run<br>12 | Rendez<br>0 | Passed<br>0 | Failed<br>0 | Error<br>0 | Gradual Exiting<br>6 | Exiting<br>0 | Stopped<br>12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Script_A | | | | | 2 | | | | | 4 | | 4 |
| Script_B | | | | | 5 | | | | | 1 | | 4 |
| Script_C | | | | | 5 | | | | | 1 | | 4 |

## Execution Notes Dialog Box

This page dialog box enables you to log comments while a scenario is running.

| | |
|---|---|
| **To access** | Select **Scenario** > **Execution Notes** |
| **Relevant tasks** | "How to Run a Scenario" on page 1094 |
| **Important Information** | Only enabled while the scenario is running. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<note writing area>** | Enter your notes in this area. |

## Output Window

This window displays error, notification, warning, debug, and batch messages that are sent to the Controller by the Vusers and load generators during a scenario run.

| | |
|---|---|
| **To access** | Use one of the following:<br>• **Run** tab > **Scenario Status** pane > **Errors** > <br>• Select **View** > **Show Output** |
| **Important information** | • LoadRunner clears the messages in the Output window at the start of each scenario execution. If you reset a scenario, messages remain in the Output window unless you instruct LoadRunner to delete messages from the window upon reset. For more information, see "Options > Output Tab" on page 1080.<br>• The Summary tab is displayed by default when you open this window. |
| **Relevant tasks** | • "How to Run a Scenario" on page 1094<br>• "How to Configure Scenario Options" on page 1072 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Filtered Tab** | See "Filtered Tab" on the next page |

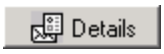
| | |
|---|---|
| **Summary Tab** | See "Summary Tab" on the next page |

## Filtered Tab

This tab displays a drilled down view by message, Vuser, script, or load generator. For example, if you drill down on the Vuser column, the Filtered tab displays all the messages with the code you selected, grouped by the Vusers that sent the messages.

| | |
|---|---|
| **To access** | **Output** window > **Filtered** tab. Click the blue link on the column about which you wish to view more information. |
| **Important information** | The tab appears when you click on a blue link in the Summary tab. |
| **See also** | "Summary Tab" on the next page |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | **Previous/Next View.** Enables you to move between the various drill down levels. |
|   | Displays the full text of the selected output message in the Detailed Message Text area at the bottom of the Output window. |
| | **Export the view.** Saves the output to a specified file. |
| | Refreshes the Filtered tab with new log information that arrived in the Output window updated in the Summary tab. |
| **<Message icon>** | Displays an icon indicating the type of message by which the current Output view is filtered. |
| **Active Filter** | Displays the category or categories by which the current Output view is filtered. |

| | |
|---|---|
| **Viewed By** | Displays the name of the column on which you selected to drill down. The following icons indicate the various message types:<br>• Batch<br>• Debug<br>• Errors<br>• Notifications<br>• Warnings |
| **Detailed Message Text** | Displays the full text of the selected output message when the **Details** button is selected. |
| **Message** | Displays all instances of the sample message text. |
| **Script** | The script on which the message was generated. If you click the blue link, VuGen opens displaying the script. |
| **Action** | The action in the script where the message was generated. If you click the blue link, VuGen opens the script to the relevant action. |
| **Line #** | The line in the script where the message was generated. If you click the blue link, VuGen opens the script and highlights the relevant line. |
| **# Lines** | The total number of lines in the script where the Vuser failed. |
| **Time** | The time the message was generated. |
| **Iteration** | The iteration during which the message was generated. |
| **Vuser** | The Vuser that generated the message. |
| **Generator** | The load generator on which the message was generated. If you click the blue link, the Load Generator dialog box opens. |
| **# Messages** | The number of messages generated by a specific Vuser. |
| **# Diff Texts** | |

## Summary Tab

This tab displays summary information about the messages sent during a scenario run.

| | |
|---|---|
| **To access** | **Output** window > **Summary** tab |
| **Important information** | You can drill down further on any information displayed in blue. |
| **See also** | "Filtered Tab" on page 1103 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Displays the full text of the selected output message in the Detailed Message Text area at the bottom of the Output window. |
| | **Remove all messages.** Clears all log information from the Output window. |
| | **Export the view.** Saves the output to a specified file. |
| | • **Freeze.** Stops updating the Output window with messages.<br>• **Resume.** Resumes updating the Output window with messages. The newly updated log information is displayed in a red frame. |
| **Detailed Message Text** | Displays the full text of the selected output message when you click the **Details** button. |
| **Generators** | Displays the number of load generators that generated messages with the specified message code. |
| **Help** | Displays an icon if there is a link to troubleshooting for the message. |
| **Message Code** | Displays the code assigned to all similar messages. The number in parentheses indicates the number of different codes displayed in the Output window. |
| **Sample Message Text** | Displays an example of the text of a message with the specified code. |
| **Scripts** | Displays the number of scripts whose execution generated messages with the specified code. |
| **Total Messages** | Displays the total number of sent messages with the specified code. |

| | |
|---|---|
| **Type** | The type of message being displayed. The following icons indicate the various message types. For more information about each type, see **Type of Message** below:<br>• Batch<br>• Debug<br>• Errors<br>• Notifications<br>• Warnings |
| **Type of Message** | Filters the output messages to display only certain message types. Select one of the following filters:<br>• **All messages.** Displays all message types.<br>• **Batch.** Sent instead of message boxes appearing in the Controller, if you are using automation.<br>• **Debug.** Sent only if the debugging feature is enabled in the Controller. (Expert mode: **Tools > Options > Debug Information**). For more information, see "Options > Debug Information Tab" on page 1075.<br>• **Errors.** Usually indicate that the script failed.<br>• **Notifications.** Provides runtime information, such as message sent using **lr_output_message**.<br>• **Warnings.** Indicates that the Vuser encountered a problem, but the scenario continued to run. |
| **Vusers** | Displays the number of Vusers that generated messages with the specified code. |

## Run Tab

The Run tab enables you to run and monitor scenarios.

| | |
|---|---|
| **To access** | **Run** tab |
| **Relevant tasks** | "How to Run a Scenario" on page 1094 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|

| | |
|---|---|
| Start Scenario | Instructs the Controller to initialize the Vusers and distribute them to their designated load generators, where they begin running their Vuser scripts.<br><br>**Note:**<br>• The Controller begins running the scenario according to the start time defined in the scenario schedule.<br>• We do not recommend changing the Time/Date and Time Zone settings on the Controller and load generators during the load test run. |
| Stop<br><br>Stop Now | Terminates the scenario.<br>The behavior depends on your selection in the **Tools** > **Options** > **Runtime Settings** tab:<br>• If you selected **Stop immediately,** all Vusers in the scenario move to the **Exiting** status.<br>• If you selected **Wait for the current iteration to end before exiting,** or **Wait for the current action to end before exiting**, the button text changes to **Stop Now** and the Vusers status changes to **Gradual Exiting**. To stop the Vusers immediately, click **Stop Now**.<br>For more information about the runtime settings options, see "Options > Runtime Settings Tab" on page 1082. |
| Reset | Resets all Vuser groups to the **Down** status. |
| | Opens the Vusers dialog box, where you can view the status of each of the Vusers in a Vuser group. |
| Run/Stop Vusers... | Opens the Run/Stop Vusers dialog box, where you can activate additional Vusers. |
| Pause Scheduler<br><br>Resume Scheduler | Pauses or resumes the scenario schedule. |
| **<Graph Legend>** | Displays statistics for the selected graph. For more information, see "Online Monitor Graphs" on page 1053. |
| **<Graph Viewing Pane>** | Displays the graphs that are listed in the **Available Graphs** pane. For more information, see "Online Monitor Graphs" on page 1053.<br>**Default:** Displays four graphs |

| | |
|---|---|
| **Available Graphs** | Displays the available online monitor graphs. For more information, see "Online Monitor Graphs" on page 1053. |
| **Scenario Groups pane** | Displays each Vuser group and it's current status. For more information. see "Scenario Groups Pane" on page 1110. |
| **Scenario Status pane** | Displays a synopsis of the running scenario. For more information, see "Scenario Status Pane" on page 1111. |

## Run/Stop Vusers Dialog Box

You use this dialog box to add Vusers to a running scenario and to stop Vusers that are running.

The actions you take on this dialog apply only to the current run. They are not saved with the scenario.

| | |
|---|---|
| **To access** | Run tab > Scenario Groups pane > |
| **Important information** | • The dialog box differs depending on which mode you are working in.<br>  • **Vuser group mode.** You specify the number of new Vusers to be added to each Vuser group, as well as the load generators on which these additional Vusers will run.<br>  • **Percentage mode.** You specify the percentage of the Vusers you enter in the **Distribute** box to be added to each script and the load generators on which these additional Vusers will run.<br>  For example, if you enter 12 in the **Distribute** box and there are three scripts with percentages 50%, 25%, 25%, 6 Vusers will be added to the %50 script and 3 to each of the 25% scripts. If you only want to add Vusers to one script, set the percentages to 100%, 0%, and 0%.<br>  The distribution depends only on the percentages, not depend whether the script is checked or not. If three scripts are each set to 33%, but only one is checked, when you add Vusers, they will be distributed 33% per script. Unchecking a script after distributing Vusers does not remove the Vusers that have been distributed.<br>• When you add Vusers to a running scenario or Vuser group, the current scheduler settings are automatically applied to all new Vusers. For example, if the scenario or Vuser group has a set duration of five minutes, all Vusers that are subsequently added run only for the remaining part of that time period.<br>Vusers that are added to a scenario or Vuser group that has finished running are not affected by schedule settings and run according to the scenario runtime settings. |
| **Relevant tasks** | • "How to Run a Scenario" on page 1094<br>• "Use-Case Scenario - Initialize/Run Additional Vusers or Stop Running Vusers" on page 1096 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| | Distributes the added Vusers to their designated load generators so that they are ready to execute their scripts. The Controller first initializes the Vusers in your scenario that have not yet run and then adds additional Vusers, as required, to reach the defined quantity. |
| | • **Run Initialized.** Runs the Vusers in the scenario that have already been initialized.<br>**Note:** You cannot run more Vusers than are currently initialized using this option.<br>• **Run New.** Runs the number of Vusers you specified. The Controller first runs the Vusers in your scenario that have not yet been run and then adds additional Vusers, as required, to reach the defined quantity. |
| | Stops the Vusers that are running. The Controller stops the Vusers according to the settings you defined in the runtime settings tab. For more information, see "Options > Runtime Settings Tab" on page 1082. |
| **<check box>** | Selects the Vuser groups or scripts that **Init**, **Run**, and **Stop** apply to. |
| **%** (Percentage mode) | Enter the percentage of Vusers to be distributed to each Vuser script. |
| **#** | Indicates the number of Vusers distributed to each script. |
| **Distribute x Vusers among the checked scripts** (Percentage mode) | Enter the number of Vusers to be distributed. The Vusers will be distributed according to the values you entered in the percentage (%) column. |
| **Load Generators** | The load generators assigned to the Vuser group/script.<br>If you select multiple load generators for a group/script, the Vusers assigned to the Vuser group/script are distributed evenly among the load generators.<br>**Default value (in Percentage mode):** All Load Generators<br>**Note:** To add a load generator to the list, select **Add** from the list. For more details see "Add New Load Generator/Load Generator Information Dialog Box" on page 1007. |

# Scenario Groups Pane

This pane enables you to monitor the actions of all the Vusers and Vuser groups in the scenario.

| | |
|---|---|
| **To access** | **Run** tab |
| **Relevant tasks** | "How to Run a Scenario" on page 1094 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Right-click menu>** | Additional actions available only via the right-click menu:<br>• **Reset IDs.** Resets the IDs of the Vusers in the group.<br>• **Run Vusers Until Complete.** Runs the selected Vusers until complete. If you run Vusers in the **Down** or **Error** state, the Controller initializes and then runs the Vusers.<br>• **Run One Vuser Until Complete.** Instructs the Controller to run a randomly selected Vuser in the Vuser group, until it completes running. The Vuser Log opens, displaying runtime information about the Vuser. For more information, see "Vuser Script Log" on page 1112.<br>• **Pause Vusers.** Temporarily pauses running the Vusers group. The status of the Vuser group changes from **Running** to **Paused**. Note that pausing a Vuser group affects its transaction response time.<br>• **Enable.** Enables the Vuser group to participate in the scenario.<br>• **Disable.** Disables the Vuser group so that it no longer participates in the scenario.<br>• **Show Vuser Log.** Opens a script log containing runtime information for each Vuser in the group. The Vuser script log is refreshed, by default, every 1000 milliseconds.<br>• **Hide Vuser Log.** Closes the Vuser script logs.<br>• **Sort By Name.** Sorts the groups alphabetically, by name. |
| **Done/Failed** | The number of Vusers that have finished running and the script failed. |
| **Done/Passed** | The number of Vusers that have finished running and the script passed. |
| **Down** | The number of Vusers that have stopped running. |
| **Error** | The number of Vusers that encountered problems. Check the Status field on the Vuser dialog box or the output window for a complete explanation of the error. |
| **Exiting** | The number of Vusers that have finished running or have been stopped, and are now exiting. |

| | |
|---|---|
| **Gradual Exiting** | The number of Vusers that are completing their iterations or actions before exiting (as defined in **Tools > Options > Runtime Settings**). |
| **Initializing** | The number of Vusers that are being initialized on the remote machine. |
| **Pending** | The number of Vusers that are ready to be initialized and are waiting for an available load generator, or are transferring files to the load generator. |
| **Ready** | The number of Vusers that have already performed the init section of the script and are ready to run. |
| **Rendezvous** | The number of Vusers that have arrived at the rendezvous and are waiting to be released by the Controller. |
| **Running** | The number of Vusers that are running, and the Vuser script is being executed on a load generator. |
| **Stopped** | The Vuser stopped when the Stop command was invoked. |

## Scenario Status Pane

This pane displays a synopsis of the running scenario.

| | |
|---|---|
| **To access:** | **Run** tab |
| **Important Information** | To detach the Scenario Status pane from the Run tab, click the detach pane button in the upper right corner. |
| **Relevant tasks** | "How to Run a Scenario" on page 1094 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Elapsed Time** | Indicates how much time has elapsed since the scenario started running. |
| **Errors** | Indicates the number of Vusers with errors.<br>To display the errors, click the **Show Snapshot** button to display the Output Window. For more information, see "Output Window" on page 1102. |
| **Hits/Second** | Indicates how many hits (HTTP requests) there have been to the Web site being tested per second that each Vuser has been running. |
| **Passed/Failed Transactions** | Indicates how many transactions have been executed successfully or unsuccessfully. For more information, see "Transactions Dialog Box" on the next page. |

| | |
|---|---|
| **RunningVusers** | Indicates the number of Vusers that are currently running. |
| **Scenario Status** | Indicates whether the scenario is **Running** or **Down**. |

## Transactions Dialog Box

This dialog box indicates how many transactions have been executed successfully or unsuccessfully.

Providing Authentication

| | |
|---|---|
| **To access** | **Run** tab > **Scenario Status** pane. Click the **Passed/Failed Transactions** > **Show Snapshot** button. |
| **Important information** | VuGen automatically defines each **Init**, **Action**, and **End** unit as a transaction. In addition, you can insert a static transaction in your script using the Start Transaction and End Transaction functions. For details, see "Insert Transactions" on page 339. |
| **Relevant tasks** | "How to Run a Scenario" on page 1094 |
| **See also** | "Scenario Status Pane" on the previous page |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Failed** | The number of times the transaction failed. |
| **Name** | The names of the individual transactions in a script. |
| **Passed** | The number of times the transaction passed. |
| **Stopped** | The number of times the transaction stopped. |
| **TPS** | The number of times per second the transaction has run. |

## Vuser Script Log

This page enables you to view runtime information about each running Vuser.

| | |
|---|---|
| **To access** | Manual scenario > **Run** tab > **Scenario Groups** pane > **Vusers** . In the Vusers dialog box select the Vuser whose log you want to view and click **Show Vuser Log** . |

| | |
|---|---|
| **Important information** | • If you disabled the logging feature in the runtime settings **Log** node, the Vuser script log will contain output only if your script contains the **lr_output_message** or **lr_message** function.<br>• If you selected the **Send messages only when an error occurs** option in the **Log** node, the Vuser script log will contain output only if there are script errors. |
| **Relevant tasks** | "How to Run a Scenario" on page 1094 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| <br> | **Show Text/Tree View.** Displays the runtime information in text/tree format. To revert to the previous view, click the button again. |
| | **Display**. Displays a snapshot of the Web page where an error occurred, when the error is highlighted in the Vuser log.<br><br>**Note:**<br>• To view a snapshot of the Web page where an error occurred, you must select the **Activate snapshot on error** option in the **General** node of the runtime settings dialog box before running the scenario.<br>• Snapshots on errors is supported for TruClient. |
| | **Find Text.** Enables you to search for text in the Vuser log. |
| <br> | **Expand/Collapse Node.** Expands the node so that you can view additional runtime information details about the Vuser. To revert to the collapsed tree view, click the button again. |

| | |
|---|---|
| **<message icons>** | The following icons may can appear in the script log:<br>• **Action.** Displays the name and description of an action.<br>• **End iteration.** Indicates the end of an iteration.<br>• **Errors.** Indicates that the Vuser encountered a problem, but test execution continued. Displays the error code and a description of the error.<br>• **Notifications.** Provides action information.<br>• **Start/End Transaction.** Indicates the start or end of a transaction.<br>• **Start iteration.** Indicates the start of an iteration.<br>• **Start User Script.** Indicates the start of the Vuser script. |
| **<Right-click options>** | • **Copy.** Enables you to copy selected text from the Vuser log.<br>• **Copy path from status bar.** Enables you to copy the path of the Vuser log. |
| **Refresh (every 1000 milliseconds)** | When selected, instructs LoadRunner to refresh the runtime information displayed every 1000 milliseconds.<br>**Note:** For information on how to change the default refresh settings, see "Options > Output Tab" on page 1080. |

# Running Scenarios from the Command Line

You can set up scenario run automation by invoking Controller from the command line without the need to manually define run settings using Controller's UI.

When invoked, Controller checks all of the received arguments and sets its start-up environment accordingly.

For example, you can instruct Controller to connect to the Load Generator, start up, save results to a folder other than the folder defined in the scenario, and invoke Analysis upon completion of the scenario run.

The Command Line Interface (CLI) tool enables you to run a scenario from the command line using an XML input file, or a LoadRunner scenario (.lrs) file.

Alternatively, you can run a LoadRunner scenario (.lrs) from the command line using the **wlrun** command.

## Running Scenarios using the Command Line Interface (CLI) Tool

This topic describes how to use the CLI tool for scenario run automation.

The CLI tool enables you to run load test scenarios from the command prompt using a LoadRunner scenario (.lrs) file or an XML input file, .

You construct the XML input prior to running scenarios, allowing for fully DevOps-oriented flows including:

- Pulling scripts from any source control
- Dynamically allocating Dockerized load generators
- Running scenarios from any CI tool

## Invoking the CLI Tool

1. **Prerequisite:** Make sure the user name and password of the admin user that will be used to run Controller are defined in **<LoadRunner root>\bin\CLIControllerApp.exe.config** in the **<appSettings>** tag.
2. Run **CLIControllerApp.exe**, located in the **<LoadRunner root>\bin**, appending the arguments as described in "Command line format" below.

## Command line format

Use the following format for the command line arguments:

```
-{Run Type} -{File Path} -{Post Run Command} -{Result Name} -{Result Path}
```

| Arguments | Description |
|---|---|
| Run Type<br><br>(Mandatory) | • **InputFile**. Use this when the input data for the test is provided in *.xml format. The XML input file must contain the structure described in "XML Input File" on the next page.<br>• **TestPath**. Use this when the test runs with data from an existing *.lrs file |
| File Path<br><br>(Mandatory) | A valid Windows path leading to a file.<br>• When Run Type is **Inputfile**, this file is an *.xml file, in the CLI format. See "XML Input File" on the next page.<br>• When Run Type is **TestPath**, this file must be an *.lrs file. |
| Post Run Command<br><br>(Mandatory) | Determines what happens when the run finishes.<br>• **Run**. Runs the scenario only.<br>• **Collate**. Runs the scenario. When the scenario finishes running, the results are collated.<br>• **CollateAndAnalyze.** Runs the scenario. When the scenario finishes running, the results are collated and analyzed. |

| Arguments | Description |
|---|---|
| Result Name<br>(optional) | Denotes where the results of the test should be stored.<br>Format: **ResultName** |
| Result Path<br>(optional) | The path to the location of the stored test run results.<br>This argument must always follow the **-ResultName** argument and must contain a valid folder path.<br>Format: **ResultPath** |

## Examples

- Running a test using an XML input file:

```
<LoadRunner root>\bin\CLIControllerApp.exe -InputFile C:\CLI.xml -
CollateAndAnalyze -ResultName C:\ControllerResultPath\Res_01
```

- Running a test using an existing *.lrs file:

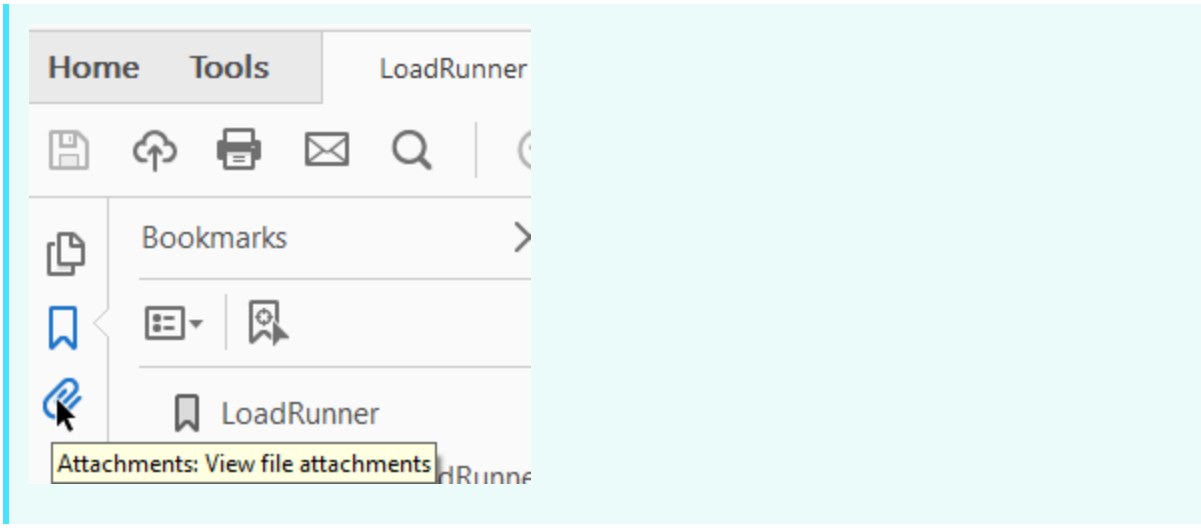
```
<LoadRunner root>\bin\CLIControllerApp.exe -TestPath C:\testName.lrs -
CollateAndAnalyze -ResultName C:\ControllerResultPath\Res
```

## XML Input File

The XML input file contains a **CLIRun** element where you provide all the information about the groups, hosts and scheduler included in the scenario.

You can see a template of the XML input file in the **CLI Input File Template.xml** file, attached to this document.

> **Tip:** To open the attachment, click the **Attachments** button in Acrobat Reader's navigation panel:




The **CLIRun** element contains 3 sections:

- **Groups.** A list of all the groups that will be used in the test
- **Hosts.** A list of all the hosts that will be available for the Groups
- **Scheduler.** The scheduler details.

| Element | Description |
|---|---|
| **Group** | A group in the list of groups.<br>• **Name.** The name of the group.<br>• **Vusers.** The number of vusers in the group. Positive integer value only.<br>• **Script.** The script used by the group. See Script.<br>• **AssignedHosts.** Contains at least one **HostName** element<br>• **HostName.** Identical to the name of a host defined in the Hosts section |
| **Script** | A script used by a group.<br>• **Name.**The name of the script.<br>• **ID.** The unique identifier for the script. Positive integer values only.<br>• **Path.** The valid path to the *.zip file containing the script files.<br>• **ProtocolType.** The protocol type of the script. Can be: JMETER, GUI, WEB, DB, TULIP, TLS, QTEWB, WINRUNNER, VB-WEB, TESTMODULE, TRUWEB<br>• For JMeter protocol scripts only (mandatory):<br>  • **JMeterPath.** A valid path to the JMETER of the script<br>  • **JMeterAdditionalPropertiesFilePath.** The valid path to the file containing additional properties for JMETER |

| Element | Description |
|---|---|
| **Host** | A host in the lists of hosts, and used by a group.<br>• **Name.** The name of the host.<br>• **ID.** The host ID. Positive integer value only.<br>• **Installation.** UNIX, WINDOWS<br>• **HostPurposes.** Contains at lease one **HostPurpose** tag.<br>• **HostPurpose.** The purpose of the host: Controller, Data Processor, Load Generator.<br>• **State.** Operational, NonOperational, Unavailable<br>• **Location.** Location of the host.<br>• **IsOverFirewall.** Boolean: TRUE, FALSE. |
| **Scheduler** | Determines the duration of the scenario run.<br>• **Name.** The scheduler name.<br>• **Data.** Contains the **Duration** tag.<br>• **Duration.** The duration (in seconds) of the scenario run. Positive integer value only. |

## XML Input File Samples

- "JMeter sample" below
- "WebHTTP Sample" on the next page

## JMeter sample

```
<CliRun>
	<Groups>
		<Group>
			<Name>JMeter1</Name>
			<Vusers>1</Vusers>
			<Script>
				<ID>1</ID>
				<Name>simple_test</Name>
				<Path>C:\Protocols\JmeterSimple_test\simple_test.jmx</Path>
				<ProtocolType>JMeter</ProtocolType>
				<JMeterPath>C:\Protocols\JmeterSimple_test\</JMeterPath>
				<JMeterAdditionalPropertiesFilePath>C:\Protocols\JmeterSimple_test\simple_test.properties</JMeterAdditionalPropertiesFilePath>
			</Script>
			<AssignedHosts>
				<HostName>localhost</HostName>
			</AssignedHosts>
		</Group>
	</Groups>
	<Hosts>
		<Host>
			<Name>WIN-2AKM2PQEN8J</Name>
			<Installation>Windows</Installation>
			<HostPurposes>
				<HostPurpose>Load Generator</HostPurpose
				<HostPurpose>Controller</HostPurpose>
			</HostPurposes>
			<State>Ready</State>
			<Location>Local</Location>
			<ID>1</ID>
			<IsOverFirewall>false</IsOverFirewall>
		</Host>
		<Host>
			<Name>localhost</Name>
			<Installation>Windows</Installation>
			<HostPurposes>
				<HostPurpose>Load Generator</HostPurpose>
			</HostPurposes>
			<State>Down</State>
			<Location>Local</Location>
			<ID>2</ID>
`			<IsOverFirewall>false</IsOverFirewall>
		</Host>
	</Hosts>
	<Scheduler>
		<Name>Scheduler 1</Name>
		<Data>
			<Duration>120</Duration>
		</Data>
	</Scheduler>
</CliRun>
```

### WebHTTP Sample

```
<CliRun>
	<Groups>
		<Group>
			<Name>a1</Name>
			<Vusers>2</Vusers>
			<Script>
				<ID>1</ID>
				<Name>kali</Name>
				<Path>C:\Users\Administrator\Documents\VuGen\Scripts\kali</Path>
			</Script>
			<ProtocolType>QTWeb</ProtocolType>
			<AssignedHosts>
				<HostName>localhost</HostName>
			</AssignedHosts>
		</Group>
	</Groups>
	<Hosts>
		<Host>
			<Name>WIN-2AKM2PQEN8J</Name>
			<Installation>Windows</Installation>
			<HostPurposes>
				<HostPurpose>Load Generator</HostPurpose>
				<HostPurpose>Controller</HostPurpose>
			</HostPurposes>
			<State>Ready</State>
			<Location>Local</Location>
			<ID>1</ID>
			<IsOverFirewall>false</IsOverFirewall>
		</Host>
		<Host>
			<Name>localhost</Name>
			<Installation>Windows</Installation>
			<HostPurposes>
				<HostPurpose>Load Generator</HostPurpose>
			</HostPurposes>
			<State>Down</State>
			<Location>Local</Location>
			<ID>2</ID>
			<IsOverFirewall>false</IsOverFirewall>
		</Host>
	</Hosts>
	<Scheduler>
		<Name>Scheduler 1</Name>
		<Data>
			<Duration>120</Duration>
		</Data>
	</Scheduler>
</CliRun>
```

## Notes about running tests using the CLI tool

When you invoke Controller from the command line, the following applies:

- Only one instance of Controller can run at a time.
- The arguments are case sensitive.

- Controller will always overwrite results.
- Controller launched through the command line behaves normally except when using the **-Run** option. Using the **-Run** option, dialogs and message boxes that usually open and require the user to close them in a usual launch, do not open in a command line launch.
- Controller's settings are loaded from **wlrun7.ini**, located in the **<LR installation>\config** folder (both)

### See also:

- "Running Scenarios Using the wlrun Command" below
- "Running Scenarios in Controller" on page 1093

## Running Scenarios Using the wlrun Command

You can run scenarios from the command line using the **wlrun** command.

### Invoking the Controller using the wlrun command

You can invoke Controller from the command line by typing `wlrun` in the command line, followed by the desired arguments.

> **Note:** Before invoking Controller using wlrun, you should be familiar with the rules relating to command line arguments. For details, see "Notes about running tests using the wlrun command" on the next page.

For example, to run the **Scenario.lrs** scenario, enter the following command:

```
wlrun -TestPath C:\LoadRunner\scenario\Scenario.lrs -Run
```

### wlrun runtime arguments

- "General arguments" below
- "Application Lifecycle Management arguments" on the next page

### General arguments

These arguments specify the runtime related scenario settings. For more information on scenario settings, see "Before Running Your Scenario" on page 1085.

| | |
|---|---|
| **TestPath** | Path to the scenario, for example,<br>`C:\LoadRunner\scenario\Scenario.lrs`<br>This argument can also be used for a scenario residing in an Application Lifecycle Management database. For example,<br>`"[TD]\Subject\LoadRunner\Scenario1"`<br>If the path includes blank spaces, use quotation marks. |

| | |
|---|---|
| **Run** | Runs the scenario, dumps all output messages into **res_dir\output.txt**, and closes Controller |
| **InvokeAnalysis** | Instructs LoadRunner to invoke Analysis upon scenario termination. If this argument is not specified, LoadRunner uses the scenario default setting. |
| **SilentMode** | Suppresses the pop-up message which normally appears in the UI when Analysis is automatically invoked upon scenario termination. |
| **ResultName** | Full results path. For example, "`C:\Temp\Res_01`" |
| **ResultCleanName** | Results name. For example, "`Res_01`" |
| **ResultLocation** | Results folder. For example, "`C:\Temp`" |

## Application Lifecycle Management arguments

These arguments define the LoadRunner integration with Application Lifecycle Management. For more information about the LoadRunner Application Lifecycle Management integration, see "Managing Scenarios Using Application Lifecycle Management" on page 1126.

| | |
|---|---|
| **ConnectToQC** | Specifies whether the Controller should connect to ALM on startup (**0/1** or **ON/OFF**) |
| **QCServer** | Application Lifecycle Management server name. Must be a machine where Application Lifecycle Management is installed |
| **QCDB** | Application Lifecycle Management database name. Use the format: **<Domain name>.<Project name>**. |
| **UserName** | User name for connecting to Application Lifecycle Management |
| **Password** | Password corresponding to the user name |
| **TestPath** | Path to scenario in Application Lifecycle Management database. For example, `"[TD]\Subject\LoadRunner\Scenario1"`<br>If path includes blank spaces, use quotation marks. |
| **TestId** | Test ID (used by ALM only) |
| **ResultCleanName** | For use with **ResultCycle** only. Example: "`Res1`" |
| **ResultCycle** | Application Lifecycle Management cycle. For example, "`LR_60_SP1_247`"<br>**Note:** The **ResultCycle** and **ResultCleanName**arguments are required if you wish to store the results within the Application Lifecycle Management database. |

## Notes about running tests using the wlrun command

When you invoke Controller from the command line using **wlrun**, consider the following:

- Only one instance of Controller can run at a time.
- The arguments are case sensitive.
- If Controller is invoked with no arguments in the command line, Controller uses its default settings.
- Controller will always overwrite results.
- Controller will automatically terminate upon scenario termination and results will be collected. If you don't want Controller to automatically terminate upon scenario termination, add the flag -**DontClose** to the command line.
- Controller launched through the command line behaves normally except when using the **-Run** option. Using the **-Run** option, dialogs and message boxes that usually open and require the user to close them in a usual launch, do not open in a command line launch.
- When Analysis is started from the command line, use the **-SilentMode** argument to suppress the pop-up message which normally appears in the UI.
- Controller's settings are loaded from **wlrun7.ini**, located in the ***<LoadRunner installation>*\config** folder.

### See also:

- "Running Scenarios using the Command Line Interface (CLI) Tool" on page 1114
- "Running Scenarios in Controller" on page 1093

# Using Unified Functional Testing Tests in LoadRunner

## Using Unified Functional Testing Tests in LoadRunner - Overview

Functional Testing software, (Unified Functional Testing) (UFT, formally known as QuickTest Professional) enables you to create complex tests that examine the full spectrum of your application's functionality.

LoadRunner can integrate UFT tests into a load testing scenario in the form of GUI Vuser scripts. These tests, that have already been designed and debugged in UFT can be used as the basis of your load test.

The main uses of running UFT tests in LoadRunner are:

- To check how your application's functionality is affected by heavy load
- To measure the response time that a typical user experiences on the client side while your application is under load (end-to-end response time)

For example, you can add UFT tests to specific points in a LoadRunner scenario to confirm that the application's functionality is not affected by the extra load at those sensitive points.

Another advantage of using a GUI Vuser script as part of your LoadRunner scenario is that the GUI Vuser script runs on your screen during the scenario, enabling you to watch the actual steps executed by the Vuser in real time.

# About GUI Vuser Scripts

GUI Vusers enable you to measure and monitor end-to-end user response times while your client/server system is under load. A GUI Vuser emulates the complete environment of a human user.

For example, a human user sits at a machine, operates applications using the keyboard and the mouse, and reads information on the machine's monitor. Similarly, a GUI Vuser runs on its own machine and operates applications. A GUI Vuser can be programmed to read and act on information that appears on its machine's display.

Suppose that you have a bank server that services many automatic teller machines (ATMs). You could create a GUI Vuser script that:

- Opens the ATM application
- Enters an account number
- Enters the amount of cash to be withdrawn
- Withdraws cash from the account
- Checks the balance of the account
- Closes the ATM application
- Repeats the process

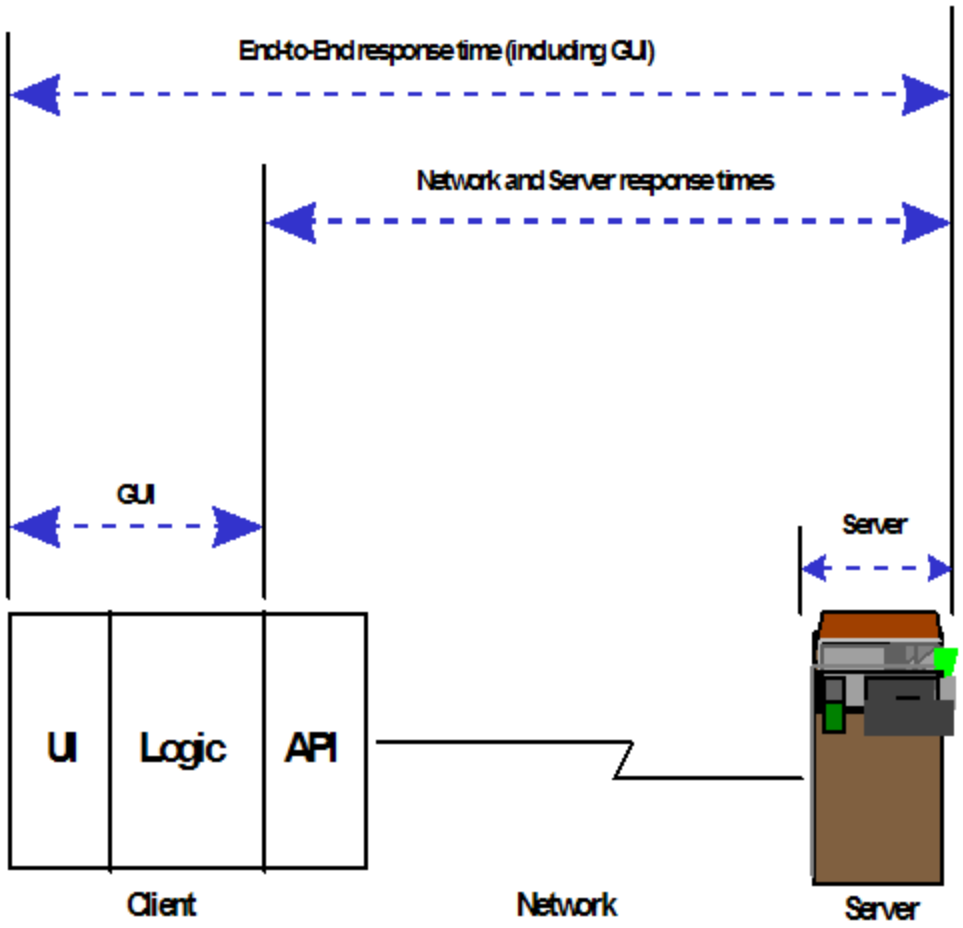
The actions of each GUI Vuser are described in a GUI Vuser script. You use Unified Functional Testing (UFT, formally known as QuickTest Professional) to create GUI Vuser scripts.

You monitor and manage GUI Vusers using the LoadRunner Controller. For instance, you can use Controller to run, pause, or view Vusers, and to monitor scenario status.

> **Note:** You cannot use VuGen to run a GUI Vuser script. You use Controller to run a GUI Vuser script as part of a scenario; you use UFT to run a GUI Vuser script in standalone mode.

## Understanding GUI Vuser Technology

GUI Vusers measure real end-to-end response times. End-to-end response times represent the total time that a user waits for a response after submitting a request. End-to-end response times include GUI response times as well as network and server response times.




## Guidelines for Using QuickTest or Unified Functional Testing Tests in LoadRunner

When creating test scripts in Unified Functional Testing (UFT) that are going to be used as GUI Vuser scripts in a LoadRunner testing scenario, you need to follow certain guidelines to ensure smooth integration of the script. For detailed explanations about creating tests in UFT, see the UFT documentation.

### Limitations

UFT offers several features that are designed specifically for integration with LoadRunner. Some UFT features, however, may not be available when they are integrated with LoadRunner. For more information about specific limitations, see the UFT readme.

### Including Transactions

To measure the performance of the server, you define **transactions**. A transaction represents an action or a set of actions that you are interested in measuring. You define transactions within your Vuser script

by enclosing the appropriate sections of the script with **start** and **end** transaction statements.

For example, you can define a transaction that measures the time it takes for the server to process a request to view the balance of an account and for the information to be displayed at the ATM.

> **Note:** LoadRunner only provides performance information for data that is included within a transaction. Therefore, your UFT test must include transactions to be used by LoadRunner.

For more information about using transactions in Unified Functional Testing, see the UFT documentation.

### Adding Statements

You can use the **Services** object and its associated methods to insert statements that are specifically relevant to performance testing. These include **Abort**, **GetEnvironmentAttribute**, **LogMessage**, **SetTransactionStatus**, **ThinkTime**, **UserDataPoint**, **StartTransaction** and **EndTransaction**. For more information on these methods, see the UFT documentation.

### Designing Tests for LoadRunner

Consider the following design guidelines when designing tests for use with LoadRunner:

- The UFT tests you use with LoadRunner should be simple tests, designed to pinpoint specific operations.
- LoadRunner cannot run nested action iterations.
- Do not include references to external actions or other external resources, such as an external Data Table file, environment variable file, shared object repositories, and so forth.
- Include transactions in your UFT test since LoadRunner only provides performance information for data that is included within a transaction.

## How to Add a Unified Functional Testing Test to a Load Test Scenario

This task describes how to integrate a Unified Functional Testing (UFT, formally known as QuickTest Professional) test into LoadRunner.

1. Navigate to the folder containing the test.
   - For a new scenario, click **Browse** in the New Scenario dialog box.
   - When adding the test to an existing scenario, click **Browse** in the Add Group/Add Script dialog box. The Open Test dialog box opens.
2. In the **Files of Type** box select **QuickTest Tests** or **Unified Functional Testing Tests**.
3. Navigate to the appropriate test and add it to your scenario.

# Managing Scenarios Using Application Lifecycle Management

## Managing Scenarios Using Application Lifecycle Management - Overview

Controller works together with Application Lifecycle Management (ALM), Micro Focus's Web-based test management tool. ALM provides an efficient method for storing and retrieving Vuser scripts, scenarios, and results. You can store scenarios in an ALM project and organize them into unique groups.

In order for Controller to access an ALM project, you must connect it to the Web server on which Application Lifecycle Management is installed. You can connect to either a local or remote Web server.

> **Note:** Integration with ALM requires a full installation of ALM. It is not supported for the Community and Express Editions of ALM.

For more information on working with Application Lifecycle Management, see the *Application Lifecycle Management User's Guide*.

## How to Work with Scenarios in ALM Projects

The following steps describe the workflow of how to work with scenarios saved in an Application Lifecycle Management project.

1. Connect to ALM

   Open a connection to the ALM server and project that contains the scenario. For task details, see "How to Connect to ALM" below.

2. Open the scenario

   Select **File > Open**. The Open Scenario from ALM Project dialog box opens. Select the name and location of the scenario to open.

3. Save the scenario

   Select **File > Save as**. If the scenario is in a project that uses version control and is not checked out, the scenario is only saved as a temporary file on your local machine.

## How to Connect to ALM

To store and retrieve scenarios from ALM, you need to connect to an ALM project. You can connect or disconnect from an ALM project at any time during the testing process.

You can connect to one version of ALM from Controller and a different version from your browser. For more information, see the **Important Information** section in "ALM Connection Dialog Box [Controller]" on the next page.

To connect to Application Lifecycle Management:

1. Select **Tools > ALM Connection**. The ALM Connection dialog box opens.
2. Enter the required information in the ALM Connection dialog box, as described in "ALM Connection Dialog Box [Controller]" on the next page.
3. To disconnect from ALM, click **Disconnect**.

**Note:**

- If you authenticated through CAC mode and disconnected from the ALM server, you need to restart the Controller before reconnecting in CAC mode.
- If you have a slow connection to ALM, this may be because the Controller machine does not have access to the WAN, and so cannot verify the digital signature of the relevant files.

  **Resolution:** Download and import the certificates from https://support.globalsign.com/customer/portal/articles/1426602-globalsign-root-certificates using certmgr MMC until the issue is resolved (you might not need to import all the certificates). This information is also available in a knowledge base article.

## How to Save Scenarios to ALM Projects

The following steps describe how to save a scenario to an ALM project.

1. Open/create the scenario

   Create or open the desired scenario.
2. Connect to Application Lifecycle Management

   Open a connection to the ALM server and project in which you want to store the scenario. For task details, see "How to Connect to ALM" on the previous page.
3. Define a test set

   Define an Application Lifecycle Management test set where to save results as follows:
   a. Select **Results > Results Settings**. The Set Results Directory dialog box opens.
   b. Click **ALM**.
   c. Enter the required information in the Set Results Directory dialog box. For user interface details, see "Set Results Directory Dialog Box" on page 1088.
   d. Click **OK**.
4. Save the scenario to ALM

   Select **File > Save as** and specify the location.

# Add Vuser Scripts from an Application Lifecycle Management Project

The following steps describe how to add Vuser scripts from an Application Lifecycle Management (ALM) project to the Controller's script list.

## Add a Vuser script to a manual scenario

1. Open a connection to the ALM server and project where the scripts are located. For task details, see "How to Connect to ALM" on page 1126.
2. In the Scenario Groups pane, click the **Add Group** button .
3. In the Add Group dialog box, click **Browse**. The Open Test from ALM Project dialog box opens.
4. Select the script and click **OK**. The Script Path field displays [TD], the full subject path, and the script name.

   For example:

   ```
   [TD]\Subject\System\test_alm
   ```
5. Click **OK**. The script is displayed in the Scenario Groups pane.

## Add a Vuser script to a goal-oriented scenario

1. Open a connection to the ALM server and project where the scripts are located. For task details, see "How to Connect to ALM" on page 1126.
2. On the Scenario Scripts pane toolbar, click the **Add Script** button. The Add Script dialog box opens. For details, see "Add Script Dialog Box" on page 960.
3. Click **Browse**. The Open Test from ALM Project dialog box opens and displays the test plan tree.
4. Select the script and click **OK**. The Script Path field displays [TD], the full subject path, and the script name.

   For example:

   ```
   [TD]\Subject\System\test_alm
   ```
5. Click **OK** to close the Add Script dialog box. The script appears in the Script Path column in the Scenario Scripts pane.

# ALM Connection Dialog Box [Controller]

This dialog box enables you to connect to an ALM project from within Controller.

| **To access** | **Tools > ALM Connection** |
|---|---|

| | |
|---|---|
| **Important information** | You can connect to one version of ALM from Controller and a different version of ALM from your browser.<br><br>You can only connect to different versions of ALM if one of the versions is ALM 11.00 or higher.<br><br>**Note:** Before you connect to results stored on ALM through this dialog box, it is recommended that you first connect to the ALM server through your browser. This automatically downloads the ALM client files to your computer. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Step 1: Connect to Server** | • **Server URL.** The URL of the server on which ALM is installed.<br>• **Reconnect to server on startup.** Automatically reconnect to the server every time you start Controller.<br>• / . Connects to the server specified in the **Server URL** box. Only one button is visible at a time, depending on your connection status. |
| **Step 2: Authenticate User Information** | **Note:** Step 2 is only visible after you successfully connect to a server.<br>• **User Name.** Your ALM user name.<br>• **Password.** Your ALM password.<br>• **Authenticate on startup.** Authenticates your user information automatically, the next time you open the application. This option is available only if you selected **Reconnect to server on startup** above.<br>• . Authenticates your user information against the ALM server.<br>After your user information has been authenticated, the fields in the Authenticate user information area are displayed in read-only format. The Authenticate button changes to .<br>You can log in to the same ALM server using a different user name by clicking **Change User**, entering a new user name and password, and then clicking **Authenticate** again. |

| UI Element | Description |
|---|---|
| **Step 3: Login to Project** | • **Domain.** The domain that contains the ALM project. Only those domains containing projects to which you have permission to connect to are displayed.<br>• **Project.** Enter the ALM project name or select a project from the list. Only those projects that you have permission to connect to are displayed.<br>• **Login to project on startup.** This option is enabled only when the **Authenticate on startup** check box is selected.<br>• / . Logs into and out of the ALM project. |

# Continuous Integration with Jenkins

As more software companies utilize continuous integration practices, you may also need to integrate load tests into your testing process. This integration helps developers insure that new builds did not introduce regressions.

The **Application Automation Tools** plugin for the Jenkins continuous integration server provides a mechanism for executing LoadRunner Controller scenarios as part of a build script. This plugin allows you to trigger an performance test as a build step and present the results in the Jenkin's user interface.

You can only integrate scenarios which have service level agreements (SLAs). This allows you to quickly determine whether the test passed or failed and if performance was affected.

To begin the integration, you must first install the **Application Automation Tools** plugin. For information about installing plugins, refer to the Jenkins documentation.

Click the following link to open the **Application Automation Tools** plugin page:https://wiki.jenkins.io/display/JENKINS/Micro+Focus+Application+Automation+Tools

For a blog post describing load testing with continuous integration, click here.

> **Note:** The Jenkins plugin requires an administrator account.

# After the Scenario Run

# Post Scenario Run Procedures - Overview

After the scenario runs you analyze the results using LoadRunner Analysis. If the run results are stored locally on each participating load generator, they need to be gathered into one location so that they can be processed for analysis. This process is known as **data collation**.

For details about collating run results and diagnostics data, see "Collating Run Data" on the next page.

For details about analyzing the scenario run, see the *LoadRunner Analysis User Guide*.

# Collating Run Data

When you run a scenario, by default all the run data is stored locally on each load generator. After scenario execution, the results must be collated—that is, the results from all of the load generators must be gathered and transferred to the results folder–before any analysis data can be generated.

In addition, data from the diagnostics servers must be collated as well.

You can set LoadRunner to collate the run data automatically, as soon as the run is complete. Alternatively, you can collate the run data manually after the run. This way, you can save and close a scenario and collate the data after reopening the scenario in Controller.

The data that is collated include the result, diagnostics, and log files. After LoadRunner has successfully collated the data, these files are deleted from the load generators from which they were gathered.

> **Note:** In Expert mode, you can disable the collation of the log file collation (see "How to Configure Scenario Options" on page 1072).

For details on how to collate run data, see "How to Collate Scenario Run Results" below.

# How to Collate Scenario Run Results

This task describes how to collate results after a scenario run.

> **Note:**
>
> - Data collation includes result, diagnostics, and log files. If you are working in Expert mode, you can disable the collation of the log files. Before collating the results, select **Tools > Options > General tab > Do not collate log files.**
> - You can set a command to run when collation is complete. Select **Tools > Options > Execution** tab, and enter the command in the **Post Collate Command** box.

- Collate results automatically

  Select **Results > Auto Collate Results.**
- Collate results manually

  Select **Results > Collate Results > Collate Results.**
- Stop the collation process

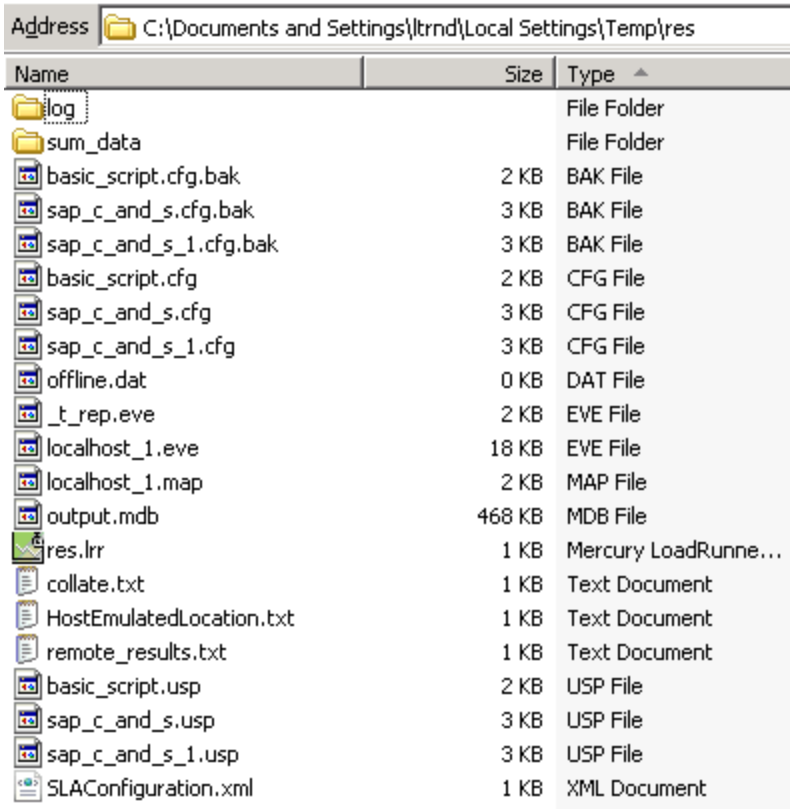
  In the Collate Results dialog box, click **Stop.**
- Resume the collation process

  If you stopped the collation process, to resume select **Results > Collate Results > Continue stopped collation**.
- If collation fails due to a lack of disk space

To re-collate, select **Results > Collate Results > Recollate**. LoadRunner attempts to collate the results again, without compressing the **.eve** file.

# Scenario Run Results

Before you run a scenario, you specify where the run results should be stored. LoadRunner saves all the data it gathers during the run to the specified folder. A typical results folder has the following structure:




The content of the results folder are described in the following table:

| Folder/File | Description |
|---|---|
| **log folder** | Output information generated during replay for each Vuser |
| **sum_data folder** | Graph summary data (**.dat**) files. |
| ***_bd directories** | Diagnostics breakdown information. |
| ***.cfg files** | A listing of the script's runtime settings as defined in VuGen (think time, iterations, log, Web, and so on). The results folder contains a .cfg file for each script. |
| ***.def files** | Definition files for graphs that describe the online and other custom monitors. |

| | |
|---|---|
| ***.usp files** | Contain the script's run logic, including how the actions sections run. The results folder contains a .usp file for each script. |
| **_t_rep.eve** | Contains Vuser and rendezvous information. |
| **<Controller>.eve** | Contains information from the Controller host. |
| **<Load_Generator>.eve.gzl files** | Information from the load generators in the scenario. These files are zipped and saved to the results folder in .gzl format. |
| **<Load_Generator>.map** | Maps transactions and data points on the load generator to IDs. |
| **<results_name> folder** | The scenario run results. |
| **<results_name>.lrr** | Information about the scenario run, such as the name, duration, scripts included, and so on. |
| **collate.txt** | The file paths of the result files and collation status information. |
| **collateLog.txt** | The status (succeeded, failed) of result, diagnostics, and log file collation from each load generator. |
| **HostEmulatedLocation.txt** | Information about network virtualization such as the locations and mode (per group or load generator). |
| **offline.dat** | Sample monitor information. |
| **output.mdb** | The database created by the Controller. Stores all the output messages reported during the scenario run. |
| **remote_results.txt** | The file paths for the host event files |
| **SLAConfiguration.xml** | SLA definition information for the scenario. |

When you generate analysis graphs and reports, the Analysis engine copies all of the scenario result files (**.eve** and **.lrr**) to a database. After the database is created, Analysis works directly with the database and does not use the result files.

> **Note:** You can also convert the results to JSON using the **LoadRunner Raw Results Exporter** tool. This is useful, for example, when the run results are too large for Analysis to process. For details, see "LoadRunner Raw Results Exporter" below.

## LoadRunner Raw Results Exporter

You can use the **LoadRunner Raw Results Exporter** tool to export scenario results to JSON files or to InfluxDB.

## Launching the tool

To launch the **LoadRunner Raw Results Exporter** enter the following command:

```
LrRawResultsExporter -source_dir <folder_path> [-to_influx] [-target_dir <folder_path> -to_json] [-f]
```

where:

| Option | Description |
|---|---|
| **-source_dir *<folder_path>*** | The location of the source file (.lrr) containing the scenario run results. |
| **-to_influx** | Instructs the Exporter to export the raw data to InfluxDB.<br>The following naming convention is used for the database that is created:<br>**<name of folder>_<timestamp>**, where timestamp is formatted as yyMMddHHmmss |
| **-target_dir *<folder_path>* -to_json** | Instructs the Exporter to export the raw data to JSON files in the specified target folder. |
| **-f** | Forces the execution of the application, automatically overwriting the target directory if it already exists. |

> **Tip:** For help, enter **-h**.

## Notes

When exporting to InfluxDB:

- The InfluxDB server must be up and running.
- **Recommended:** To allow for unlimited values per tag, in your InfluxDB configuration file, set **max-values-per-tag= 0**. For details, see the InfluxDB documentation.

## JSON output

When you export results to JSON, the information is stored as follows:

| | |
|---|---|
| RunInfo.txt | Contains information about the scenario run that generated the processed raw results. |
| offline.txt | Contains the offline monitor data recorded by the monitors set up in the scenario run. |

| | |
|---|---|
| offlineMetadata.txt | Contains additional metadata for the offline monitor datapoints stored in offline.txt. |
| *.eve | As in the original .eve files, contain all the information recorded from the run: Transaction data, Transaction breakdown data, Errors (only script errors) and Web related datapoints. |

### InfluxDB output

When you export results to InfluxDB, the information is stored in the database as follows:

| Table | Description |
|---|---|
| Error | The error events that occurred during the scenario run. |
| Transaction | Complete transaction information. |
| TransactionBd | Transaction breakdown information. |
| RunInfo | Information about the scenario run. |
| VuserInfo | Information about the Vusers that ran during the scenario run. |
| Datapoint | The Web-related datapoints recorded during the run. |
| MonitorDatapoint | Datapoints recorded by offline monitors during the scenario run. |
| MonitorMetadata | Additional metadata for the offline monitor datapoints recorded during the scenario run. |

# Collate Results Dialog Box

This dialog box enables you to view the progress of result collation after a scenario run.

| | |
|---|---|
| **To access** | • **Results > Collate Results > Collate Results**<br>• If **Auto Collate Results** is activated, this dialog box opens automatically when LoadRunner starts collating the run results after a scenario run. |
| **Relevant tasks** | • "How to Collate Scenario Run Results" on page 1131<br>• "How to Prepare a Scenario to Run" on page 1085 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Stop | Stops collating the results. |

| | |
|---|---|
| **Close automatically** | Select to automatically close the Collate Results dialog box after collation is completed. |
| **General Status** | The collation status and file size of the Event, Diagnostics, and Log files.<br>**Note:** File size shown is before compression. |
| **Progress Details** | The status (succeeded, failed) of result, diagnostics, and log file collation from each load generator. This information is stored in the **collateLog.txt** file. |

# Working with Firewalls in LoadRunner

If you need to monitor servers that sit on the other side of a firewall, you need to set up your system properly, including installing and configure a dedicated Monitor-Over-Firewall machine, configure the LoadRunner agent to communicate with the MI Listener (which serves as a router between the LoadRunner agent and the Controller), and then define the server measurements that you want the Monitor-Over-Firewall machine to monitor.

To ensure a secure environment, you must also set up client-server authentication. LoadRunner provides a security solution based on an industry standard SSL/TLS library (OpenSSL), which includes both data encryption and authentication. Using this authentication method helps to protect your data from being exposed to third-parties, and to prevent your LoadRunner components from being used by unauthenticated users.

### What do you want to do?

- Set up LoadRunner over a firewall
- Configure the LoadRunner Agent
- Set up secure communication over a firewall
- Configure Secure Communication with Two-way TLS (SSL) Authentication

### See also:

- Network and Security Manager utility
- Troubleshooting for firewalls

# Set Up Your LoadRunner System Over Firewalls

The MI Listener and Monitor over Firewall setup files are provided in the LoadRunner installation package's **/Additional Components** folder.

Setting up your system to work with firewalls involves the following stages of configuration:

1. Install the Over-Firewall Components.

   To enable over firewall communication, ensure that you have installed the following LoadRunner components:

   - **MI Listener.** Serves as a router between the Controller and the LoadRunner agent. You install the MI Listener component on a dedicated machine. To access the setup file, see "Additional Components" on page 1566.

     > **Note:** You can also use the Controller machine as the MI Listener without the need for a separate installation. When acting as the MI Listener, the Controller machine cannot have Vusers running on it. In this case, the Controller must be a pure Controller and not a Controller + Load Generator.

   - **Monitor Over Firewall component.** Used to monitor the servers that are located over a firewall. You install the Monitors over Firewall component on a dedicated machine. To access the setup file, see "Additional Components" on page 1566.

2. Perform the Initial Configuration of the Over-Firewall system.

   See "Set Up an Over-Firewall Deployment" below.

3. Configure the LoadRunner Agent

   Configure the LoadRunner agent on the monitor-over-firewall machine as well as on each load generator machine that will be running over the firewall, to communicate with the MI Listener

   See "Configure the LoadRunner Agent" on page 1140.

4. Connect the Controller to the Load Generator and MI Listener machines.

   Configure the Controller to recognize the Load Generator and MI Listener machines and then verify the connection between the machines.

   See "Create and Verify the Connection Between Controller and Agent Machines" on page 1142.

5. Define What to Monitor on Your Servers.

   Choose the server measurements that you want your Monitor Over Firewall machine to monitor.

   See "Set Firewall Monitoring Preferences" on page 1143.

6. Check Connectivity.

   After installing and configuring all the necessary components, check that you are able to establish a connection between the LoadRunner agent, MI Listener, and the Controller machine. If you have difficulties, see "Troubleshooting and Limitations for Firewalls" on page 1150.

### See also:

- "Secure communication over Firewalls" on page 1149

## Set Up an Over-Firewall Deployment

This task describes how to configure the Over-Firewall System.

1. Prerequisites.

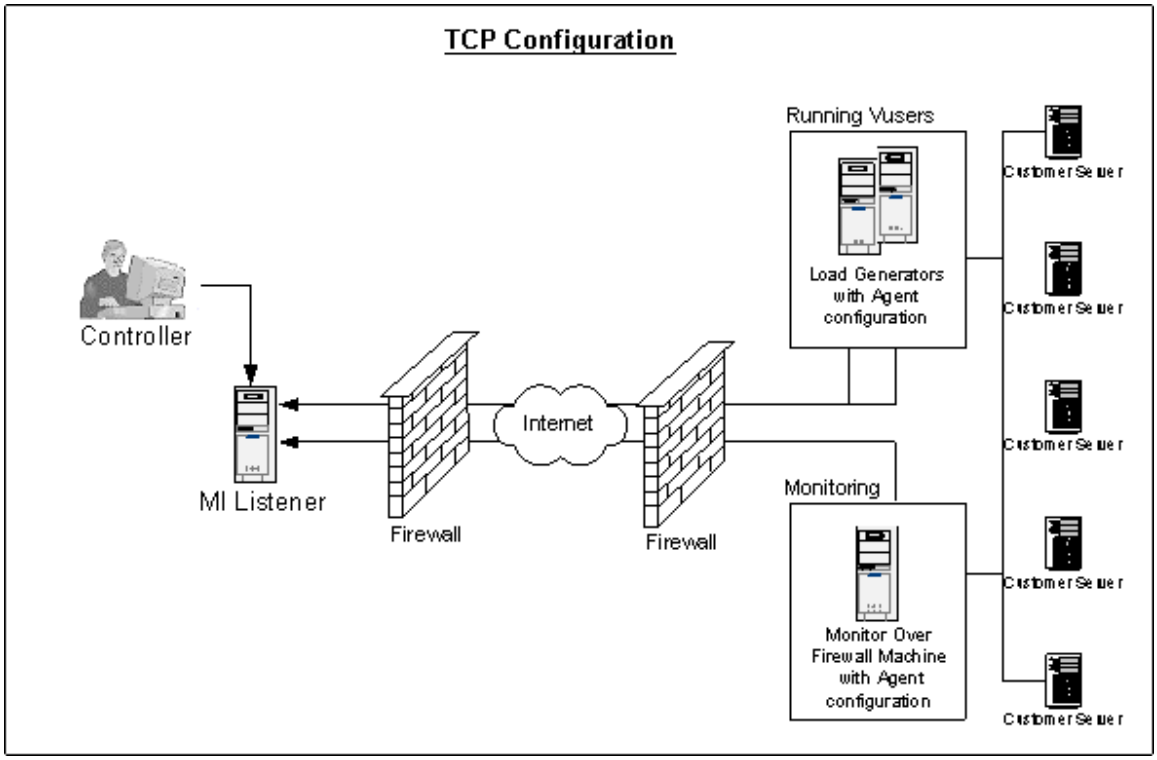
   Make sure you have installed the necessary components as described in the first step of "Set Up Your LoadRunner System Over Firewalls" on page 1136.

2. Set up your deployment (TCP or HTTPS)

   To run Vusers or monitor servers over the firewall, configure your system according to one of the following configurations. Note that these configurations contain a firewall on each LAN. There may also be configurations where there is a firewall for the Over-Firewall LAN only.

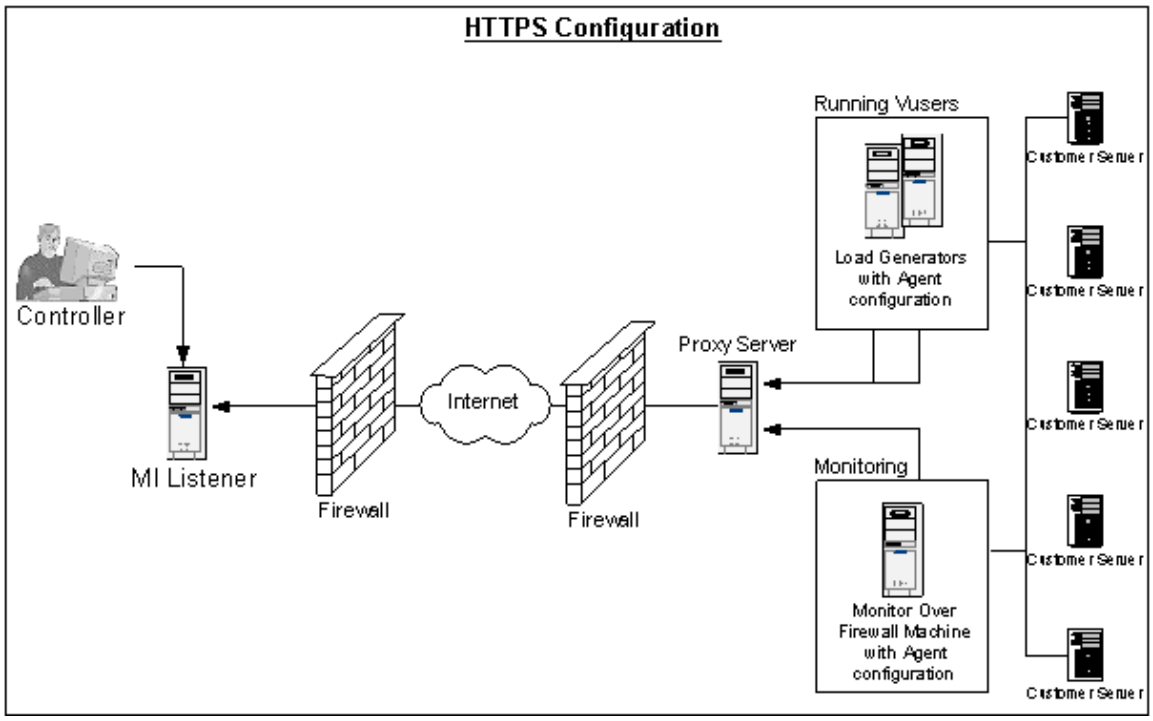
   - **TCP configuration**

     The TCP configuration requires every LoadRunner agent machine behind the customer's firewall to be allowed to open a port in the firewall for outgoing communication.




   - **HTTPS configuration**

     In the HTTPS configuration, only one machine (the proxy server) is allowed to open a port in the firewall. Therefore, it is necessary to tunnel all outgoing communications through the proxy server.




> **Tip:** The LoadRunner standalone Load Generator (LG SA) and standalone Monitor over Firewall (MOFW SA) cannot be installed on the same machine.
>
> However, LG SA can be used for monitoring purposes, the same way as the MOFW SA.
>
> A single machine cannot be used simultaneously for both running Vusers and monitoring.

3. Configure the firewall to allow agent access.

   Modify your firewall settings to enable communication between the machines inside the firewall and machines outside the firewall.

   a. **If your system has a TCP configuration:**

      The LoadRunner agent attempts to establish a connection with the MI Listener using port 443, at intervals specified in the Connection Timeout field in the Agent Configuration dialog box.

      To enable this connection, allow an outgoing connection for HTTPS service on the firewall for port 443. The agent can then connect to the MI Listener, and the MI Listener can connect back to the agent.

      From this point on, the agent listens to commands from the MI Listener.

   b. **If your system has an HTTPS configuration:**

      The LoadRunner agent attempts to establish a connection with the MI Listener, using the proxy port specified in the Proxy Port field, and at intervals specified in the Connection Timeout field in the Agent Configuration dialog box. When the connection is established, the proxy server connects to the MI Listener.

To enable this connection, allow an outgoing connection for HTTPS service on the firewall for port 443. The proxy server can then connect to the MI Listener, and the MI Listener can connect back to the agent through the proxy server.

From this point on, the agent listens to commands from the MI Listener.

c. **If you intend to start the LR Agent service from the Local System account, you need to grant it permissions:**

   i. Add a local user on the AUT machine with the same name and password as the local user on Agent machine.

   ii. Add the AUT local user to the **Performance Monitor Users** group

   iii. Restart the Agent process.

> **Note:** If you do not provide permissions, the monitor graph will not display any data.

4. Configure the MI Listener.

   To enable running Vusers or monitoring over a firewall, you need to install the MI Listener on one or more machines in the same LAN as the Controller outside the firewall. For installation instructions, see "Additional Components" on page 1566.

> **Note:**
>
> - The Controller installation automatically includes the MI Listener, so you can designate the Controller as the MI Listener machine.
> - The MI Listener can be installed only on Windows machines.

   For information on how to configure the MI Listener, see "MI Listener Configuration Dialog Box" on page 1144.

## Configure the LoadRunner Agent

On each load generator machine that will be running over a firewall and on each Monitor-Over-Firewall machine, you configure the LoadRunner agent to communicate with the MI Listener. The MI Listener serves as a router between the LoadRunner agent and the Controller.

1. Configure the Windows LoadRunner Agent.
   a. Stop the LoadRunner agent by right-clicking its icon in the system tray and selecting **Close**, and then setting the options in the Agent Configuration dialog box as described in "Agent Configuration Dialog Box" on page 1146.
   b. Restart the LoadRunner agent by double-clicking the shortcut on the desktop.
2. Configure and Run the Linux LoadRunner Agent.
   a. Open **<LoadRunner root folder>/dat/br_lnch_server.cfg** in a text editor.
   b. In the Firewall section, set `FireWallServiceActive` to `1` and save your changes.

c. Run **agent_config** from the **<LoadRunner root folder>/bin** folder to display the following menu:

```
Menu:
1. Show current settings.
2. Change a setting.
3. Save changes and exit.
4. Exit without saving.
5. Use default values.
```

> **Note:** If **agent_config** does not display the menu shown above, see "Troubleshooting and Limitations for Firewalls" on page 1150 for a possible solution.

d. Enter `1` to display the current settings:

```
Settings:
---------
1. MI Listener Name =
2. Local Machine Key =
3. Connection Timeout (seconds) = 20
4. Connection Type = TCP
5. Use Secure Connection (SSL) = False
6. Check Server Certificates = False
7. Client Certificate Owner = False
8. Private Key User Name =
9. Private Key Password =
10. Proxy Name =
11. Proxy Port =
12. Proxy User Name =
13. Proxy Password =
14. Proxy Domain =


Menu:
1. Show current settings.
2. Change a setting.
3. Save changes and exit.
4. Exit without saving.
5. Use default values.
```

e. To change a setting, enter `2` to display the settings menu:

```
Settings:
---------
1. MI Listener Name =
2. Local Machine Key =
3. Connection Timeout (seconds) = 20
4. Connection Type = TCP
5. Use Secure Connection (SSL) = False
6. Check Server Certificates = False
7. Client Certificate Owner = False
8. Private Key User Name =
9. Private Key Password =
10. Proxy Name =
11. Proxy Port =
12. Proxy User Name =
13. Proxy Password =
14. Proxy Domain =

Enter number of setting to change or 0 to go back to menu.
```

Enter the setting and continue according to the menu instructions. Set each option according to the "Agent Configuration Settings Dialog Box" on page 1148.

3. Restart the LoadRunner Agent.
   a. To remove (turn off) the LoadRunner agent, run the command `m_daemon_setup -remove` from the **<LoadRunner root folder>/bin** folder.

> **Note:** When the LoadRunner agent is configured to run over a firewall, and the agent is connected to the MI Listener, a file called `<local_machine_key>_connected_to_MI_Listener` is created in the temporary folder of the LoadRunner agent machine. The file is removed when the LoadRunner agent disconnects from the MI Listener.

   b. To start the LoadRunner agent, run the command `m_daemon_setup -install` from the **<LoadRunner root folder>/bin** folder.

## Create and Verify the Connection Between Controller and Agent Machines

This task describes how to set up your system to monitor the servers over a firewall.

1. Configure the Controller for Running Over a Firewall

   To run or monitor Vusers over a firewall, you need to create a unique connection between the Controller and the agent machines. Agent machines include load generator machines that will be running over a firewall and all Monitor-Over-Firewall machines.

   This connection is made through the MI Listener, which serves as a router between the Controller and the LoadRunner agent. To establish this connection, you configure the Controller machine to define the agent machine as a load generator.

   To configure the Controller for running Vusers or monitoring over the firewall:

a. Define the load generator settings in the Load Generators dialog box. For the server name, be sure to use the same Local Machine Key setting as in the Agent Configuration. For Details, see the "Load Generators Dialog Box" on page 1019.

b. Define the settings in the "MI Listener Configuration Dialog Box" on the next page, using the same MI Listener name as in the Agent Configuration.

> **Note:** You cannot change the temporary folder on the host running or monitoring Vusers over the firewall.

2. Enable and Confirm your Connection

   After you have configured the LoadRunner Agent, the MI Listener and the Controller, select the load generator in the Load Generators dialog box and click **Connect**.

   A green or red light next to the LoadRunner agent in the system tray indicates a successful or unsuccessful connection respectively.

## Set Firewall Monitoring Preferences

After you have installed and configured the LoadRunner agent, the Monitors over Firewall component, the MI Listener, and the Controller machine, you need to choose the server measurements that you want the Monitor-Over-Firewall machine to monitor.

You configure the server monitor properties from the Monitor-Over-Firewall machine, using the "Monitor Configuration Dialog Box" on page 1145. You can select the type of monitors to run and the server whose resources you want to monitor, add the measurements to monitor for each server, and specify the frequency with which you want the monitored measurements to be reported.

The following steps describe how to configure monitors over a firewall:

1. Open the Monitor Configuration Dialog Box.

   On the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Advanced Settings > Monitor Configuration**. For machines without the complete LoadRunner installation, rom the Windows Start menu, go to **Server Monitor > Monitor Configuration**.

2. Add monitored servers.

   a. To add servers, click the **Add Server** button . Type the name or IP address of the server whose resources you want to monitor in the **Monitored Server** field.

   > **Note:** To add several servers simultaneously, separate the server names or IP ranges with commas. For example: `255.255.255.0-255.255.255.5, server1, server2`.

   b. From the **Available Monitors** list, select the monitors appropriate for the server being monitored.

For certain monitors, LoadRunner displays default measurements in the **Measurements to be Monitored** section. You can specify the frequency by which LoadRunner reports the measurement in the **Measurement Properties** section. For details on selecting measurements, see "How to Set Up a Monitoring Environment" on page 1171.

3. (Optional) Clone a monitored server's properties.

   If you want to monitor the same properties on different server machines, right-click the server you want to clone, and select **Clone**. In the **Monitored Server** box, type the name or IP address of the clone server you want to create.

## MI Listener Configuration Dialog Box

This dialog box enables you to configure the MI Listener.

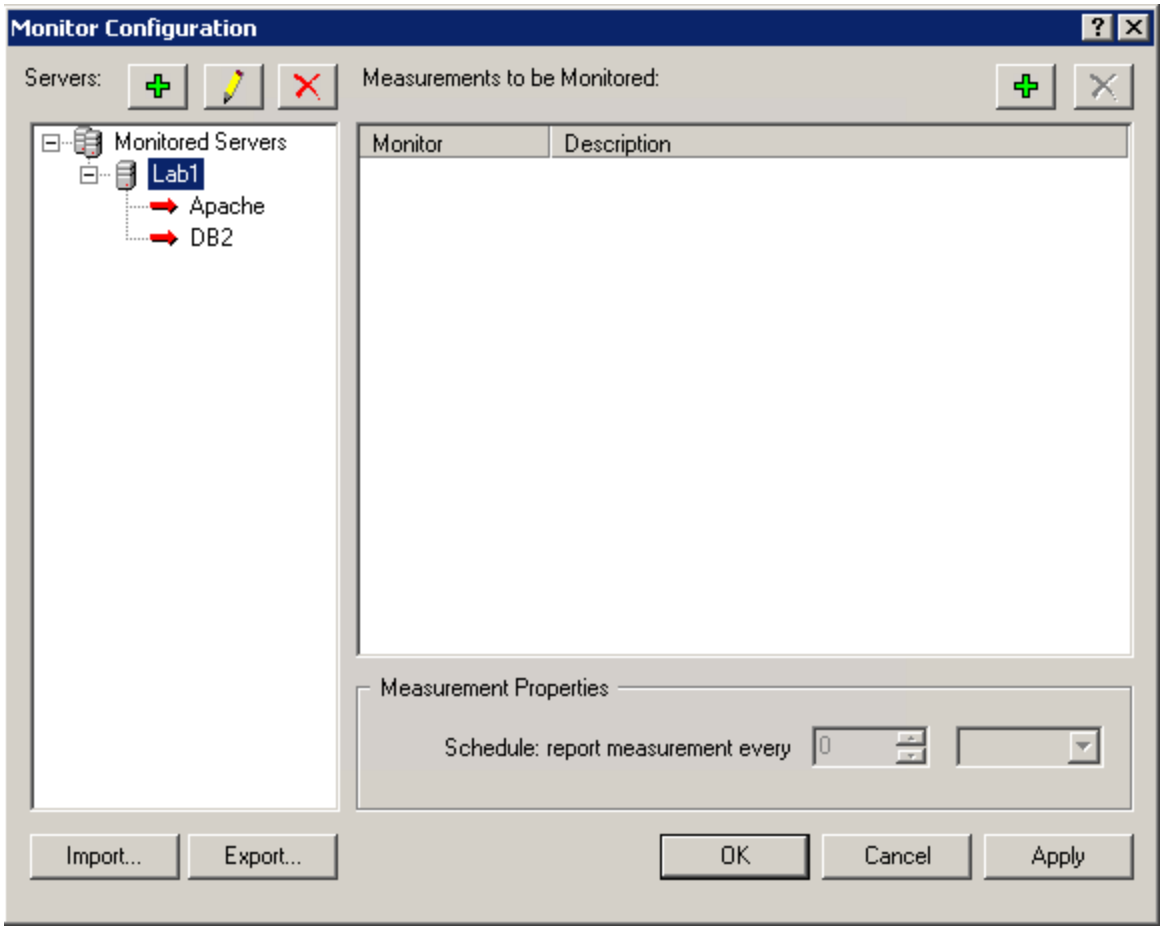
| | |
|---|---|
| **To access** | Do one of the following:<br>• From the Windows Start menu, go to **Micro Focus > LoadRunner > Advanced Settings > MI Listener Configuration**.<br>• Run **<LoadRunner root folder>\launch_service\bin\MILsnConfig.exe**. |
| **Important information** | • Before configuring the MI Listener:<br>  • Open incoming HTTPs service for port 443. The port settings are set by your system administrator.<br>  • Stop the LoadRunner agent on the MI Listener machine by right-clicking it's icon in the system tray and selecting **Close** from the popup menu.<br>• After configuring the MI Listener:<br>  • Restart the LoadRunner agent by double-clicking the shortcut on the LoadRunner machine's desktop, or by selecting **Micro Focus > LoadRunner Advanced Settings > Agent Service** from the Windows Start menu.<br>• Make sure that no Web Servers are running on the MI Listener or Monitor-Over-Firewall machine. These servers use port 443 and will not allow the access required by the listening and monitoring processes. |
| **Relevant tasks** | "Set Up an Over-Firewall Deployment" on page 1137 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Check Client Certificates** | Select **True** to request that the client send a TLS (SSL) certificate when connecting, and to authenticate the certificate. Default value is **False**. |
| **Private Key Password** | The password that may be required during the certificate authentication process. There is no default value. |

# Monitor Configuration Dialog Box

This dialog box enables you to select the type of monitors to run and the server whose resources you want to monitor, add the measurements to monitor for each server, and specify the frequency with which you want the monitored measurements to be reported.




| | |
|---|---|
| **To access** | From the Windows Start menu, go to **Micro Focus > LoadRunner > Advanced Settings > Monitor Configuration**.<br>For machines without the complete LoadRunner installation, from the Windows Start menu, go to **Server Monitor > Monitor Configuration**. |
| **Relevant tasks** | "Set Firewall Monitoring Preferences" on page 1143 |

User interface elements are described below:

| UI Element | Description |
|---|---|

| | |
|---|---|
| | Adds a server (left pane) or measurement (right pane) to the monitored server list. Type the name or IP address of the server whose resources you want to monitor in the **Monitored Server** field.<br><br>From the **Available Monitors** list, select the monitors appropriate for the server being monitored.<br><br>**Tip:**<br>• To add several servers simultaneously, separate the server names or IP ranges with commas. For example: `255.255.255.0-255.255.255.5, server1, server2`.<br>• If you want to monitor the same properties on different server machines, right-click the server you want to clone, and select **Clone**. In the Monitored Server box, type the name or IP address of the clone server you want to create. |
| | Removes a server or measurement. |
| | Opens the Monitored Server Properties dialog box, allowing you to modify the settings. |
| **Measurement Properties pane** | Allows you to set a measurement schedule for each measurement to be reported. Select the configured server measurement you want to schedule and specify the frequency at which you want LoadRunner to report the measurement. Click **Apply** to save your settings. |
| **Import / Export** | Imports or exports the monitor configuration details, including the server and their measurements, to/from an XML file. |

## Agent Configuration Dialog Box

This dialog box enables you to enable and configure the LoadRunner agent on Windows machines.

| | |
|---|---|
| **To access** | Do one of the following on the LoadRunner machine:<br>• From the Windows Start menu, go to **Micro Focus > LoadRunner > Advanced Settings > Agent Configuration**.<br>• Run **<LoadRunner root>\launch_service\bin\AgentConfig.exe**. |

| | |
|---|---|
| **Important information** | • The Agent icon does not appear in Windows 2008 when the Load Testing Agent Service is launched.<br>• When LoadRunner Agent runs as service (magentservice.exe), files (script, parameter file, etc.) that are stored on remote network drives or referred to by UNC path cannot be accessed. To access files this way, run the LoadRunner Agent as process (magentproc.exe).<br>**Workaround:** To access network share, configure the Load Testing Agent Service to run with an account that has network access permissions.<br>• When running the LoadRunner Agent as a service (magentservice.exe), and the Agent Configuration as administrator under UAC or a standard user: If you click the **OK** button and attempt to restart the Load Testing Agent Service, it issues the warning "Access is denied" .<br>**Workaround:**<br>a. Run the Agent Configuration as administrator when UAC is off.<br>b. If the current user is administrator or a user with appropriate permissions to work with Load Testing Agent Services, go to Service Manager (services.msc) and manually start the LoadRunnerAgent service (and for Performance Center, the RemoteManagementAgent service).<br>c. You can also run <LoadRunner root>\bin\subinacl.exe as administrator to grant a user (user1 in examples below) full access permission of the target service. For example,<br>**For LoadRunner:** `subinacl.exe /service LoadRunnerAgent /grant=user1`<br>**For Performance Center:** `subinacl.exe /service RemoteManagementAgent /grant=user1` |
| **Relevant tasks** | "Configure the LoadRunner Agent" on page 1140 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable Firewall Agent** | Enables running Vusers over a firewall. |
| Settings | This button is enabled only when the **Enable Firewall Agent** check box is selected.<br>Opens the "Agent Configuration Settings Dialog Box" on the next page. |
| **Enable Terminal Services** | Enables distributing Vusers on a terminal server. |

# Agent Configuration Settings Dialog Box

Define relevant settings to enable the LoadRunner agent on Windows machines.

| | |
|---|---|
| **To access** | On the LoadRunner machine:<br>1. From the Windows Start menu, go to **Micro Focus > LoadRunner > Advanced Settings > Agent Configuration**.<br>2. Select Enable Firewall Agent and click **Settings**. |
| **Relevant tasks** | "Configure the LoadRunner Agent" on page 1140 |
| **See also** | The Network and Security Manager configuration tool provides greater flexibility for defining the agent settings through the command line. For details, see "Network and Security Manager - Command Line Tool" on page 1163. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **MI Listener Name** | The name, full name, or IP address of the MI Listener. |
| **Local Machine Key** | A symbolic string identifier used to establish a unique connection between the Controller host and the agent machine, via the MI Listener machine. |
| **Connection Timeout (seconds)** | The length of time you want the agent to wait before retrying to connect to the MI Listener machine. If zero, the connection is kept open from the time the agent is run.<br>**Default value:** 20 seconds |
| **MI Listener User Name/Password** | The credentials needed to connect to the MI Listener machine. |
| **Server Domain** | The domain name needed to connect to the MI Listener machine. This field is required only if NTLM is used. |
| **Connection Type - TCP/HTTP** | Select either **TCP** or **HTTP**, depending on the configuration you are using.<br>**Default:** TCP |

| UI Element | Description |
|---|---|
| **Connection Type - HTTP** | • **Proxy Name.** The name of the proxy server.<br>This field is mandatory if the **Connection Type** setting is **HTTP**.<br>• **Proxy Port.** The proxy server connection port.<br>This field is mandatory if the **Connection Type** setting is **HTTP**.<br>• **Proxy User Name/Password.** The credentials of a user with connection rights to the proxy server.<br>• **Proxy Domain.** The user's domain if defined in the proxy server configuration. This option is required only if NTLM is used. |
| **Use Secure Connection (SSL)** | Enable connection using the TLS (formally SSL) protocol.<br>**Default:** Disabled<br>If you enable this option, enter the following information:<br>• **Check Server Certificates.** Authenticates the TLS certificates that are sent by the server.<br>  • **Medium.** Verifies that the server certificate is signed by a trusted Certification Authority.<br>  • **High.** Verifies that the sender IP matches the certificate information. This setting is available only if **Use Secure Connection** is enabled.<br>• **Private Key Password.** The password that might be required during the TLS certificate authentication process. This option is relevant only if the **Client Certificate Owner** option is enabled. |

# Secure communication over Firewalls

When the MI Listener sends its Public Key to the LoadRunner agent, it always sends its certificate as well: this is the server-side certificate. The LoadRunner agent can also be configured to authenticate the certificate it received. If the agent is configured to authenticate the certificate, it can verify whether the sender is really the machine that it claims to be by:

- Comparing the certificate's IP address with the sender's IP address
- Checking the validation date
- Looking for the digital signature in its Certification Authorities list

The MI Listener may also require the LoadRunner agent to send a certificate at any point in the session: this is the client-side certificate. You can set this option in the "MI Listener Configuration Dialog Box".

If the LoadRunner agent owns a certificate, it sends the certificate to the MI Listener for the same authentication process. If the LoadRunner agent does not own a certificate, the communication might not be continued.

**See also:**

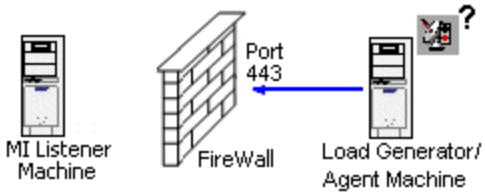
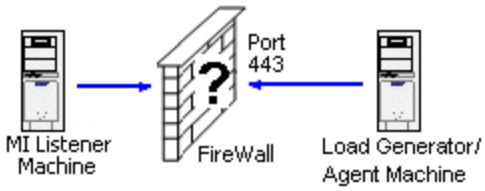
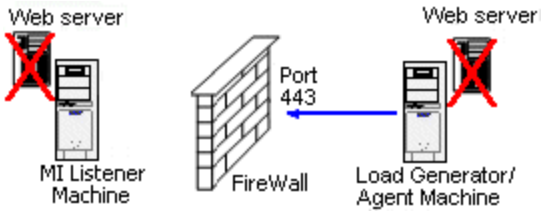
- "Secure Communication with TLS (SSL)" on page 1153
- "Two-way TLS (SSL) Authentication" on page 1158

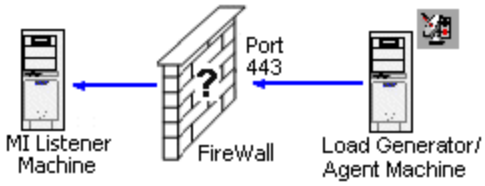
# Troubleshooting and Limitations for Firewalls

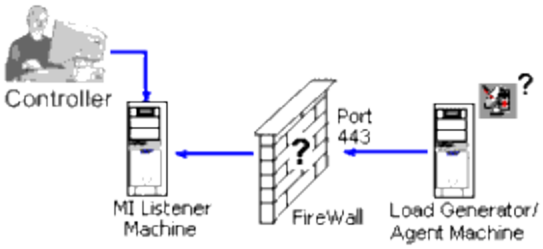
## Checking Connectivity

To run Vusers or monitor servers over a firewall, you must be able to establish a connection between the LoadRunner agent, MI Listener, and the Controller machine.

If you encounter connectivity problems after installing and configuring all the necessary components, check the table below for troubleshooting tips.

| Check | Solution |
|---|---|
| To check that the Firewall service was activated on the agent machine:<br>  | There should be a light on the right side of the LoadRunner Agent icon on the machine running/ monitoring Vusers over a firewall. If there is no light, this indicates that `FirewallServiceActive=1' is not set in the [FireWall] section of the Agent Settings. See "Agent Configuration Settings Dialog Box" on page 1148. |
| To check that port 443 is open:<br>  | On the agent machine, open a Command Prompt window, and type the following:<br>`telnet <MI_Listener_IP>443`<br>For example: `telnet 111.111.111.1111 443`<br>If port 443 is open, a new Telnet window will open.<br>If port 443 is not open, contact your network administrator. |
| To check that port 443 is available:<br>  | If a Web server is running on the MI Listener or Monitor-Over-Firewall machine, port 443 will not allow the access required by the listening and monitoring processes. Contact your network administrator to change the Web server port. |

| | |
|---|---|
| To check connectivity between the agent and the MI Listener, when running the LoadRunner agent as a service:<br>  <br> | If there is a red light on the right side of the LoadRunner Agent icon when running the LoadRunner agent as a service, do the following:<br>• Check that port 443 is open. See the troubleshooting tip "Troubleshooting and Limitations for Firewalls" on the previous page.<br>• Check that the Agent Settings and Agent Configuration are correctly set. See "Agent Configuration Settings Dialog Box" on page 1148.<br>• Run the agent as a Process. Launch **<Installation>\Launch_service\bin\ magentproc.exe**. If this works, this indicates an authentication issue with the Load Testing Agent Service. Browse to the **Service > LoadRunner Agent Service**, and change the properties of this service to `System User Account` or provide the user name and password of someone who has administrative privileges on this machine. |

| | |
|---|---|
| To check connectivity between the agent and the Controller, when monitoring over a firewall<br>  <br> | • Check that you entered the servers that you want to monitor in the Monitor Configuration dialog box. (See "Monitor Configuration Dialog Box" on page 1145.)<br>• Start the LoadRunner Agent Process on the Monitor-Over-Firewall machine. (See "Configure the LoadRunner Agent" on page 1140)<br>• On the Controller, enter the name of the Monitor-Over-Firewall machine in the Load Generators dialog box, and click **Connect**. After about a minute, data should start streaming in from the Monitor-Over-Firewall machine through the MI Listener to the Controller. (See "Set Firewall Monitoring Preferences" on page 1143.)<br>• If no data arrives at the Controller, try connecting the Controller to the MI Listener as if the Listener was used as a load generator. This will help identify the cause of the problem. Examine the log file on the Monitor-Over-Firewall machine by right-clicking the LoadRunner Agent icon. There should be no error messages.<br>• Start the MI Listener, and then manually start the LoadRunner Agent Process by running **<installation>\launch_service\bin\magnetproc.exe** on the Monitor-Over-Firewall machine. Allow the Monitor-Over-Firewall machine sufficient time to connect to the MI Listener, then connect the Controller to the Monitor-Over-Firewall machine. If the LoadRunner Agent Process crashes, either restart the agent or reboot the Monitor Over Firewall machine. |

## Windows Firewall Considerations

- In most Windows environments, Windows Firewall is turned on by default. The firewall does not allow certain LoadRunner components to communicate with each other. The Windows firewall therefore needs to be turned off.

> **Note:** Turning off Windows Firewall increases the risk to your computer's security.

- For each process that needs the firewall you can unblock the process by clicking the unblock button in the popup window that indicates that the program needs it, or by manually tuning the Windows firewall from the Exceptions tab.

### Linux LoadRunner Agent Menu

When you configure the Linux LoadRunner Agent, you must run **agent_config** to display the following menu:

```
Menu:
1. Show current settings.
2. Change a setting.
3. Save changes and exit.
4. Exit without saving.
5. Use default values.
```

For details, see "Configure the LoadRunner Agent" on page 1140. If the menu does not display, type the following command to check if the M_LROOT environment variable is set:

**Example:** echo $M_LROOT

If M_LROOT is not set, do either of the following:

1. Type `sudo su -` to inherit all environment variables of the current user.
2. Type `source <LoadGenerator_directory>/env.sh` to set all the Load Generator related environment variables after switching to root (i.e. "sudo su").

**Note:** This limitation occurs on Amazon cloud Linux machines, usually when you log in to the system using the special user named "ec2-user".

# Secure Communication with TLS (SSL)

Set up TLS (SSL) for secure communication between your LoadRunner machines.


- Configure Secure Communication with TLS (SSL) ..... 1153
- Two-way TLS (SSL) Authentication ..... 1158
- Configure Secure Communication with Two-way TLS (SSL) Authentication ..... 1160
- LoadRunner Certificate Manager ..... 1161
- Authentication Settings Tool ..... 1163
- Network and Security Manager - Command Line Tool ..... 1163


## Configure Secure Communication with TLS (SSL)

To set up secure communication using TLS (SSL), you need to install a CA certificate and TLS certificate on each LoadRunner machine. You can manage these certificates using the LoadRunner Certificate

Manager, or using a command line interface.

## Manage certificates using the LoadRunner Certificate Manager

Using the LoadRunner Certificate Manager, you can create a CA certificate (or select an already existing one), create server/client certificates, and install the certificates on your LoadRunner machines.

1. Launch the LoadRunner Certificate Manager from the Windows Start menu: go to**Micro Focus > LoadRunner > Tools > LoadRunner Certificate Manager**.

   If you have previously installed CA and TLS certificate with this application, these certificates are displayed. If not, the default LoadRunner certificates are displayed.

2. Select a CA certificate:
   a. Click the **Change** button.
   b. On the Select CA Certificate page, select a CA Certificate. If no certificate is displayed in the list, do one of the following:

| | |
|---|---|
| Create a new CA certificate | Click **New** and, in the Create New CA Certificate screen, enter the required details. When finished, click **Create**. Then select the certificate in the list.<br><br>**Tip:** To export the CA certificate so that you can install it on other LoadRunner machines, click **Export**. |
| Import an existing CA certificate | If you already have a CA certificate in your organization, you can import it:<br>i. Click **Import**.<br>ii. On the Import a CA Certificate page, click the **CA Certificate** and **Private Key** buttons to browse to and select the required files. Then click **Import**.<br>The imported certificate appears in the CA Certificates list.<br>iii. Select the imported certificate. |

   a. Click **Next**. The SSL Certificate page opens.

3. Select or create a TLS (SSL) certificate.

**Note:** For higher security, it is recommended to create and install a separate certificate for each machine.

| | |
|---|---|
| Create a new TLS certificate | a. On the Select SSL Certificate page, click **New** to create a new TLS certificate.<br><br>**Tip:** If you are creating one certificate for all LoadRunner machines, click **Export** to export the TLS certificate so that you can install it on other LoadRunner machines.<br><br>b. Click **Next**. |
| Import an existing TLS certificate | To import an existing certificate, use the **gen_cert** utility with the **-install** option, as described in "Manage certificates using the command line utility" below. |

4. Click **Finish**. The CA certificate and corresponding TLS certificate are installed on the current LoadRunner machine.
5. Restart the LoadRunner Agent service.

## Manage certificates using the command line utility

Use the **gen_ca_cert** utility to create a new CA certificate, and the **gen_cert** utility to create the TLS certificate.

1. Create a CA certificate using the **gen_ca_cert** command line utility:

   From the **<LoadRunner root>\bin** folder, run **gen_ca_cert**, using at least one of the following options:

   - **-country_name**
   - **-organization_name**
   - **-common_name**

   This process creates two files in the folder from which the utility was run: the CA Certificate (**cacert.cer**), and the CA Private Key (**capvk.cer**).
2. (Optional) Rename the files created by the utility.

   To rename the certificate files, use the **-CA_cert_file_name** and the **-CA_pk_file_name** options respectively.

**Note:** By default, the CA certificate is valid for three years from when it is generated. To change the validation dates, use the **-nb_time** (beginning of validity) and/or **-na_time** (end of validity) options.

**Example:**

The following command creates two files: **ca_igloo_cert.cer** and **ca_igloo_pk.cer** in the current folder, and sets the validity to 10/10/2013-11/11/2017:

```
gen_ca_cert -country_name "North Pole" -organization_name "Igloo Makers" -
common_name "ICL" -CA_cert_file_name "ca_igloo_cert.cer" -CA_pk_file_name "ca_
igloo_pk.cer" -nb_time 10/10/2017 -na_time 11/11/2017
```

3. Install the CA certificate.

   Use one of the following options:

| | |
|---|---|
| **-install <name of certificate file>** | Replaces any previous CA list and creates a new one that includes this CA certificate only. |
| **-install_add <name of certificate file>.** | Adds the new CA certificate to the existing CA list. |

**Note:** The **-install** and **-install_add** options install only the certificate file. Keep the private key file in a safe place and use it only for issuing certificates.

4. Create a TLS certificate using the **gen_cert** command line utility.

   From the **<LoadRunner root>\bin** folder, run:

| | |
|---|---|
| Windows | **gen_cert.exe** |
| Linux | **gen_cert** |

Run the **gen_cert** command with at least one of the following options:

- **-country_name**
- **-organization_name**
- **-organization_unit_name**
- **-eMail**
- **-common_name**

**Note:**

- The CA Certificate and the CA Private Key files are necessary for the creation of the TLS certificate. By default, it is assumed that they are in the current folder, and are named **cacert.cer** and **capvk.cer** respectively. In any other case, use the **-CA_cert_file_name** and **-CA_pk_file_name** options to give the correct files and locations.
- The certificate file is created in the folder from which the utility was run. By default, the file name is **cert.cer**. To rename the TLS certificate, use the **-cert_file_name** option.

5. Install the TLS certificate using the **gen_cert** command with the **-install** option.
6. Restart the LoadRunner Agent service.

## Other tools for setting up communication with communication with TLS

You can also use the following tools to set up TLS authentication:

| | |
|---|---|
| Network and Security Manager command line tool | Automate your certificate setup process with the LoadRunner Network and Security Manager command line tool using the **-generate_new_cert option** command to create a new TLS certificate.<br><br>For details, see "Network and Security Manager - Command Line Tool" on page 1163. |
| Controller | You can automatically generate a TLS certificate on Controller, associating the certificate with your scenario run.<br><br>To do this, in the "Authentication Settings Tool", select the **Generate a certificate automatically** option. |

## About Certificates

To set up the components in your LoadRunner environment to communicate securely using TLS (SSL), you need to install your Certification Authority (CA) certificate and a TLS certificate issued by that CA on each LoadRunner machine.

### LoadRunner Default Certificate

LoadRunner provides a default CA and TLS certificate for all LoadRunner machines. They are located in the **<LoadRunner root>\dat\cert** folder.

However, for a more secure process, create your own CA and issue matching TLS certificates for your machines. For details, see "Configure Secure Communication with Two-way TLS (SSL) Authentication" on page 1160.

### Certificate Attributes and Requirements

#### Certificates created by LoadRunner

In general, all certificates created by LoadRunner utilities have the following attributes:

- Signature hash algorithm: sha256
- Encryption algorithm: RSA (2048 Bits)

#### Requirements for using existing CA certificates

You can use an existing CA certificate from your own organization—one that was not created by LoadRunner—as long as it complies with the following:

- enclosed between `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----`.
- base64 encoded DER certificate (*.pem)

> **Tip:** If your certificate is not already in PEM format, you can use any known tool to convert it.

You can also provide certificate files that contain a root CA and one or more intermediate CAs. LoadRunner supports chain verification as long as all the certificates in the chain from the root to the client certificate can be verified.

For example, suppose your Controller machine **cacert.cer** verification file contains A (root), B (signed by root), C (signed by root).

Then suppose that on a load generator machine, the **cert.cer** certificate file contains D (signed by B) and E (signed by D).

The certificate chain is valid: A > B > D > E.

### See also:

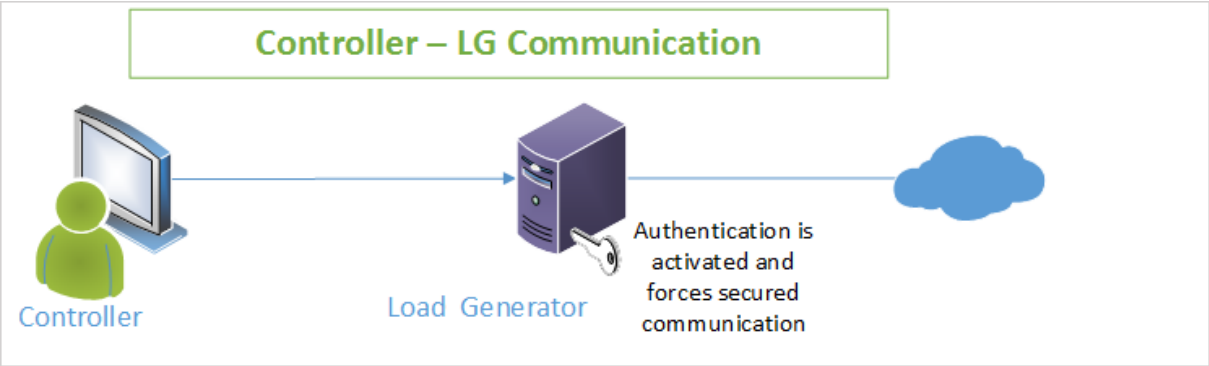
- "Configure Secure Communication with TLS (SSL)" on page 1153
- "Secure communication over Firewalls" on page 1149

# Two-way TLS (SSL) Authentication

You can configure your LoadRunner environment to communicated with two-way TLS (SSL) authentication. In this mode, the server send its own certificate and requests from the client to send it client certificate for validation.

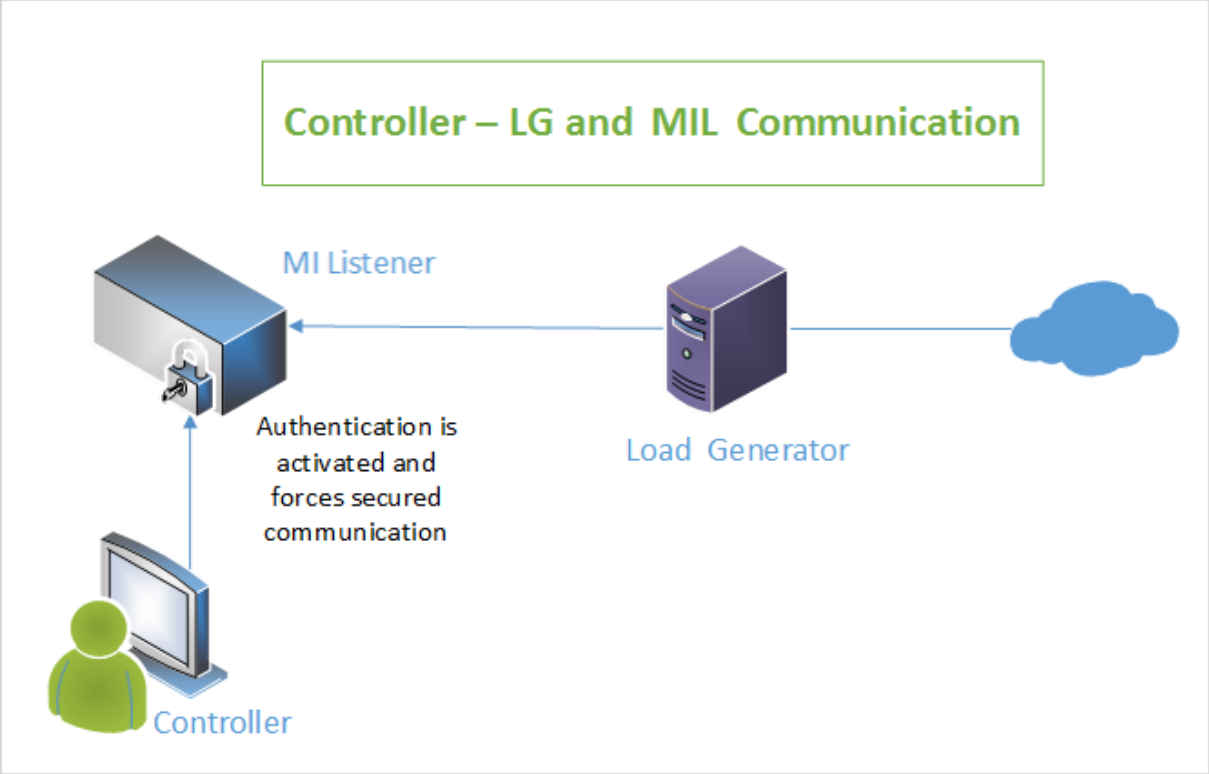
The following images illustrate some possible configurations when using two-way TLS authentication:

- **Controller--Load Generator Communication.** In this environment, the load generator machine is the server and Controller is the client. You activate authentication on the load generator, enforcing security between the machines.

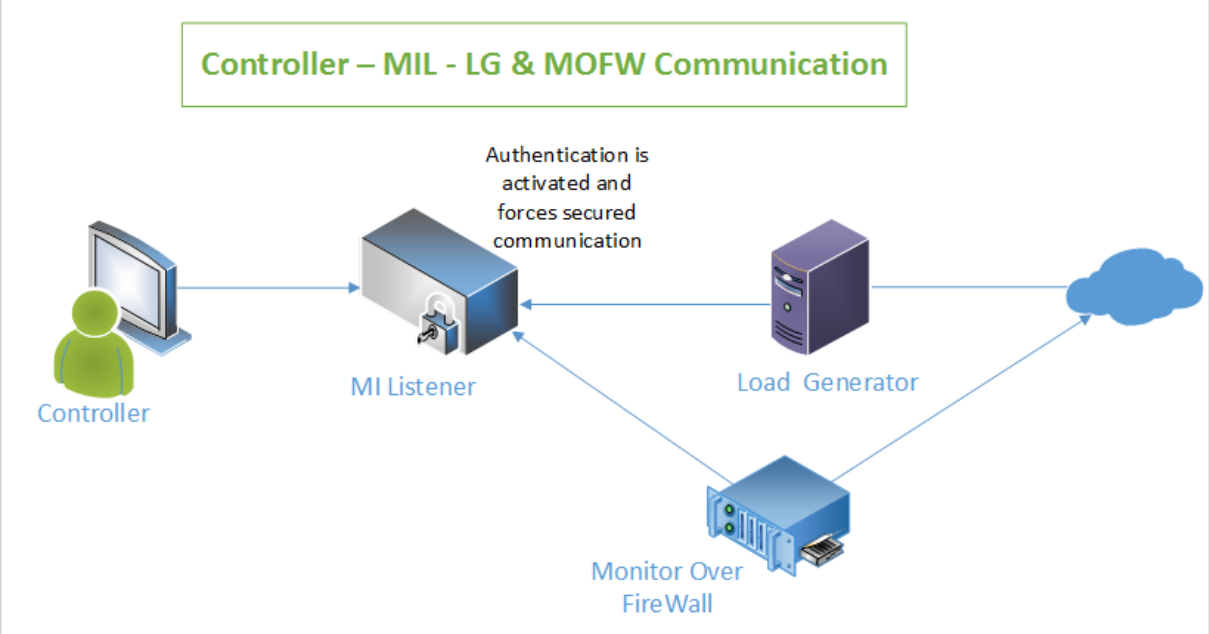



- **Controller--Load Generator--MI Listener Communication.** In this over-firewall environment, the MI Listener machine is the server, and Controller and load generators are the clients. You activate authentication on the MI Listener machine which, in turn, enforces security facing Controller and load

generators.




- **Controller--MI Listener--Load Generator--Monitor-Over-Firewall Communication.** In this over-firewall environment, the MI Listener machine is the server. The Monitor over Firewall, Controller, and load generator machines are the clients. You activate authentication on the MI Listener machine which, in turn, enforces security on the other machines.



### See also:

- "Configure Secure Communication with Two-way TLS (SSL) Authentication" below

# Configure Secure Communication with Two-way TLS (SSL) Authentication

This task describes the steps required to set up secure communication with two-way TLS (SSL) authentication on your LoadRunner environment.

1. Set up the CA and TLS certificates
   a. Create a Certificate Authority (CA) certificate or ensure that you have a valid existing CA certificate.
   b. Create or select a TLS certificate.
   c. Install the CA and TLS certificate on all relevant LoadRunner machines in your system.

   For details, see "Configure Secure Communication with TLS (SSL)" on page 1153.
2. Determine which machine in your system is the **server** for client-server communications. For details, see "Two-way TLS (SSL) Authentication" on page 1158.
3. Activate client authentication on the machine acting as the server.

   Do one of the following:

   - Use the Network and Security Manager command line tool to activate authentication for the host machine that is acting as the server. Set -check_client_cert to **1**. .
   - When working over a firewall, in the Agent Configuration Over Firewall Settings Dialog Box, select **Use Secure Connection (SSL)** and enter relevant details.
4. Configure Authentication Settings in Controller.

   In Controller, select the CA and TLS certificates to be used for the scenario run. For details, see "Authentication Settings Tool" on page 1163.
5. Activate server authentication on client machines (optional).

   If you want the client machines to validate the server certificate, set this option for each client machine.

   Do one of the following:

   - Use the following Network and Security Manager command line from a single location to activate server authentication on all relevant host machines: set **-check_server_cert** to **High** for the highest level of security. For more information, see **-check_server_cert**.
   - On each client machine in your system, enable the **Use Secure Connection (SSL)** option in the "Agent Configuration Settings Dialog Box" on page 1148, and set the **Check Server Certificates** level.

   > **Note:** When provisioning Load Generators on the cloud, the certificates will be taken from the Controller and automatically copied to the Load Generators, so the communication will

be secure by default. For details about working with load generators on the cloud, see "Load Generators on the Cloud" on page 987.

### See also:

- Blog: Illustrating the Security Model between Load Generators and LoadRunner's Controller
- Blog: LoadRunner's new Certificate Manager tool makes security easier than ever before!

# LoadRunner Certificate Manager

The LoadRunner Certificate Manager is a wizard that enables you to manage and install LoadRunner CA and TLS (formally SSL) certificates on your Controller machine and on other machines with LoadRunner components.

| | |
|---|---|
| **To access** | On the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > LoadRunner Certificate Manager** |
| **Relevant tasks** | "Configure Secure Communication with TLS (SSL)" on page 1153<br>"Configure Secure Communication with Two-way TLS (SSL) Authentication" on the previous page |

### CA Certificate page

This page displays the LoadRunner CA Certificate currently installed on the machine, together with its expiration date.

If there is a TLS certificate linked to the CA certificate, it is displayed below, together with its expiration date.

If no LoadRunner certificates have yet been added with the LoadRunner Certificate Manager, this page displays the default LoadRunner certificates.

To change the certificate, click **Change** to open the "Select CA certificate page" below.

### Select CA certificate page

This page enables you to manage CA certificates, and select one to install on the machine.

| UI Element | Description |
|---|---|
| **New** | Opens the Create New CA Certificate page, enabling you to add a CA certificate to the list of CA certificates. Enter the required details, and click **Create**. |

| UI Element | Description |
|---|---|
| **Delete** | Deletes the selected CA certificate from the list of certificates.<br>This also deletes any TLS certificates associated with the selected certificate.<br>**Note:** If the certificate is installed, deleting the certificate from this list does not uninstall it from the machine. |
| **Export** | Exports the selected certificate to a **.cer** file, enabling you to import it to other LoadRunner machines. |
| **Import** | Enables you to import a CA certificate (.cer) file to other LoadRunner machines. |
| **Next** | Opens the "Select SSL certificate of <CA certificate> page" below where you can link a TLS certificate to the selected CA certificate. |

### Select SSL certificate of <CA certificate> page

This page enables you to manage TLS certificates, and select one to install on the machine.

| UI Element | Description |
|---|---|
| **New** | Opens the Create New SSL Certificate page, enabling you to link a new TLS certificate to the parent CA certificate. Enter the required details, and click **Create**.<br>**Tip:** The parent CA certificate is displayed below the details. |
| **Delete** | Deletes the selected certificate from the list of certificates. |
| **Export** | Saves the selected certificate to a **.cer** file, enabling you to import it to other LoadRunner machines. |
| **Next** | Opens the "Pre-installation summary page" below. |

### Pre-installation summary page

This page displays a summary of the certificates you selected to install.

Click **Finish** to install the certificates on the machine.

## See also:

- "Secure Communication with TLS (SSL)" on page 1153
- Blog: LoadRunner's new Certificate Manager tool makes security easier than ever before!
- Blog: Illustrating the Security Model between Load Generators and LoadRunner's Controller

# Authentication Settings Tool

This dialog box enables you to select a TLS (SSL) certificate for your scenario run, or create one automatically.

| **To access** | **Controller > Tools > Authentication Settings** |
|---|---|
| **Important information** | • Once you create a CA certificate, you must use it on all of the LoadRunner machines.<br>• After setting up or modifying the certificates, you must restart the Controller to apply the changes. |
| **Relevant tasks** | "Configure Secure Communication with Two-way TLS (SSL) Authentication" on page 1160 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **CA certificate file** | The CA certificate file (with a .cer extension). |
| **Generate a certificate automatically** | Generates an TLS certificate based on the common name and CA private key.<br>You can specify a common name and CA private key for the certificate.<br>Click **Advanced settings** to specify a country, organization, organization unit, email, and validity range.<br>**Note:** If you make changes to the default CA, the connection will no longer be secure. |
| **Choose an existing certificate file** | The TLS certificate file (with a .cer extension).<br>This certificate will be sent to a load generator or MI Listener with enforced TLS client authentication. Only a certificate issued on the other end by the same CA will be trusted. |

# Network and Security Manager - Command Line Tool

You can use the LoadRunner Network and Security Manager command line tool, **lr_agent_settings**, to update and configure agent-related settings on local and remote machines.

Using a single command, you can update and configure **agent ports**, **agent settings**, and **authentication**.

You can also use this tool to automate your test and run it through the command line.

## Launching the Network and Security Manager tool

To launch the tool, open a command line window and run the following:

| | |
|---|---|
| **Windows** | For LoadRunner, Standalone Load Generator, Standalone MI Listener, and Monitor Over Firewall:<br>**<LoadRunner root>\bin\lr_agent_settings.exe**<br>**Note:** You must have write permissions to the LoadRunner installation folder. |
| **Linux** | For Standalone Load Generator:<br>**<LG path>\bin\lr_agent_settings.exe**<br>**Guidelines for Linux machines:**<br>• You must have administrator privileges to run this on a Linux machine.<br>• In Amazon cloud environments (AWS), you need to set the M_LROOT environment variable, as shown in the following example:<br>`~$ sudo M_LROOT=/opt/MF/MF_LoadGenerator /opt/MF/MF_LoadGenerator/bin/lr_agent_settings -check_client_cert 0` |

## Command line arguments for the Network and Security Manager tool

The following command line arguments are supported by this tool:

**Note:**

- To retrieve the list of available arguments, on the LoadRunner machine type **lr_agent_settings.exe -usage** or **lr_agent_settings** with no arguments.
- LoadRunner currently supports basic and NTLM proxy authentication.
- You can update the certificates on a remote machine only if:
  - A secure connection was established using TLS (SSL) authentication.
  - The client (the machine the tool is running on) was authenticated by the CA certificate on the remote load generator.
  - Both of the above items were achieved by using certificates other than the defaults.

| Option | Arguments | Description |
|---|---|---|
| **Remote Update Options** | | |

| | | |
|---|---|---|
| **-remote_host** | remote host name or IP | The names of the hosts to update with the new settings.<br>To access the local machine, specify **localhost** or **127.0.0.1**.<br>For multiple machines, repeat the command: For example,<br>**-remote_host host1 - remote_host host2**. |
| **-remote_hosts_ file** | file name | The name of a file containing the host names or IP addresses. Separate multiple host names with a line break. For hosts over a firewall, specify a port. For example,<br>**myserver1**<br>**myserver2:my_ofw_unix**<br>**myserver2:my_ofw_1** |
| **Agent Port Options** | | |
| **-m_agent_port** | port | The Load Generator m_agent listening port.<br>**Default**: 54345 |
| **-al_agent_port** | port | The Load Generator al_agent listening port.<br>**Default**: 54245 |
| **-mil_port_ controller** | port | MIL listening port from Controller. This option is not available on Linux.<br>**Default**: 50500 |
| **-mil_port_lg** | port | MIL listening port from Load Generator over firewall.<br>If you change this port value, you should also change the port for the Load Generator over firewall machine using **-mil_port**. This option is not available on Linux.<br>**Default**: 443 |
| **Load Generator Over Firewall Options** | | |
| **-is_ofw** | 0 \| 1 | Indicates whether to communicate over a firewall. |
| **-mil_name** | Host name or IP address | Changes the MI listener name or IP address from the side of the load generator over a firewall. |
| **-mil_port** | port number | Changes the port for the MI listener from the side of the load generator over a firewall.<br>**Default:** 443 |

| | | |
|---|---|---|
| **-local_machine_key** | Local machine key | Changes the host symbol (or local machine key) for load generator over a firewall, to establish a unique connection from behind the firewall. |
| **-mil_string** | MILname:local machine key | Changes the MI Listener name and the local machine key in one string separated by a colon, ":". |
| **-mil_username**, **-mil_passwd**, **-mil_domain** | username, password, domain | Changes the user name and password with which to connect to the MI Listener machine, and the domain for MI Listener machine (required only if NTLM is used). |
| **-sampling_interval** | sampling interval in seconds | Changes the sampling interval in seconds—the time the agent waits before retrying to connect to MI Listener machine.<br>The Over Firewall load generator machine polls the MI Listener regularly to see if any Controller needs to use it for a test run. If no request is found, it closes the connection and waits this specified timeout period, before polling it again. |
| **-channel_type** | TCP \| HTTP | Changes the connection type: HTTP or TCP. |
| **-proxy_name**, **-proxy_port** | hostname, port | Changes the name and port of proxy server when using HTTP connection. |
| **-proxy_string** | proxy name:proxy port | Changes the proxy name and port in one string separated by a colon, ":", |
| **-proxy_username**, **-proxy_passwd**, **-proxy_domain** | username, password, domain | Changes the user name and password with which to connect to the proxy server. |
| **-use_ssl** | 0 \| 1 | Changes the flag to connect using the TLS (formally SSL) protocol. |
| **-private_key_pwd** | username, password, domain | Changes the password that is optionally required during TLS (SSL) certificate authentication. |
| **Certificate Authentication Options** | | |

| | | |
|---|---|---|
| **-check_client_cert** | 0 \| 1 | Use **0** to allow both TLS (SSL) and non-TLS connections.<br><br>Use **1** to enforce TLS (SSL) connections only and check if the client certificate is trusted by the CA installed on the agent machine.<br><br>**Note:** When monitoring over firewall, set this flag to **1** on the **server** machine. For guidelines on determining which machine is considered the server, see "Two-way TLS (SSL) Authentication" on page 1158. |
| **-check_server_cert** | None \| Medium \| High | Indicates how to authenticate TLS (SSL) certificates that are sent by the server.<br><br>**None**. Does not authenticate the certificate.<br><br>**Medium**. Verifies that the server certificate is signed by a trusted Certification Authority.<br><br>**High**. Verifies that the sender IP matches the certificate information. |
| **-CA_cert_file_name** | CA certificate file name | Installs a CA certificate locally. It overwrites the **dat\cert\verify\cacert.cer** file, but does not affect any configuration file.<br><br>**Note:** You need to generate a CA certificate before installing it. To generate the CA certificate, run **gen_ca_cert -common_name <your_selected_common_name, e.g. LoadRunner >** from the **bin** folder.<br><br>Two files, **cacert.cer** and **capvk.cer**. are generated in the current directory, for the CA certificate and private key.<br><br>Store **capvk.cer** securely in a designated folder.<br><br>Install **cacert.cer** as a CA certificate on all of your LR/PC installations. |
| **-cert_file_name** | full path of certificate file | Installs authentication certificate locally. It overwrites the **dat\cert\verify\cacert.cer** file, but does not affect any configuration file.<br><br>**Note:** This step assumes you already generated a TLS certificate ahead of time. To generate a TLS certificate, run the following from the **bin** folder:<br><br>**gen_cert -common_name <your_selected_common_name, e.g. LoadRunner> -CA_cert_file_name <CA_cert_file_full_path> -CA_pk_file_name <CA_private_key_file_full_path>**. |

| | | |
|---|---|---|
| **-generate_new_cert_file -CA_private_key_file_name** | CA_private_key_file_full_path | Generates a new authentication certificate and installs it in **dat\cert\cert.cer**.<br><br>**Note:** A CA private key is mandatory to generate a self-signed certificates. The CA certificate will be read from **dat\cert\verify\cacert.cer** from the current machine. |
| **-private_key_file_name** | private key file name | Sets the matching private key of the TLS (SSL) certificate you installed with **gen_ca_cert -common_name** (see above). If you generate a TLS certificate using **gen_cert** or using the **-generate_new_cert_file** option in this tool, you can skip this step. You only need to do this step if you use a certificate which does not include a private key in the certificate file, such as the **openssl** tool. |
| **Restart Agent Options** | | |
| **-restart_agent** | | Restarts the **magent** or **alagent**. It automatically detects whether it is running as a service or process.<br><br>**Note:** If the agent is running as a process and you want to use the command line to restart it as a service, first use the "Agent Configuration Settings Dialog Box" on page 1148 to change between the Process and Service mode for the agent. |
| **Read Input Parameters** | | |
| **-prm** | parameter file | Retrieves the value of input parameters listed in a file. The prm file should have the following format:<br><br>`-mil_name MyHost1 -local_machine_key my_ofw_win -channel_type HTTP -proxy_name www.acme.com -proxy_port 8080.`<br><br>**Note:**<br>• When using the **-prm** argument in the command line, all other arguments are ignored.<br>• The parameter file should only contain arguments for changing settings. It should not contain the Remote Update arguments, **-remote_host** and **-remote_file**, which will be ignored. |

## Common Examples

Below are some examples for using the Network and Security Manager command line tool to change settings for agent ports, load generator over firewall settings, host security settings, certificate

authentication, and so on.

> **Note:** To use this utility on Linux load generators on an Amazon cloud environment (AWS), you need to set the M_LROOT environment variable as shown in the following example:
>
> ```
> ~$ sudo M_LROOT=/opt/MF/MF_LoadGenerator /opt/MF/MF_LoadGenerator/bin/lr_agent_settings -check_client_cert 0
> ```

### Set the agent proxy and port, and the MI Listener over a firewall

```
lr_agent_settings.exe -proxy_name www.apache.com -proxy_port 8080
```

```
lr_agent_settings.exe -m_agent_port 54888
```

```
lr_agent_settings.exe -proxy_string web-proxy.sgp.proxy-server.com:8080
(The string before ":" is proxy name, the string after ":" is proxy port)
```

```
lr_agent_settings.exe -mil_string MyServer2:my_ofw_unix (The string before
":" is MIL name, the string after ":" is the local machine key)
```

### Read parameters from a file

```
lr_agent_settings.exe -prm C:\my_settings.prm
```

where the parameter file is a text file with the parameters you want to use to change the settings, for example:

```
-mil_name MyServer3 -local_machine_key my_ofw_win -channel_type HTTP -proxy_name
www.apache.com -proxy_port 8080
```

### Remote updates

```
lr_agent_settings.exe -remote_host MyServer1 -proxy_string
www.apache.com:8080
```

```
lr_agent_settings.exe -remote_host MyServer1 -prm C:\my_settings.prm
```

```
lr_agent_settings.exe -remote_host MyServer2:my_ofw_unix -prm C:\my_
```

```
settings.prm (MyServer2:my_fow_unix says the remote host is OFW, the name
before ":" is MIL name, the string after ":" is local machine key)
```

```
lr_agent_settings.exe -remote_host localhost/127.0.0.1 -proxy_string
www.apache.com:8080 (Updates local host)
```

### Remote updates - multiple

```
lr_agent_setttings.exe -remote_host MyServer1 -remote_host vmlrrnd192 -
use_ssl 1
```

```
lr_agent_settings.exe -remote_host localhost -remote_host vmlrrnd192 -use_
ssl 1
```

```
lr_agent_settings.exe -remote_host MyServer1 -remote_host MyServer2:my_
ofw_unix -prm C:\my_settings.prm
```

### Remote updates - multiple from file

```
lr_agent_settings.exe -remote_file C:\remote_hosts.txt -proxy_string
www.apache.com:8088
```

```
lr_agent_settings.exe -remote_file C:\remote_hosts.txt -prm C:\my_
settings.prm
```

The file contains the hosts separated by line breaks:

```
myserver1
myserver2:my_ofw_unix
myserver2:my_ofw_1
```

### Restarting the agent

```
lr_agent_settings.exe -restart_agent
```

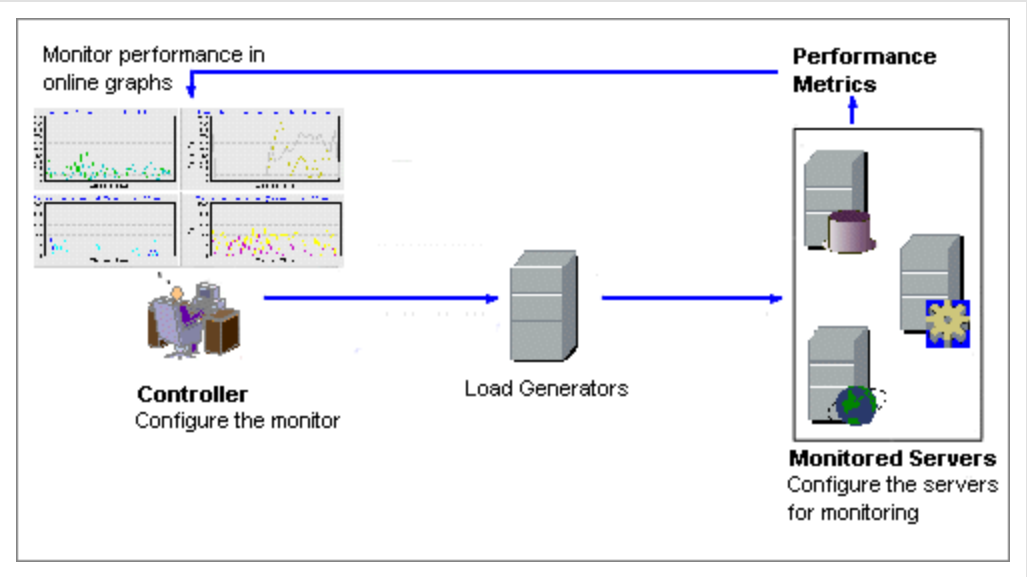
```
lr_agent_settings.exe -is_ofw 1 -mil_string MyServer2:my_ofw_win -restart_
agent
```

```
lr_agent_settings.exe -remote_host MyServer1 -remote_host MyServer2:my_
ofw_unix -restart_agent
```

> **Note:** If you encounter Access Denied warnings when restarting the service, see "Agent Configuration Dialog Box" on page 1146 for details.

# Monitoring Load Test Scenarios

Before monitoring a scenario, you need to set up and configure the LoadRunner monitoring components. Each monitor has different configuration requirements that are explained in the specific monitoring chapters. The following diagram illustrates the LoadRunner monitoring process.




### What would you like to do?

- Set up the monitoring environment
- Learn about the monitor types

# How to Set Up a Monitoring Environment

This task describes how to set up the LoadRunner online monitoring environment. You specify the machines and measurements that Controller will monitor during a scenario execution in Controller's Run tab. During scenario execution, the collected measurement data appears in the online graphs.

To set up the LoadRunner online monitoring environment:

1. Configure the monitoring environment on the server machine

   To use the following monitors, you must first install or configure monitoring components on the server machine. For details about configuring the monitoring components, see the specific monitoring sections.

   - "How to Set up the Citrix Monitoring Environment" on page 1178
   - "How to Set Up the IBM WebSphere MQ Monitor" on page 1193
   - "How to Configure a LoadRunner Scenario to use J2EE/.NET Diagnostics" on page 1259
   - "How to Set Up the Network Monitoring Environment" on page 1202
   - "How to Set Up the Oracle Monitoring Environment" on page 1185
   - "How to Set up the UNIX Monitoring Environment" on page 1225
   - "Dynatrace Monitor" on page 1189

   > **Note:** For information on how to set up the SiteScope monitoring environment, refer to the SiteScope documentation.

2. Add the monitored server to Controller

   Select the server whose monitors you want to configure.

   To monitor a server from Controller, you need to add the machine and the measurements that you want to monitor.

   a. Click the desired monitor graph in the graph tree, and drag it into the right pane of the Run tab.
      - Some monitors are native LoadRunner monitors (by default), but you can also monitor through the SiteScope monitor engine. To monitor through the SiteScope monitor, double-click it in the **Sitescope Graphs** node.

   b. Right-click the graph and select **Add Measurements**, or click anywhere on the graph and select **Monitors > Add Measurements**. The <Monitor> dialog box opens.

   c. In the **Monitored Server Machines** section of the <Monitor> dialog box, click **Add**. The Add Machine dialog box opens.
      - Enter the server name or IP address of the machine you want to monitor. Select the platform on which the machine runs.
      - For SiteScope monitors, enter the name and port number of the SiteScope server, and specify whether you are using a Secure HTTP connection. You can also provide different user credentials. For details, see "Add Machine Dialog Box" on page 1175.

3. Select the measurements that you want to monitor

   a. Make sure that the monitor you are configuring is selected in the **Monitored Server Machines** area of the <Monitor> dialog box.

   b. In the **Resource Measurements** section of the <Monitor> dialog box, click **Add**. The <monitor> Configuration dialog box opens. Choose the measurements for the specific server.

> **Note:** For the Citrix monitor, if the dialog box freezes after clicking **Add**, you may need to rebuild the localhost cache on the Citrix server machine.

For user interface details, see "<Monitor Type> Monitor Configuration Dialog Box" on page 1177.

For details about each monitor's default measurements, refer to the relevant reference section for the monitor.

4. Change the monitor's default counters - Optional

   When you configure the System Resource, Microsoft IIS, Microsoft ASP, and SQL Server monitors, you are presented with a list of default counters that you can measure on the server you are monitoring. You can change the default counters for these monitors by editing the **res_mon.dft** file found in the **LoadRunner\dat** folder.

   a. Open a new scenario and click the **Run** tab.

   b. For each of the monitors, select the counters you want to measure.

   c. Save the scenario and open the scenario **.lrs** file and **res_mon.dft** file with an editor.

   d. From the scenario **.lrs** file, copy the **MonItemPlus** section of the each counter you selected into the **res_mon.dft** file.

   e. Count the number of new counters in the **res_mon.dft** file and update the **ListCount** parameter with this number.

# Monitor Types

All of the monitors allow you to view a summary of the collected data at the conclusion of the scenario. Using LoadRunner Analysis, you can generate a graph for any of the monitors. For more information, see "Analysis Graphs" on page 1370.

The online monitors are divided into the following categories:

| | |
|---|---|
| **Application Deployment Solutions graphs** | Measures statistics related to the Citrix server during a scenario run. For more information, see "Application Deployment Solution Monitoring" on page 1178. |
| **Database Server Resource graphs** | Measure statistics related to Oracle and SQL Server databases during the scenario run. For more information, see "Database Server Resource Monitoring" on page 1185. |
| **Dynatrace monitor** | Provides information from the Dynatrace server about monitored applications. For more information, see "Dynatrace Monitor" on page 1189. |
| **Flex monitors** | Measure statistics related to Flex RTMP connections and throughput, as well as buffering time. For details, see "Flex Monitoring Graphs" on page 1190. |

| | |
|---|---|
| **Infrastructure Resources graphs** | Display information about network client data points during a scenario run using the Network Client graph. For more information, see "Infrastructure Resources Monitoring" on page 1191. |
| **JMeter graphs** | Display data for the JMeter test in a scenario run. For more information, see "JMeter Monitoring Graphs" on page 1191. |
| **Middleware Performance graphs** | Measure statistics related to IBM WebSphere MQ during a scenario run. For more information, see "Middleware Performance Monitoring" on page 1192. |
| **MQTT Statistic graphs** | Measure statistics related to MQTT messages, throughput, and connections during a scenario run. For more information, see "MQTT Statistics Monitoring" on page 1198. |
| **Network graphs** | Displays information about the network delays on your system. For more information, see "Network Delay Monitoring" on page 1201. |
| **Network Virtualization graphs** | Measure statistics related to network virtualization, such as Packet Loss and Latency. For information, see "Network Virtualization Monitoring" on page 1209. |
| **Runtime graphs** | Display the number and status of Vusers participating in the scenario, as well as the number and types of errors that the Vusers generate. For more information, see "Runtime and Transaction Monitoring" on page 1217. |
| **Service Virtualization graphs** | Measure statistics related to Service Virtualization, per operation and service. For information, see "Service Virtualization Monitors" on page 1255. |
| **SiteScope graphs** | Use the SiteScope monitor and its counters to measure resources. For information, see the documentation provided with SiteScope. |
| **Streaming State Statistics graphs** | Monitor statistics about streaming videos during a scenario run. For more information, see "Streaming State Statistics Monitoring" on page 1221. |
| **System Resource graphs** | Measure the Windows, UNIX, Server, and SNMP resources used during a scenario run. For more information, see "System Resource Monitoring" on page 1223. |
| **Transaction graphs** | Display the transaction rate and response times. For more information, see "Runtime and Transaction Monitoring" on page 1217. |
| **TruClient Native Mobile graphs** | Provide information about CPU utilization, free memory on the device, and memory consumption for TruClient Native Mobile Vusers. For information, see "TruClient - Native Mobile Graphs" on page 1487or go to the TruClient Help Center (select the relevant version). |

| | |
|---|---|
| **Web Application Server graphs** | Measure statistics related to the Microsoft ASP application servers during the scenario run. For more information, see "Web Application Server Monitoring" on page 1228. |
| **Web Resource graphs** | Provide information about the number of Web connections, throughput volume, HTTP responses, server retries, and downloaded pages at the Web servers during the scenario run. For more information, see "Web Resource Monitors" on page 1230. |
| **Web Server Resource graphs** | Measure statistics related to the Microsoft IIS and Apache Web servers during the scenario run. For more information, see "Web Server Resource Monitoring" on page 1235. |

The **J2EE & .NET Diagnostics Monitor** provides trace, time, and troubleshooting information for individual transactions through J2EE Web, application, and database servers. This monitor requires an additional installation, and is accessed from the **Diagnostics for J2EE/NET** tab in the Controller. For more information, contact your Customer support representative.

# Add Machine Dialog Box

This dialog box enables you to add the machine that you want to monitor to the Monitored Server Machines list.

## Access information

| | |
|---|---|
| **To access** | Right-click a graph **>** **Add Measurements**. In the dialog box that opens, click **Add** in the **Monitored Server Machine** section. |
| **Relevant tasks** | "How to Set Up a Monitoring Environment" on page 1171 |

## Monitored Machine Information

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Name** | The name or IP address of the machine that you want to monitor.<br>• **SNMP monitor:** If the SNMP agent is running on a different port than the default SNMP port, you need to define the port number using the format: **<server name>:<port number>**<br>• **MS IIS monitor:** To monitor an IIS server through a firewall, use TCP, port 139. |

| | |
|---|---|
| **Platform** | Type of platform (**Windows** or **Linux**) of the machine you want to monitor.<br>**Note:** This field is disabled for monitor configurations that do not have platform specifications (such as Apache), or when the monitor is intended to be used with a single platform type. |

## SiteScope Server Information

For SiteScope server monitoring there are additional user interface elements:

> **Note:** In order to connect to a Sitescope server with specific credentials, you must first stop the Sitescope monitor and remove all of its counters.

| UI Element | Description |
|---|---|
| **Name** | The name of the SiteScope server. |
| **Port** | The SiteScope port.<br>**Default:** 8888 |
| **Use Secure HTTP** | Uses a secure HTTP connection to the SiteScope server. |
| **Use Account** | Instructs LoadRunner to use a specific SiteScope user account with the following account details:<br>• **Account.** Internal SiteScope account name.<br>• **Username.** The display name of SiteScope account.<br>• **Password.** The password for the SiteScope account. |

## Dynatrace Server Information

For Dynatrace server monitoring there are additional user interface elements:

> **Note:** In order to connect to a Dynatrace server with specific credentials, you must first stop the Dynatrace monitor and remove all of its counters.

| UI Element | Description |
|---|---|
| **Name** | The name of the Dynatrace server. |
| **Port** | The Dynatrace port.<br>• HTTP: 8020<br>• HTTPS: 8021 (Default) |

| | |
|---|---|
| **Use Secure HTTP** | Uses a secure HTTP connection to the Dynatrace server. |
| **User Name** | Instructs LoadRunner to use the name of the Dynatrace account. |
| **Password** | Instructs LoadRunner to use the password for the Dynatrace account. |

# <Monitor Type> Monitor Configuration Dialog Box

This dialog enables you to select the measurements to monitor during a scenario run.

| | |
|---|---|
| **To access** | Right-click a graph **>** **Add Measurements**<br>Click **Add** in the **Resource Measurements** section of **<monitor name>** dialog. |
| **Relevant tasks** | "How to Set Up a Monitoring Environment" on page 1171 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Component/Counter Description** | Describes the selected component or counter. |
| **Host** | The name of the monitored machine. |
| **Measured Components** | A hierarchical view of the available components. Browse the tree and select the components you want to monitor. A description of the highlighted component appears in the **Component/Counter Description** box. |
| **Performance Counters** | Select the required performance counters. For details about the default monitor counters, see the relevant reference section for your monitor. |

# <Monitor Name> Dialog Box

This dialog box enables you to add monitored server machines and access the dialogs to configure the measurements and data collection method.

| | |
|---|---|
| **To access** | Right-click a graph **>** **Add Measurements** |
| **Important information** | Before configuring a monitor's measurements, many servers require initial setup. The first step of "How to Set Up a Monitoring Environment" on page 1171 contains links to the setup instructions. |
| **Relevant tasks** | "How to Set Up a Monitoring Environment" on page 1171 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Advanced** | Disabled. |
| **Description** | Displays a description of the selected resource measurement. |
| **Monitored Server Machines** | The machines whose resources are being monitored.<br>Add ... Displays the Add Machine dialog box, which adds the machine that you want to monitor to the existing list.<br>Delete Removes the selected machine from the list. |
| **Resource Measurements on <machine name>** | Displays the resource measurements being monitored on the selected machine.<br>Add ... Displays the Resources dialog box that lets you create a list of resource measurements on the selected machine.<br>Delete Removes the selected resource measurement from the list. |
| **Import / Export** (for SiteScope monitor only) | Allows you to save and restore SiteScope monitor settings to/from .SSM files (XML format). |

# Application Deployment Solution Monitoring

## Application Deployment Solution Monitoring Overview

Using LoadRunner's Application Deployment Solution monitor, you can isolate server performance bottlenecks by monitoring the Citrix server during a scenario run.

LoadRunner's Citrix monitor provides you with information about the application deployment usage of the Citrix server during scenario execution. The Citrix monitor allows you to monitor the server performance statistics from Citrix servers. You can monitor multiple parameters (counters) with a single monitor instance. This allows you to watch server loading for performance, availability, and capacity planning.

To obtain performance data, you need to activate the online monitor for the server and specify which resources you want to measure before executing the scenario.

## How to Set up the Citrix Monitoring Environment

This task describes the working order for setting up the monitoring environment.

1. Prerequisites
   - Make sure that Citrix Server has been installed and is running.

- If Citrix Server machine is running Windows 2000, make sure that the server machine is also running the Remote Registry service.
- Make sure that the LoadRunner machine has administrator privileges to access the Citrix server.
- Measurements that monitor instances are valid for the currently running Citrix session only. If you run this test again, you will need to reconfigure the measurements that are instance-oriented.

  To monitor the different instances, ensure that the server login and logout procedures are recorded in the **Vuser_init** and **Vuser_end** sections respectively, and not in the Action section of the script. For more information, see "Vuser Script Sections" on page 155.

2. Map the Network Drive

   From the Controller machine, map a network drive to the Citrix server machine. This ensures that the required authentication is provided to the Controller to access the resource counters.

3. Launch PerfMon

   Launch PerfMon from the Controller machine to enable the counters on the Citrix server. This allows you to monitor the same counters for the ICA Session object on the Citrix monitor.

4. Open the Connection with the Citrix Server

   You can configure the Citrix monitor to view ICA Session object counters only if at least one session is being run on the Citrix server. If no "real" user has opened a connection with the Citrix server, you need to first initialize or run a Citrix Vuser against the server, and only then configure the Citrix Monitor and add the ICA Session counters. If you configure the Citrix monitor without first initializing or running a Citrix Vuser (or connecting to the Citrix server as a "real" user), you will not be able to view the ICA Session object.

5. Configure the Citrix monitor from the Controller

   For task details, see "How to Set Up a Monitoring Environment" on page 1171.

## Citrix Server Performance Counters

The following sections describe some of the counters that can be measured.

### Non-Virtual Counters

The following table describes non-virtual counters:

| Measurement | Description |
|---|---|
| **% Disk Time** | The percentage of elapsed time that the selected disk drive services read or write requests. |

| | |
|---|---|
| **% Processor Time** | The percentage of time that the processor executes a non-Idle thread. This counter is a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the Idle process in each sample interval, and subtracting that value from 100%. (Each processor has an Idle thread which consumes cycles when no other threads are ready to run.) It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%. |
| **File data Operations/sec** | The rate that the computer issues Read and Write operations to file system devices. This does not include File Control Operations. |
| **Interrupts/sec** | The average number of hardware interrupts the processor receives and services per second. It does not include DPCs, which are counted separately. This value is an indirect indicator of the activity of devices that generate interrupts, such as the system clock, the mouse, disk drivers, data communication lines, network interface cards and other peripheral devices. These devices normally interrupt the processor when they have completed a task or require attention. Normal thread execution is suspended during interrupts. Most system clocks interrupt the processor every 10 milliseconds, creating a background of interrupt activity. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval. |
| **Output Session Line Speed** | This value represents the line speed from server to client for a session in bps. |
| **Input Session Line Speed** | This value represents the line speed from client to server for a session in bps. |
| **Page Faults/sec** | A count of the Page Faults in the processor. A page fault occurs when a process refers to a virtual memory page that is not in its Working Set in main memory. A Page Fault will not cause the page to be fetched from disk if that page is on the standby list, and hence already in main memory, or if it is in use by another process with whom the page is shared. |
| **Pages/sec** | The number of pages read from the disk or written to the disk to resolve memory references to pages that were not in memory at the time of the reference. This is the sum of Pages Input/sec and Pages Output/sec. This counter includes paging traffic on behalf of the system Cache to access file data for applications. This value also includes the pages to/from non-cached mapped memory files. This is the primary counter to observe if you are concerned about excessive memory pressure (that is, thrashing), and the excessive paging that may result. |

| | |
|---|---|
| **Pool Nonpaged Bytes** | The number of bytes in the Nonpaged Pool, a system memory area where space is acquired by operating system components as they accomplish their appointed tasks. Nonpaged Pool pages cannot be paged out to the paging file, but instead remain in main memory as long as they are allocated. |
| **Private Bytes** | The current number of bytes this process has allocated that cannot be shared with other processes. |
| **Processor Queue Length** | The instantaneous length of the processor queue in units of threads. This counter is always 0 unless you are also monitoring a thread counter. All processors use a single queue in which threads wait for processor cycles. This length does not include the threads that are currently executing. A sustained processor queue length greater than two generally indicates processor congestion. This is an instantaneous count, not an average over the time interval. |
| **Threads** | The number of threads in the computer at the time of data collection. Notice that this is an instantaneous count, not an average over the time interval. A thread is the basic executable entity that can execute instructions in a processor. |
| **Latency – Session Average** | This value represents the average client latency over the life of a session. |
| **Latency – Last Recorded** | This value represents the last recorded latency measurement for this session. |
| **Latency – Session Deviation** | This value represents the difference between the minimum and maximum measured values for a session. |
| **Input Session Bandwidth** | This value represents the bandwidth from client to server traffic for a session in bps. |
| **Input Session Compression** | This value represents the compression ratio for client to server traffic for a session. |
| **Output Session Bandwidth** | This value represents the bandwidth from server to client traffic for a session in bps. |
| **Output Session Compression** | This value represents the compression ratio for server to client traffic for a session. |
| **Output Session Linespeed** | This value represents the line speed from server to client for a session in bps. |

## Virtual Channel Counters

The following table describes virtual channel counters:

| Measurement | Description |
|---|---|
| **Input Audio Bandwidth** | This value represents the bandwidth from client to server traffic on the audio mapping channel. This is measured in bps. |
| **Input Clipboard Bandwidth** | This value represents the bandwidth from client to server traffic on the clipboard mapping channel. This is measured in bps. |
| **Input COM1 Bandwidth** | This value represents the bandwidth from client to server traffic on the COM1 channel. This is measured in bps. |
| **Input COM2 Bandwidth** | This value represents the bandwidth from client to server traffic on the COM2 channel. This is measured in bps. |
| **Input COM Bandwidth** | This value represents the bandwidth from client to server traffic on the COM channel. This is measured in bps. |
| **Input Control Channel Bandwidth** | This value represents the bandwidth from client to server traffic on the ICA control channel. This is measured in bps. |
| **Input Drive Bandwidth** | This value represents the bandwidth from client to server traffic on the client drive mapping channel. This is measured in bps. |
| **Input Font Data Bandwidth** | This value represents the bandwidth from client to server traffic on the local text echo font and keyboard layout channel. This is measured in bps. |
| **Input Licensing Bandwidth** | This value represents the bandwidth from server to client traffic on the licensing channel. This is measured in bps. |
| **Input LPT1 Bandwidth** | This value represents the bandwidth from client to server traffic on the LPT1 channel. This is measured in bps. |
| **Input LPT2 Bandwidth** | This value represents the bandwidth from client to server traffic on the LPT2 channel. This is measured in bps. |
| **Input Management Bandwidth** | This value represents the bandwidth from client to server traffic on the client management channel. This is measured in bps. |
| **Input PN Bandwidth** | This value represents the bandwidth from client to server traffic on the Program Neighborhood channel. This is measured in bps. |

| | |
|---|---|
| **Input Printer Bandwidth** | This value represents the bandwidth from client to server traffic on the printer spooler channel. This is measured in bps. |
| **Input Seamless Bandwidth** | This value represents the bandwidth from client to server traffic on the Seamless channel. This is measured in bps. |
| **Input Text Echo Bandwidth** | This value represents the bandwidth from client to server traffic on the local text echo data channel. This is measured in bps. |
| **Input Thinwire Bandwidth** | This value represents the bandwidth from client to server traffic on the Thinwire (graphics) channel. This is measured in bps. |
| **Input VideoFrame Bandwidth** | This value represents the bandwidth from client to server traffic on the VideoFrame channel. This is measured in bps. |
| **Output Audio Bandwidth** | This value represents the bandwidth from server to client traffic on the audio mapping channel. This is measured in bps. |
| **Output Clipboard Bandwidth** | This value represents the bandwidth from server to client traffic on he clipboard mapping channel. This is measured in bps. |
| **Output COM1 Bandwidth** | This value represents the bandwidth from server to client traffic on the COM1 channel. This is measured in bps. |
| **Output COM2 Bandwidth** | This value represents the bandwidth from server to client traffic on the COM2 channel. This is measured in bps. |
| **Output COM Bandwidth** | This value represents the bandwidth from server to client traffic on the COM channel. This is measured in bps. |
| **Output Control Channel Bandwidth** | This value represents the bandwidth from server to client traffic on the ICA control channel. This is measured in bps. |
| **Output Drive Bandwidth** | This value represents the bandwidth from server to client traffic on the client drive channel. This is measured in bps. |
| **Output Font Data Bandwidth** | This value represents the bandwidth from server to client traffic on the local text echo font and keyboard layout channel. This is measured in bps. |
| **Output Licensing Bandwidth** | This value represents the bandwidth from server to client traffic on the licensing channel. This is measured in bps. |
| **Output LPT1 Bandwidth** | This value represents the bandwidth from server to client traffic on the LPT1 channel. This is measured in bps. |
| **Output LPT2 Bandwidth** | This value represents the bandwidth from server to client traffic on the LPT2 channel. This is measured in bps. |

| | |
|---|---|
| **Output Management Bandwidth** | This value represents the bandwidth from server to client traffic on the client management channel. This is measured in bps. |
| **Output PN Bandwidth** | This value represents the bandwidth from server to client traffic on the Program Neighborhood channel. This is measured in bps. |
| **Output Printer Bandwidth** | This value represents the bandwidth from server to client traffic on the printer spooler channel. This is measured in bps. |
| **Output Seamless Bandwidth** | This value represents the bandwidth from server to client traffic on the Seamless channel. This is measured in bps. |
| **Output Text Echo Bandwidth** | This value represents the bandwidth from server to client traffic on the local text echo data channel. This is measured in bps. |
| **Output Thinwire Bandwidth** | This value represents the bandwidth from server to client traffic on the Thinwire (graphics) channel. This is measured in bps. |
| **Output VideoFrame Bandwidth** | This value represents the bandwidth from server to client traffic on the VideoFrame channel. This is measured in bps. |

## Citrix Monitor Dialog Box

This dialog box enables you to configure the measurements for the Citrix monitor.

| | |
|---|---|
| **To access** | Right-click a graph **>** **Add Measurements**<br>Click **Add** in the **Resource Measurements** section of **Citrix Monitor** dialog. |
| **Important information** | **Note:** For Citrix monitoring, if the dialog box freezes after clicking **Add**, you may need to rebuild the localhost cache on the Citrix server machine. |
| **Relevant tasks** | "How to Set up the Citrix Monitoring Environment" on page 1178 |
| **See also** | "How to Set Up a Monitoring Environment" on page 1171 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Add ... | Adds the selected measurement to the list of measurements in the **Measurements on <machine>** section of the Citrix dialog box. |
| **Counters** | Select a resource counter to monitor. Select multiple counters using the CTRL key. For a definition of each counter, click **Explain**. |

| | |
|---|---|
| **Instances** | If multiple instances of the selected counter are running, select one or more instances to monitor for the selected counter. |
| **Object** | Select the object being monitored on the specified machine. |

# Database Server Resource Monitoring

## Database Resource Monitoring Overview

LoadRunner's Database Server Resource monitors measure database resource usage statistics for Oracle or SQL Servers during a scenario run. You use these monitors to isolate database server performance bottlenecks.

The **SQL Server** monitor can be configured as a native LoadRunner monitor, or as a SiteScope monitor.

The **Oracle** monitor displays information from Oracle V$ tables: Session statistics, V$SESSTAT, system statistics, V$SYSSTAT, and other table counters defined by the user in the custom query.

Before defining the monitoring measurements for the Oracle monitor in Controller, you must set up the monitoring environment on the database server:

> **Note:** If there is no application working with a database, you can only monitor the database manager instance.

For details about the Oracle monitor configuration, see "How to Set Up the Oracle Monitoring Environment" below.

You then enable each database resource monitor from Controller by selecting the counters you want the monitor to measure.

## How to Set Up the Oracle Monitoring Environment

This task describes how to set up the monitor environment before monitoring an Oracle database server using the native LoadRunner monitor.

> **Note:** If a problem occurs in setting up the Oracle environment, check the Oracle server to view the error messages.

1. Prerequisites
   - Ensure that the Oracle client libraries are installed on the Controller machine.
   - Verify that **%OracleHome%\bin** is included in the path environment variable. If it is not, add it.
   - Ensure that the registries are updated for the version of Oracle that you are using and that they have the following key: **HKEY_LOCAL_MACHINE\SOFTWARE\ORACLE**
   - Verify that the Oracle server you want to monitor is up and running. Note that it is possible to

monitor several Oracle database servers concurrently.

> **Note:** Only the 32-bit Oracle client should be installed on the Controller machine running the Oracle monitor. If you have a 16-bit and a 32-bit Oracle client installation on the Controller machine, the 16-bit installation should be uninstalled.

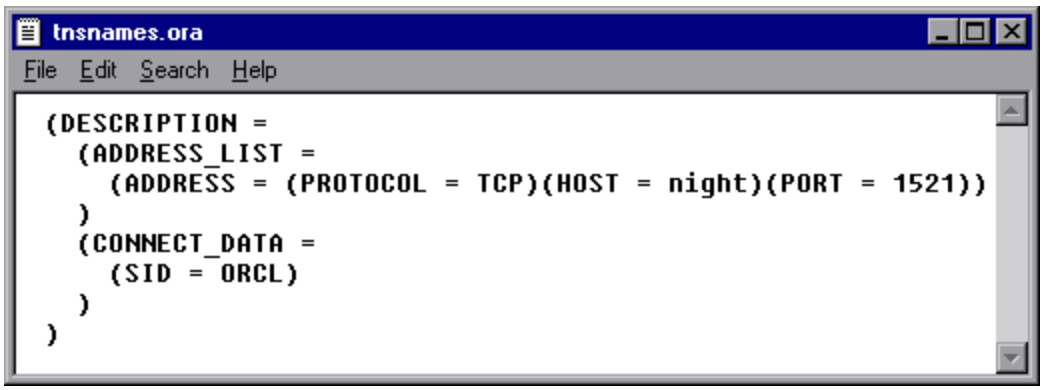
2. Configure the Oracle client/server connection

   Set the connection parameters so the Oracle client (Controller machine) can communicate with the Oracle server(s) you plan to monitor.

   On the Controller machine, set the following configuration parameter either by editing the **tnsnames.ora** file in a text editor, or using the Oracle service configuration tool (for example, **Start > Programs > Oracle for Windows NT > Oracle Net8 Easy Config**):

   - a new service name (TNS name) for the Oracle instance
   - TCP protocol
   - the host name (name of monitored server machine)
   - the port number (usually 1521)
   - the database SID (the default SID is ORCL)

   For example:




3. Connect to the monitored server machine and verify the connection

   a. Obtain a username and password for the service from your database administrator, and ensure that the Controller has database administrator privileges for the Oracle **V$** tables (V$**SESSTAT**, **V$SYSSTAT**, **V$STATNAME**, **V$INSTANCE**, **V$SESSION**).

   b. Verify connection with the Oracle server by performing **tns ping** from the Controller machine.

   > **Note:** There may be a problem connecting if the Oracle server is behind a DMZ/firewall that limits its communication to application servers accessing it.

   c. Run SQL*Plus from the Controller and attempt to log in to the Oracle server(s) with the desired username/password/server combination.

   d. Type **SELECT * FROM V$SYSSTAT** to verify that you can view the V$SYSSTAT table on the

Oracle server. Use similar queries to verify that you can view the V$SESSTAT, V$SESSION, V$INSTANCE, V$STATNAME, and V$PROCESS tables on the server.

4. Modify the monitoring sample rate (optional)

   To change the length of each monitoring sample (in seconds), edit the **dat\monitors\vmon.cfg** file in the LoadRunner root folder. The default rate is 10 seconds.

   The minimum sampling rate for the Oracle Monitor is 10 seconds. If you set the sampling rate at less than 10 seconds, the Oracle Monitor will continue to monitor at 10 second intervals.

5. Configure the Oracle monitor from the Controller

   For task details (beginning with step 2), see "How to Set Up a Monitoring Environment" on page 1171.

## Oracle Performance Counters

The following measurements are most commonly used when monitoring the Oracle server (from the V$SYSSTAT table):

| Measurement | Description |
|---|---|
| **CPU used by this session** | The amount of CPU time (in 10s of milliseconds) used by a session between the time a user call started and ended. Some user calls can be completed within 10 milliseconds and, as a result, the start and end-user call time can be the same. In this case, 0 milliseconds are added to the statistic. A similar problem can exist in the operating system reporting, especially on systems that suffer from many context switches. |
| **Bytes received via SQL*Net from client** | The total number of bytes received from the client over Net8. |
| **Logons current** | The total number of current logons |
| **Opens of replaced files** | The total number of files that needed to be reopened because they were no longer in the process file cache. |
| **User calls** | Oracle allocates resources (Call State Objects) to keep track of relevant user call data structures every time you log in, parse, or execute. When determining activity, the ratio of user calls to RPI calls gives you an indication of how much internal work is generated as a result of the type of requests the user is sending to Oracle. |
| **SQL*Net roundtrips to/from client** | The total number of Net8 messages sent to, and received from, the client. |

| | |
|---|---|
| **Bytes sent via SQL*Net to client** | The total number of bytes sent to the client from the foreground process(es). |
| **Opened cursors current** | The total number of current open cursors. |
| **DB block changes** | Closely related to consistent changes, this statistic counts the total number of changes that were made to all blocks in the SGA that were part of an update or delete operation. These are changes that generate redo log entries and hence will cause permanent changes to the database if the transaction is committed. This statistic is a rough indication of total database work and indicates (possibly on a per-transaction level) the rate at which buffers are being dirtied. |
| **Total file opens** | The total number of file opens being performed by the instance. Each process needs a number of files (control file, log file, database file) to work against the database. |

## SQL Server Performance Counters

The following table describes the default counters that can be monitored on version 6.5 of the SQL Server:

| Measurement | Description |
|---|---|
| **% Total Processor Time** | The average percentage of time that all the processors on the system are busy executing non-idle threads. On a multi-processor system, if all processors are always busy, this is 100%, if all processors are 50% busy this is 50% and if 1/4 of the processors are 100% busy this is 25%. It can be viewed as the fraction of the time spent doing useful work. Each processor is assigned an Idle thread in the Idle process which consumes those unproductive processor cycles not used by any other threads. |
| **% Processor Time** | The percentage of time that the processor is executing a non-idle thread. This counter was designed as a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the idle process in each sample interval, and subtracting that value from 100%. (Each processor has an idle thread which consumes cycles when no other threads are ready to run). It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%. |

| | |
|---|---|
| **Cache Hit Ratio** | The percentage of time that a requested data page was found in the data cache (instead of being read from disk). |
| **I/O - Batch Writes/sec** | The number of 2K pages written to disk per second, using Batch I/O. The checkpoint thread is the primary user of Batch I/O. |
| **I/O - Lazy Writes/sec** | The number of 2K pages flushed to disk per second by the Lazy Writer. |
| **I/O - Outstanding Reads** | The number of physical reads pending. |
| **I/O - Outstanding Writes** | The number of physical writes pending. |
| **I/O - Page Reads/sec** | The number of physical page reads per second. |
| **I/O - Transactions/sec** | The number of Transact-SQL command batches executed per second. |
| **User Connections** | The number of open user connections. |

# Dynatrace Monitor

The Dynatrace monitor provides information from the Dynatrace server about monitored applications.

## Dynatrace dashboards and dashlets

Using Dynatrace's dashboards and dashlets you can get insights into your full application stack, in the form of deep transaction tracing, synthetic monitoring, real user monitoring, and network monitoring.

A dashboard represents a specific view of one or more Dynatrace data sources. A dashboard is comprised of dashlets, the building blocks of a dashboard that contain specific types of data (in the form of a graph or data tables).

> **Note:** Only dashlets that are represented as graphs in Dynatrace are available in Performance Center during the performance test run.

## Set up the Dynatrace monitor

1. **Prerequisites:** The Dynatrace monitor must be installed on the AUT server.
2. In Controller, in the **Run** tab, add the Dynatrace monitor. **Available graphs** > **Dynatrace graphs** > **Dynatrace**.
3. Right-click the Dynatrace monitor and click **Add measurements**.
4. In the Dynatrace dialog box, click **Add** to add the monitor, and enter the Dynatrace server details:

> **Note:** In order to connect to a Dynatrace server with specific credentials, you must first stop the Dynatrace monitor and remove all of its counters.

| UI Element | Description |
|---|---|
| **Name** | The name of the Dynatrace server. |
| **Port** | The Dynatrace port.<br>• HTTP: 8020<br>• HTTPS: 8021 (Default) |
| **Use Secure HTTP** | Uses a secure HTTP connection to the Dynatrace server. |
| **User Name** | Instructs LoadRunner to use the name of the Dynatrace account. |
| **Password** | Instructs LoadRunner to use the password for the Dynatrace account. |

Click **OK**.

For detailed information on the performance measurements that Dynatrace can monitor, refer to the Dynatrace documentation.

### See also:

- "How to Set Up a Monitoring Environment" on page 1171
- "Add Machine Dialog Box" on page 1175

# Flex Monitoring Graphs

The Flex Monitoring graphs track the same measurements for online monitoring and post-run analysis.

See the documentation in the Analysis section: "Flex Graphs" on page 1412.

- "Flex RTMP Connections Graph" on page 1413
- "Flex RTMP Throughput Graph" on page 1412
- "Flex RTMP Other Statistics Graph" on page 1413

- "Flex Streaming Delivery Graph" on page 1414
- "Flex Average Buffering Time Graph" on page 1415

# Infrastructure Resources Monitoring

## Infrastructure Resources Monitoring Overview

Using LoadRunner's Network Client monitor, you can monitor network client resources for FTP, POP3, SMTP, IMAP, and DNS Vusers during a scenario run and isolate client performance bottlenecks.

The Network Client online monitor graph is only available during scenarios that run relevant scripts, such as FTP, POP3, and so forth.

You can view this graph by dragging it from the Infrastructure Resources Graph section in the graph tree into the right pane of the Run view. The graph appears in the graph view area.

## Network Client Performance Counters

The following table describes the Network Client measurements that are monitored:

| Measurement | Description |
|---|---|
| **Pings per sec** | Number of pings per second |
| **Data transfer bytes per sec** | Number of data bytes transferred per second |
| **Data receive bytes per sec** | Number of data bytes received per second |
| **Connections per sec** | Number of connections per second |
| **Accept connections per sec** | Number of connections accepted per seconds |
| **SSL Connections per sec** | Number of SSL connections per second |
| **SSL Data transfer bytes per sec** | Number of SSL data bytes transferred per second |
| **SSL Data receive bytes per sec** | Number of SSL data bytes received per second |
| **SSL Accept connections per sec** | Number of SSL connections accepted per seconds |

# JMeter Monitoring Graphs

The JMeter monitoring graphs display data for the JMeter test in a scenario run.

The Backend Listener is able to receive online measurements from JMeter tests.

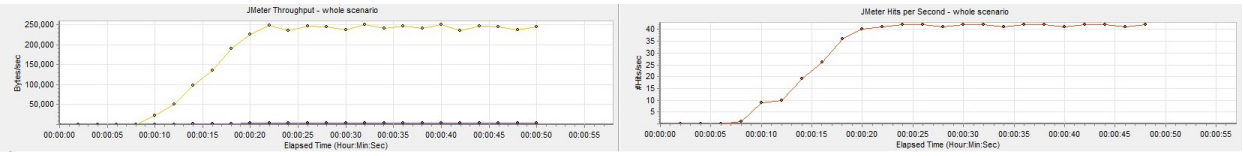
**Note:**

- If you do not see any data in the graphs, make sure that the **Start measurement** check box is selected in **Runtime Settings**.
- Best practices is to use online measurements (**Start measurement** check box selected) when creating and testing the load test. For the best performance, it is recommended to turn the measurements off for an actual load testing.
- All thread groups from a single JMeter test are executed and displayed as one group, regardless of the actual number of thread groups.
- You can view more data for JMeter tests in the Running Vusers and Transaction Monitor graphs. For details, see "Runtime and Transaction Monitoring" on page 1217.

| Monitor | Description |
|---|---|
| **JMeter Hits per Second** | Shows the number of hits (y-axis) per second (x-axis) generated by the test plan.<br>It does not include child samples from transactions and embedded resources. |
| **JMeter Throughput** | Shows the amount of throughput (y-axis) during each second of the test run (x-axis). Throughput is measured in bytes and represents the amount of bytes sent and received by JMeter during the load test. |

The default interval used in the graphs is 5 seconds. You can change the default interval in Controller: Select **Tools > Options > Monitors > Frequency**.

### Example




### See also:

- "JMeter Tests" on page 945
- "JMeter Analysis Graphs" on page 1446

# Middleware Performance Monitoring

A primary factor in a transaction's response time is the Middleware performance usage. LoadRunner's Middleware Performance monitors provide you with information about the Middleware performance

usage of the IBM WebSphere MQ server during a scenario execution. To obtain performance data, you need to activate the online monitor for the server and specify which resources you want to measure before executing the scenario.

The IBM WebSphere MQ monitor is used to monitor channel and queue performance counters on an IBM WebSphere MQ Server.

## How to Set Up the IBM WebSphere MQ Monitor

This task describes how to configure the Controller and IBM WebSphere MQ machines:

1. Prerequisites

   Ensure that an IBM WebSphere MQ Client Connection is installed on the Controller machine.

2. Configure the server environment to monitor events

   The LoadRunner MQ Monitor retrieves event messages from two standard MQSeries queues only:

   - SYSTEM.ADMIN.PERFM.EVENT – performance events, such as "queue depth high"
   - SYSTEM.ADMIN.CHANNEL.EVENT – channel events, such as "channel stopped"

   Events must be enabled for the queue manager (and in many cases, on the applicable object, as well). Performance events are enabled by setting attributes for the queue on the MQ Server. Channel events are enabled by default, and cannot be disabled.

   > **Note:** The IBM WebSphere MQ monitor does not retrieve data from a queue manager after the queue manager has been restarted.

   a. Run the following MQSC command:
   `ALTER QMGR PERFMEV(ENABLED)`.
   b. Set the queue attributes. For a list of queue attributes, "IBM WebSphere MQ Queue Attributes" on page 1196.

3. Add the monitored server to the Controller
   a. In the Controller Run view, click the IBM WebSphere MQ graph in the graph tree, and drag it into the right pane.
   b. Right-click the graph and select **Add Measurements**, or click anywhere on the graph and select **Monitors > Add Measurements**. The IBM WebSphere MQ dialog box opens.

      In the **Monitored Server Machines** section, click **Add.** The Add Machine dialog box opens.
   c. The first time that you add measurements, enter the server name or IP address of the machine you want to monitor. The format of the server name is `<machine name>:<port number>`. Select the platform on which the machine runs, and click **OK**.
   d. In the **Resource Measurements** section of the IBM WebSphere MQ dialog box, click **Add**.
4. Configure the IBM WebSphere MQ monitor

   The IBM WebSphere MQ monitor connects to the IBM WebSphere MQ server (via the MQ Client Connection installed on the Controller machine). In MQ Client environments, the client machine

connects to an MQ Server instance, and uses the Server's resources as if they were local to the client machine.

Specify the connection information and measurements in the MQ Monitor Add Measurements dialog.

# IBM WebSphere MQ Performance Counters

## Queue Performance Counters

The following table describes the Queue Performance counters:

| Measurement | Description |
|---|---|
| **Event - Queue Depth High (events per second)** | An event triggered when the queue depth reaches the configured maximum depth. |
| **Event - Queue Depth Low (events per second)** | An event triggered when the queue depth reaches the configured minimum depth. |
| **Event - Queue Full (events per second)** | An event triggered when an attempt is made to put a message on a queue that is full. |
| **Event - Queue Service Interval High (events per second)** | An event triggered when no messages are put to or retrieved from a queue within the timeout threshold. |
| **Event - Queue Service Interval OK (events per second)** | An event triggered when a message has been put to or retrieved from a queue within the timeout threshold. |
| **Status - Current Depth** | Current count of messages on a local queue. This measurement applies only to local queues of the monitored queue manager. |
| **Status - Open Input Count** | Current count of open input handles. Input handles are opened so that an application may "put" messages to a queue. |
| **Status - Open Output Count** | Current count of open output handles. Output handles are opened so that an application may "get" messages from a queue. |

## Channel Performance Counters

The following table describes the Channel Performance counters:

| Measurement | Description |
|---|---|

| | |
|---|---|
| **Event - Channel Activated (events per second)** | Event generated when a channel, waiting to become active but inhibited from doing so due to a shortage of queue manager channel slots, becomes active due to the sudden availability of a channel slot. |
| **Event - Channel Not Activated (events per second)** | Event generated when a channel, attempts to become active but inhibited from doing so due to a shortage of queue manager channel slots. |
| **Event - Channel Started (events per second)** | Event generated when a channel is started. |
| **Event - Channel Stopped (events per second)** | Event generated when a channel is stopped, regardless of source of stoppage. |
| **Event - Channel Stopped by User (events per second)** | Event generated when a channel is stopped by a user. |
| **Status - Channel State** | The current state of a channel. Channels pass through several states from stopped (inactive state) to running (fully active state). Channel states range from 0 (stopped) to 6 (running). |
| **Status - Messages Transferred** | The count of messages that have been sent over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available. |
| **Status - Buffer Received** | The count of buffers that have been received over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available. |
| **Status - Buffer Sent** | The count of buffers that have been sent over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available. |
| **Status - Bytes Received** | The count of bytes that have been received over the channel. If no traffic is occurring over the channel, this measurement will appear as zero. If the channel has not been started since the queue manager was started, no measurement will be available. |

| | |
|---|---|
| **Status - Bytes Sent** | The count of bytes that have been sent over the channel. If no traffic is occurring over the channel, this measurement will appear as zero. If the channel has not been started since the queue manager was started, no measurement will be available. |

## IBM WebSphere MQ Queue Attributes

You set the following queue attributes using the MQSC command `ALTER QMGR PERFMEV(ENABLED)`:

| Measurement | Set Event Attributes |
|---|---|
| **Event - Queue Depth High** | • **QDEPTHHI(integer)** - where integer is a value expressed as a percentage of maximum messages allowed, and is in the range of 0 to 100 inclusive.<br>• **QDPHIEV(action) -** where action is the word "ENABLED" or "DISABLED", enabling or disabling the generation of the event, respectively. |
| **Event - Queue Depth Low** | To enable the event for a queue, the following attributes of the queue must be set:<br>• **QDEPTHLO(integer) -** where integer is a value expressed as a percentage of maximum messages allowed, and is in the range of 0 to 100 inclusive.<br>• **QDPLOEV(action) -** where action is the word "ENABLED" or "DISABLED", enabling or disabling the generation of the event, respectively. |
| **Event - Queue Full** | • **QDEPTHHI(integer)** – where integer is a value expressed as a percentage of maximum messages allowed, and is in the range of 0 to 100 inclusive.<br>• **QDPMAXEV(action)** – where action is the word "ENABLED" or "DISABLED", enabling or disabling the generation of the event, respectively. |
| **Event - Queue Service Interval High** | • **QSVCINT(integer)** – where integer is a value expressed as milliseconds, in the range of 0 and 999,,999, inclusive. Note: this value is shared with Queue Service Interval OK.<br>• **QSVCIEV(type)** – where type is the word "HIGH", "OK", or "NONE", enabling service interval high events, enabling service interval ok events, or disabling the generation of the event, respectively. |
| **Event - Queue Service Interval OK** | • **QSVCINT(integer)** – where integer is a value expressed as milliseconds, in the range of 0 and 999,999,999, inclusive. Note: this value is shared with Queue Service Interval High.<br>• **QSVCIEV(type)** – where type is the word "HIGH", "OK", or "NONE", enabling service interval high events, enabling service interval ok events, or disabling the generation of the event, respectively. |

# MQ Monitor Add Measurements Dialog Box

This dialog box enables you to configure the monitor by choosing which measurements to monitor on the machine.

| | |
|---|---|
| **To access** | Right-click a graph **>** **Add Measurements**<br>Click **Add** in the **Resource Measurements** section of the **IBM WebSphere MQ** dialog. |
| **Important information** | User entries for any text box are limited to 48 characters. |
| **Relevant tasks** | "How to Set Up the IBM WebSphere MQ Monitor" on page 1193 |
| **See also** | "IBM WebSphere MQ Performance Counters" on page 1194 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Alternate Queue** | If the event configured for monitoring is from a remote queue manager (other than the one identified in the queue manager field of the IBM WebSphere MQ Add Measurements dialog box), click **Alternate Queue** to enter the name of an alternate queue manager.<br><br>**Note:** When you add an alternate queue manager, this becomes the default queue manager for any events that you subsequently add. To return to the queue manager to which you are connected, enter that name in the Alternate Queue Manager dialog box. |
| **Available Measurements** | **Object Type.**Select an object type from either Channel or Queue.<br>**Object Name.** Enter a name for object you want to monitor.<br>**Event/Attribute.** Select the events and attributes you want to monitor for the selected object.<br>**Filter System Objects.** Select to enable the system objects filter.<br>**Add Object.** Enables you to add a new object name to the Object name list.<br>**Add.** Enables you to add an Event or Attribute to an object.<br>**Remove.** Enables you to remove a monitored object event or attribute from the Object name list.<br>**Alternate Queue.** Enter the name of an alternate queue manager if the event is from a remote queue manager. |

| | |
|---|---|
| **Connections Information** | **Server.** The name of the server you are monitoring.<br>**Client Channel.** Enter the name of the channel through which a client connection is made to an MQ Server.<br>• **Note:** You can set up a specific channel on an MQ Server instance, or use the default "SYSTEM.DEF.SVRCONN" channel. If the client channel is undefined, the MQ Server will be inaccessible via client connections (the MQ Monitor will not work, as it will not be able to connect to the queue manager which it is supposed to monitor).<br>**Queue Manager.** Enter the name of the queue manager to be monitored.<br>• **Note:** The monitor is not restricted to monitoring only the queue manager to which it is connected. You can configure multiple queue managers to write to the event queue of a central queue manager for centralized monitoring (this applies to Events only, not polled object attributes). All events contain a queue manager attribute identifying their source.<br>• **Note:** A queue manager can only be accessed by one Controller or monitoring application at any one time. |
| **Filter system objects** | By default, only user-defined objects are displayed in the **Object name** list. To show all objects, clear the **Filter System Objects** check box. You can modify the filter settings, in the **<LoadRunner root>\dat\monitors\mqseries.cfg** file. |
| **Monitored Object list** | A list of monitored objects, including the object's name, events and attributes, and alternate queue manager. |

# MQTT Statistics Monitoring

MQTT is a simple, lightweight, publish/subscribe messaging protocol designed for constrained devices and low-bandwidth, high-latency or unreliable networks. MQTT is ideal for machine-to-machine (M2M) and Internet of Things (IoT) communications, as well as for mobile applications where bandwidth and battery power are at a premium. MQTT focuses on minimizing network bandwidth and device resource requirements, while attempting to ensure reliability and some degree of assurance of delivery.

## MQTT Statistics monitors

| Monitor | Description |
|---|---|
| **MQTT Message Rate** | Monitors the number of messages published (**Publish rate**) and received (**Receive rate**) per second at any given time during the load test scenario run. |
| **MQTT Throughput** | Monitors the amount of data (in bytes) that the Vusers received from (**Receipt throughput**) or published to (**Publish throughput**) the MQTT broker at any given time during the load test scenario run. |
| **MQTT Active Connection** | Monitors the number of active MQTT broker connections (**Active connections**) at any given time during the load test scenario run. |
| **MQTT In-flight Messages** | Monitors the number of in-flight messages (**In-flight messages**) on the MQTT broker at any given time during the load test scenario run. In-flight messages are QoS 1 or 2 messages that can be in the process of being transmitted, including messages currently going through handshakes and messages that are being retried. |
| **MQTT $SYS Topics** | Monitors the MQTT broker machine's system resource usage during the load test scenario run. This can help isolate server performance bottlenecks.<br>In addition to the usual $SYS measurements monitored, see custom system measurements monitored. |

## MQTT Statistic Measurements

The following are the default counters that can be monitored on the MQTT Statistics monitors:

| Measurement | Description |
|---|---|
| **Publish rate** | The messages published per second over the duration of the scenario run. |
| **Receipt rate** | The messages received per second over the duration of the scenario run. |
| **Active connections** | The active connections to the MQTT broker over the duration of the scenario run. |
| **Publish throughput** | The average data traffic (in bytes per second) from published messages over the duration of the scenario run. |
| **Receipt throughput** | The average data traffic (in bytes per second) from received messages over the duration of the scenario run. |
| **In-flight messages** | The number of current in-flight messages over the duration of the scenario run. |

| Measurement | Description |
|---|---|
| **Average publish duration** | The average amount of time (in seconds) to publish a message over the duration of the scenario run. |
| **Average connect duration** | The average duration (in seconds) that the broker was connected over the duration of the scenario run. |
| **Connect statistics** | Successful vs failed broker connections. |

### Custom system measurements monitored on the MQTT broker machine ($SYS/broker/):

| Measurement | Description |
|---|---|
| **clients/active** | The number of currently connected clients. (deprecated) |
| **clients/expired** | The number of disconnected persistent clients that have expired and have been removed through the persistent_client_expiration option. |
| **clients/inactive** | The total number of persistent clients (with clean session disabled) that are registered at the broker but are currently disconnected. (deprecated) |
| **load/bytes/...** | The moving average of the number of bytes received / sent by the broker over different time intervals (1/5/15 min).<br>The value returned represents the number of bytes received / sent in 1 min, averaged over 1/5/15 min. |
| **load/connections/...** | The moving average of the number of CONNECT packets received by the broker over different time intervals (1/5/15 min).<br>The value returned represents the number of connections received in 1 min, averaged over 1/5/15 min. |
| **load/messages/...** | The moving average of the number of all types of MQTT messages received / sent by the broker over different time intervals (1/5/15 min).<br>The value returned represents the number of messages received / sent in 1 min, averaged over 1/5/15 min. |
| **load/publish/sent/...** | The moving average of the number of publish messages sent by the broker over different time intervals (1/5/15 min).<br>The value returned represents the number of publish messages sent in 1 min, averaged over 1/5/15 min. |

| Measurement | Description |
|---|---|
| **load/sockets/...** | The moving average of the number of socket connections opened to the broker over different time intervals (1/5/15 min).<br><br>The value returned represents the number of socket connections in 1 min, averaged over 1/5/15 min. |
| **messages/stored** | The number of messages currently held in the message store. This includes retained messages and messages queued for durable clients. |
| **publish/bytes/...** | The total number of bytes received / sent in PUBLISH messages since the broker started. |

See also:

- "MQTT Statistics Graphs" on page 1450
- "MQTT Protocol" on page 605

# Network Delay Monitoring

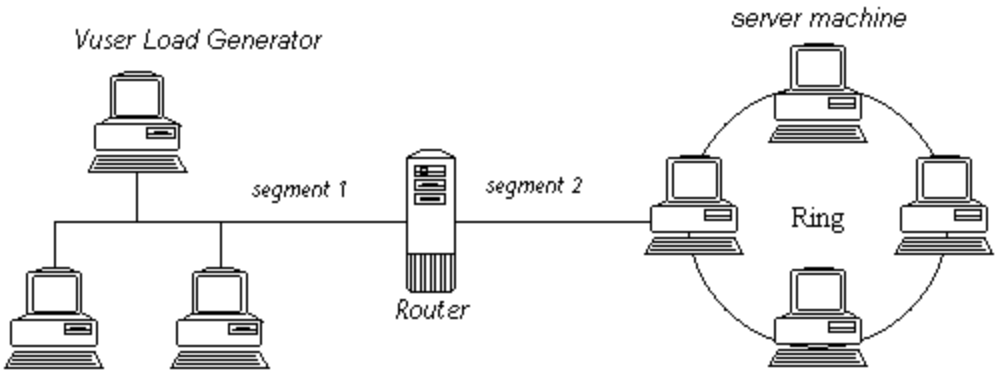
## Network Monitoring Overview

Network configuration is a primary factor in the performance of applications. A poorly designed network can slow client activity to unacceptable levels.

You use Network monitoring to determine whether your network is causing a delay in the scenario. You can also determine the problematic network segment.

In a true Web or client/server system, there are many network segments. A single network segment with poor performance can affect the entire system.

The following diagram shows a typical network. To go from the server machine to the Vuser machine, data must travel over several segments.



The Network Delay Time monitor shows the delays for the complete path between the source and destination machines (for example the database server and Vuser host). The graph maps the delay as a function of the elapsed scenario time. Each defined path is represented by a separate line with a different color in the graph.

To measure network performance, the Network monitor sends packets of data across the network. When a packet returns, the monitor calculates the time it takes for the packet to go to the requested node and return. This time is the delay which appears in the Network Delay Time graph.

> **Note:** The delays from the source machine to each of the nodes are measured concurrently, yet independently. It is therefore possible that the delay from the source machine to one of the nodes could be greater than the delay for the complete path between the source and destination machines.

## How to Set Up the Network Monitoring Environment

This task describes how to prepare your environment for network monitoring.

1. Prerequisites

   To enable network monitoring, you must install the LoadRunner agent on the source machine. You do not have to install the LoadRunner agent on the destination machine.

   To run the Network monitor, you must have administrator privileges on the Windows source machine (unless you are using the ICMP protocol).

2. Configure the Linux source machine - optional

   You can run the Network monitor on Linux source machines, using UDP or ICMP. Before running the Network monitor from a Linux source machine, configure the source machine. For task details, see "How to Configure the Linux Source Machine for Network Monitoring" on the next page.

3. Configure the firewall between the source and destination machines - Optional

   If you are monitoring a network in which there are firewalls between the source and the destination machines, you must configure the firewalls to allow the network data packets to reach their destinations.

   - If you are using the TCP protocol, the firewall that protects the destination machine should not block outgoing ICMP_TIMEEXCEEDED packets (packets that are sent outside the firewall from the machine). In addition, the firewall protecting the source machine should allow ICMP_TIMEEXCEEDED packets to enter, as well as TCP packets to exit.
   - If you are using the ICMP protocol, the destination machine's firewall should not block incoming ICMP_ECHO_REQUEST packets, or outgoing ICMP_ECHO_REPLY and ICMP_ECHO_TIMEEXCEEDED packets. In addition, the firewall protecting the source machine should allow ICMP_ECHO_REPLY and ICMP_ECHO_TIMEEXCEEDED packets to enter, and ICMP_ECHO_REQUEST packets to exit.
   - If you are using the UDP protocol, ensure that the UDP protocol can access the destination machine from the source machine. The destination machine's firewall should not block outgoing

ICMP_DEST_UNREACHABLE and ICMP_ECHO_TIMEEXCEEDED packets. In addition, the firewall protecting the source machine should allow ICMP_DEST_UNREACHABLE and ICMP_ECHO_TIMEEXCEEDED packets to enter.

> **Note:** To run the Network Delay monitor when there are firewalls between the Controller and the source machine, you must configure the LoadRunner agent, MI Listener, and Network Delay monitor for monitoring over a firewall.

4. Specify the network monitor paths

   In the Controller **Run** tab graph tree view, select the **Network Delay Time** graph and drag it into the right pane. Right-click the graph and select **Add Measurements**. Define the paths using the following three dialog boxes:

   a. Add source and destinations machines. For details, see the "Network Delay Time Dialog Box" on page 1205.

   b. Define the network monitor path. For details, see the "Adding Destination Machines for Network Delay Monitoring Dialog Box" on page 1205.

   c. Configure the monitor settings for the defined path. For details, see the "Network Monitor Settings for Defined Path Dialog Box" on page 1207.

## How to Configure the Linux Source Machine for Network Monitoring

This task describes how to configure a Linux source machine before running the network monitor.

1. Assign permissions where LoadRunner is installed locally.

   Follow these steps to assign root permissions to the **merc_webtrace** process:

   a. Log in to the source machine as root.

   b. Type: **cd <LoadRunner root>/bin** to change to the **bin** folder.

   c. Type: **chown root merc_webtrace** to make the root user the owner of the **merc_webtrace** file.

   d. Type: **chmod +s merc_webtrace** to add the s-bit to the file permissions.

   e. To verify, type **ls -l merc_webtrace**. The permissions should look like this: **-rwsrwsr-x**.

2. Assign permissions where LoadRunner is installed on the network.

   In a LoadRunner network installation, the **merc_webtrace** process is on the network, not on the source machine disk. The following procedure copies the **merc_webtrace** file to the local disk, configures **mdrv.dat** to recognize the process, and assigns root permissions to **merc_webtrace**:

   a. Copy **merc_webtrace** from **<LoadRunner root>/bin** to anywhere on the local disk of the source machine. For example, to copy the file to the **/local/<LoadRunner>** folder, type:

   ```
   cp /net/tools/LoadRunner_installation/bin/merc_webtrace /local/<LoadRunner>
   ```

   > **Note:** All of the source machines that use the same network installation must copy

> **merc_webtrace** to the identical folder path on their local disk (for example, `/local/<LoadRunner>`), since all of them use the same **mdrv.dat**.

b. Add the following line to the **<LoadRunner root>/dat/mdrv.** file, in the `[monitors_server]` section:

```
ExtCmdLine=-merc_webtrace_path /local/xxx
```

c. Log in to the source machine as root.

d. Type: `cd LoadRunner root/bin` to change to the **bin** folder.

e. Type: `chown root merc_webtrace` to make the root user the owner of the **merc_webtrace** file.

f. Type: `chmod +s merc_webtrace` to add the s-bit to the file permissions.

g. To verify, type `ls -l merc_webtrace`. The permissions should look like: `-rwsrwsr-x`.

3. Connect to the Linux Source Machine through RSH

Follow these instructions if the Controller is connected to the source machine through RSH (default connection mode). In this case you do not need to activate the agent daemon.

Before running the Network monitor the first time, you enter an encrypted user name and password in the Network monitor configuration file.

a. From the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > Password Encoder**. The Password Encoder window opens.

b. In the **Password** box, type your RSH user name and password, separated by a vertical bar symbol. For example, `myname|mypw`.

c. Click **Generate**. An encoded string is displayed in the Encoded string field.

d. Click **Copy** to copy the encoded string to the clipboard.

e. Add the following line to the **<LoadRunner root>/dat/monitors/ndm.cfg** file, in the `[hosts]` section:

```
Host = <encrypted string copied from clipboard>
```

f. Close and open the current scenario. LoadRunner will read the updated configuration file and recognize the source machine for monitoring.

4. Connect to the Linux Source Machine through the Agent

Follow these instructions for activating agent daemon on the source machine if the Controller is not connected to the source machine through RSH.

a. Type `m_daemon_setup -install` from the **<LoadRunner root>/bin** folder.

b. Make sure that the agent daemon is running whenever you activate the Network monitor.

c. To stop the Network Delay Monitor agent daemon, type `m_daemon_setup -remove`.

## Adding Destination Machines for Network Delay Monitoring Dialog Box

This dialog box enables you to add destination machines for network delay monitoring, and configure additional network monitor settings.

| | |
|---|---|
| **To access** | **Network Delay Time** dialog box **>** **To machine(s)** section **>** click **Add** |
| **Important information** | The Network Delay Time Monitor cannot be configured to work in TCP mode on Windows XP SP2 or Vista. |
| **Relevant tasks** | "How to Set Up the Network Monitoring Environment" on page 1202 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Add ... | Enter the name or URL of the machine at the final destination of the path you want to monitor in the New Machine Name dialog box. Repeat this for each path you want to monitor.<br>**Note:** If the destination machine is **localhost**, enter the local machine's name and not **localhost**. |
| Delete | Deletes the destination machine, to remove this path from the monitor graph. |
| Rename | Renames the destination machine. |
| Properties | Opens the Configuring Network Monitor Settings for Defined Path dialog box. |
| **From Machine** | Displays the name of the source machine. |
| **To Machines** | Displays the names or URLs of the destination machines. |

## Network Delay Time Dialog Box

This dialog box enables you to select the network path you want to monitor.

| | |
|---|---|
| **To access** | Right-click the Network Delay Time graph and select **Add Measurements**.<br>This dialog appears only when you add measurements for the first time. |
| **Important information** | To run the Network monitor, you must have administrator privileges on the source machine (unless you are using the ICMP protocol). |

| | |
|---|---|
| **Relevant tasks** | "How to Set Up the Network Monitoring Environment" on page 1202 |

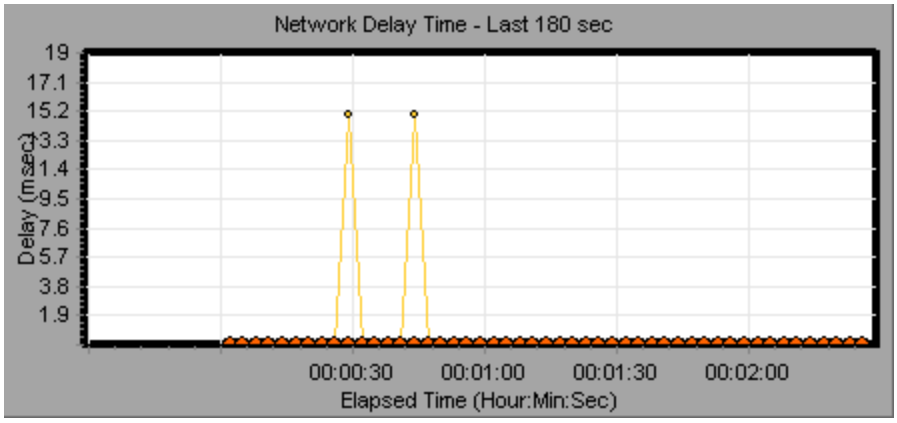
User interface elements are described below:

| UI Element | Description |
|---|---|
| **Monitor the network delay from machine** | Displays the name of the machine from which network monitoring begins (source machine).<br>To add a machine, click Add ... and specify the server name or IP address and machine platform.<br>Repeat this for each path you want to monitor.<br>**Important:** If there is a firewall between the Controller machine and the source machine, enter the server name or IP address of the source machine according to the following format:<br>`<MI Listener machine>:<source machine local key>`<br>where source machine local key is the Local Machine Key that you chose when configuring the LoadRunner agent on the source machine. (See "Agent Configuration Settings Dialog Box" on page 1148)<br>**Example:** 12..12.3:vds |
| **To machine (s)** | Displays the network path in the format of **sourcemachine -> destination machine**.<br>To add a new destination machine, click Add ... and define the machine in the Adding Destination Machines for Network Delay Monitoring dialog box. |

## Network Delay Time Graph

The Network Delay Time graph shows the delay for the complete path between the source and destination machines (y-axis) as a function of the elapsed scenario time (x-axis).

Each path defined in the Add Destination Machines for Network Delay Monitoring dialog box is represented by a separate line with a different color in the graph.



# Network Monitor Settings for Defined Path Dialog Box

This dialog box enables you to set the network protocol, port, monitoring frequency, and monitoring packet retries.

| | |
|---|---|
| **To access** | **Add Destination Machines for Newtork Delay Monitoring >** click **Properties.** |
| **Relevant tasks** | "How to Set Up the Network Monitoring Environment" on page 1202 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Monitor Settings** | • **Send request using X protocol.** Select the network protocol you want the monitor to use: TCP, UDP, or ICMP. It is recommended that you use the default protocol. The default in Windows is TCP, and in Linux is UDP.<br>**Note:** When you use TCP or UDP protocols, administrator privileges are required on the source machine.<br>• **Send request to port.** Enter the port number to be used by the network path.<br>• **Enable display of network nodes by DNS names.** Enables you to view the DNS name of each node along the network path, in addition to its IP address.<br>**Note:** Selecting this option will decrease the speed of the Network monitor. |
| **Monitoring Frequency** | **Send next packet X milliseconds after receipt of previous packet.** Select the number of milliseconds the monitor should wait between receiving a packet and sending out the next packet.<br>**Default:** 3000 milliseconds.<br>**Note:** If you have a long, steady scenario, you can increase the interval by several seconds. |

| | |
|---|---|
| **Monitoring Packet Retries** | • **Wait X seconds for packet to return before retrying.** Select the maximum number of seconds that the monitor should wait for a packet to return before it retries to send the packet.<br>**Default:** 3 seconds.<br>**Note:** If your network is very large and loaded (an internet connection with a low capacity), you should increase the value by several seconds. If you have a small network (such as a LAN), you can decrease the value.<br>• **Number of retries.** Select the number of times the monitor should try resending a packet to a node if the packet is not initially returned.<br>**Default:** 0. |

## Troubleshooting and Limitations - Network Delay Monitor

This section describes troubleshooting for the Network Delay monitor.

If monitoring is unsuccessful and LoadRunner cannot locate the source or destination machines, make sure that the specified machines are available to your machine. Perform a "ping" operation.

At the command line prompt, type `ping server_name`.

To check the entire network path, use the trace route utility to verify that the path is valid.

For Windows, type `tracert <server_name>`.

For Linux, type `traceroute <server_name>`.

If the monitoring problem persists once you verify that the machines are accessible and that the network path is valid, perform the following procedures:

1. If you are using the TCP protocol, run **<**LoadRunner **root folder>\bin\webtrace.exe** from the source machine to determine whether the problem is related to the Controller, or the WebTrace technology on which the Network Delay monitor is based. If you are using the UDP or ICMP protocols, the problem must be related to the Controller and not WebTrace, since these protocols are not WebTrace technology-based.
2. If you receive results by running **webtrace.exe**, the problem is related to the Controller. Verify that the source machine is not a Linux machine, and contact the Customer Support Web site with the following information:
   - the Controller log file, **drv_log.txt**, located in the **temp** folder of the Controller machine.
   - the **traceroute_server** log file, located on the source machine.
   - the debug information located in the **TRS_debug.txt** and **WT_debug.txt** files in the path folder. These files are generated by adding the following line to the [monitors_server] section of the **<**LoadRunner **root folder>\dat\mdrv.** file, and rerunning the Network monitor:

     ```
     ExtCmdLine=-traceroute_debug path
     ```

3. If you do not receive results by running **webtrace.exe**, the problem is related to the WebTrace technology, on which the Network Delay monitor is based. Perform the following procedures on the source machine:
   - Verify that the **packet.sys** file (the Webtrace driver) exists in the **WINNT\system32\drivers** folder.
   - Check whether a driver (such as "Cloud" or "Sniffer") is installed on top of the network card driver. If so, remove it and run WebTrace again.
   - Verify that there are administrator permissions on the machine.
   - Using `ipconfig /all`, check that only one IP address is assigned to the network card. WebTrace does not know how to handle multiple IP addresses assigned to the same card (IP spoofing).
   - Check the number of network cards installed. Run `webtrace –devlist` to receive a list of the available network cards.
   - If there is more than one card on the list, run `webtrace -dev <dev_name> <destination>`, where `<dev_name>` is one of the network card names shown in the list. If you discover that WebTrace is binding to the wrong card, you can use `webtrace set_device <dev_name>` to set a registry key that instructs WebTrace to use a specified card instead of the default one.
   - Verify that the network card is of the Ethernet type.
   - Contact the Customer Support Web site with the output of `webtrace.exe –debug` (for example, `webtrace.exe –debug www.merc-int.com`) and `ipconfig /all` on the machine.

# Network Virtualization Monitoring

Network virtualization starts and stops automatically as you start and stop a scenario. Network virtualization metrics are automatically collected during the scenario run, and displayed in the Network Virtualization monitor.

The most typical network effects which you can configure with the network virtualization software are:

- **Latency.** The **Latency** value you define represents the time in milliseconds that it takes an IP packet to cross the network. This is usually affected by geographical distance, the available bandwidth, the network load on the route between the two ends, and whether this is a terrestrial link or not.
- **Packet Loss.** The **Packet Loss** value you define represents the chance of losing IP packets while data travels through a network. Packets can get lost due to link faults or due to extreme network load.
- **Bandwidth.** The **Bandwidth** value you define represents the capacity of your network to transfer data.

For information on setting these parameters, see the Network Virtualization User Guide, available from the Start menu on the machine with the NV installation. For icon-based interfaces such as Windows 8.1, search for **NV User Guide**.

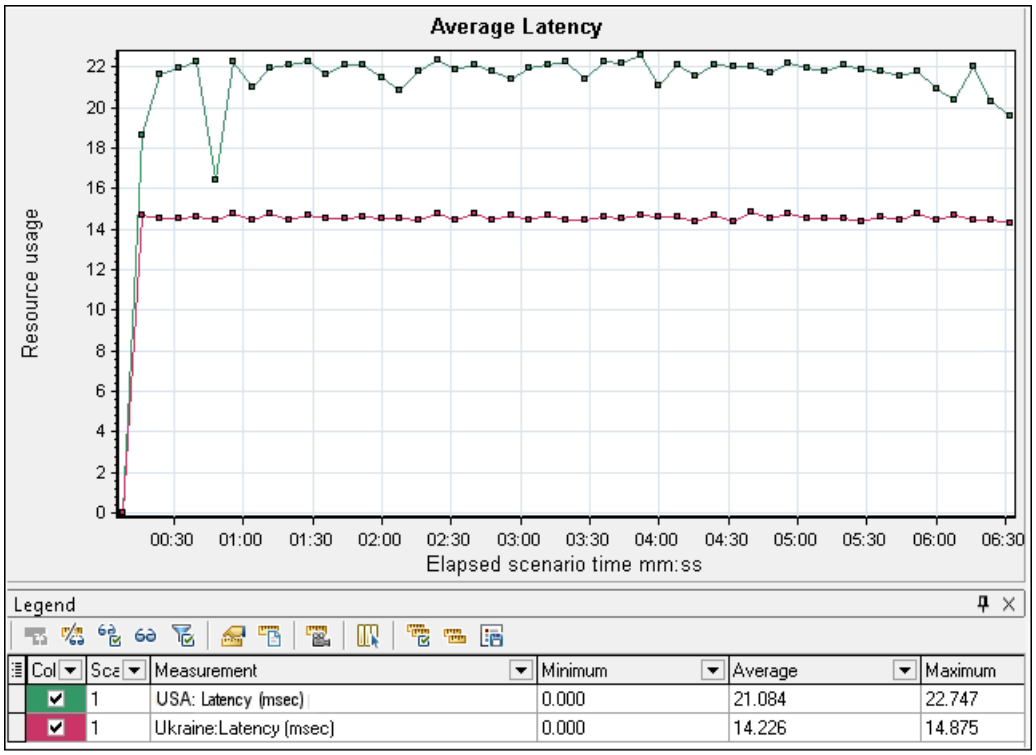
## Average Latency Monitor Graph

This graph shows the average recorded time required for a packet of data to travel from the indicated source point to the required destination, measured in milliseconds in the last 60 seconds.

| | |
|---|---|
| **Purpose** | Helps you evaluate the time required for a packet of data to travel over the network. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The average latency—the time in milliseconds required for a packet of data to reach its destination, per 60 second intervals. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **See also** | "Network Virtualization Monitoring" on the previous page |

**Example - Network Virtualization Per Group**

In the following example, you can see that the latency for the **USA** group reached its peak at nearly 4 minutes into the scenario run, while the **Ukraine** group remained fairly constant at approximately 14 msec.




If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Monitor Graph" below.

## Packet Loss Monitor Graph

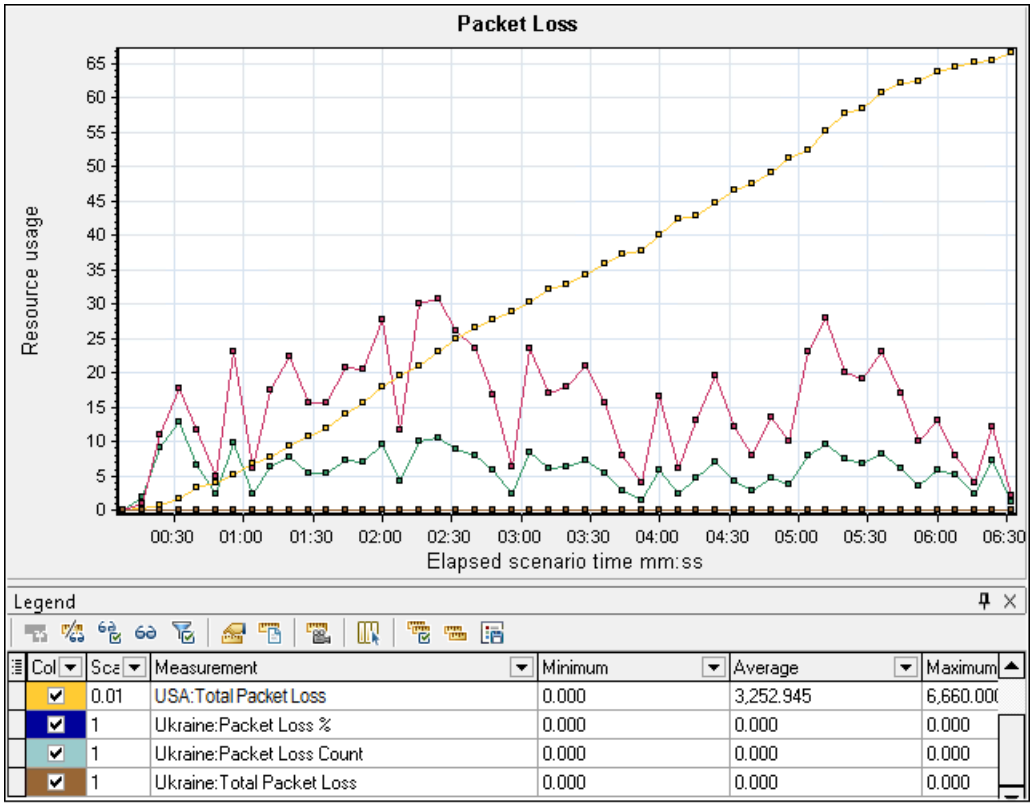
This graph shows packets lost during the last second of the scenario run. Packet loss occurs when data packets fail to reach their destination. It can result from gateway overload, signal degradation, channel

congestion, or faulty hardware.

| | |
|---|---|
| **Purpose** | Helps you understand how many data packets were lost over a specific time interval. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The following measurements:<br>• The percentage of lost packets from all packets that were sent.<br>• The number of data packets that were lost over 60 seconds.<br>• The total number of packets that were lost. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **Tip** | For LoadRunner Analysis (not applicable to monitoring graphs):<br>To view information for a specific location:<br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Network Virtualization Monitoring" on page 1209 |

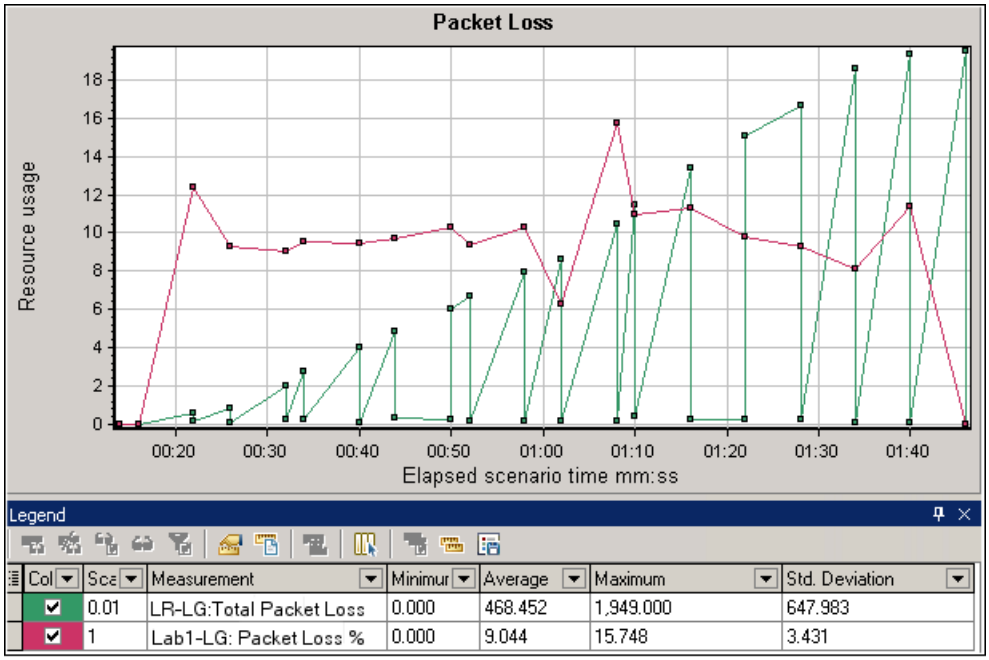
### Example - Network Virtualization Per Group

The following example shows how the total of packet loss for the **USA** group increased as the scenario progressed.



**Example - Network Virtualization Per Load Generator**

In the following example, you can see that the packet loss is grouped by load generator. This was the mode selected when you enabled Network Virtualization for the scenario.




## Average Throughput Monitor Graph

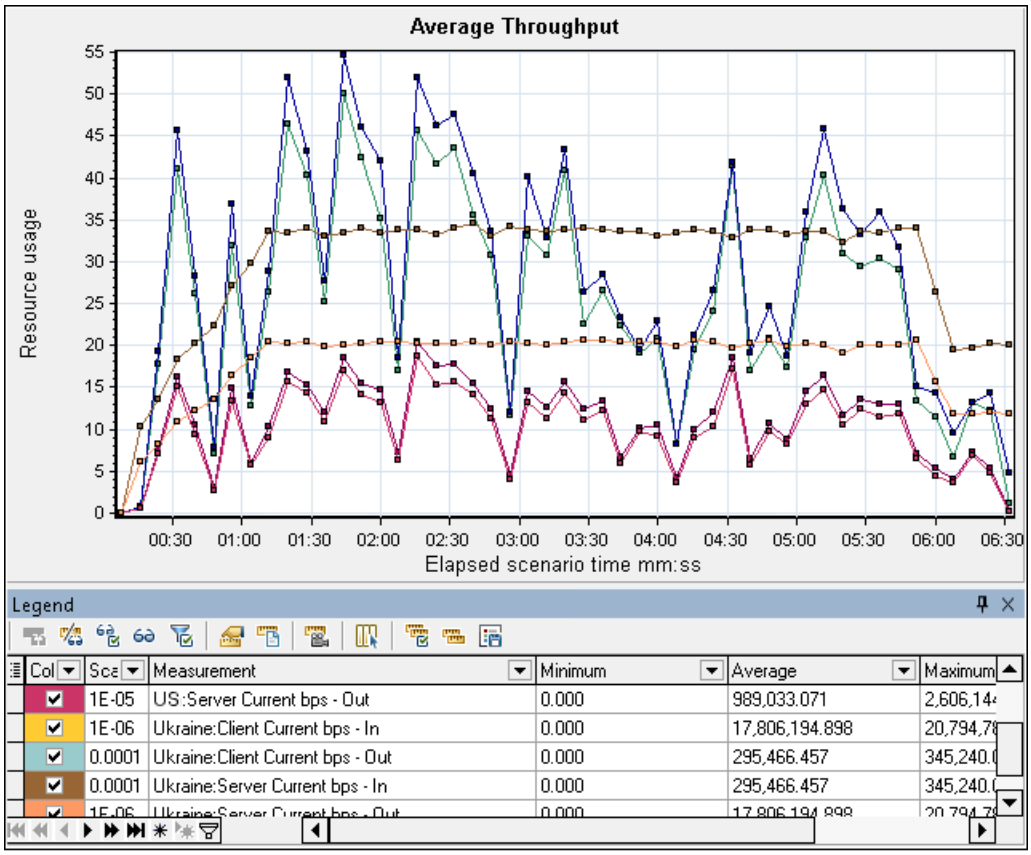
This graph shows the average data traffic passing to or from the virtualized location, measured in kilobytes per second (kBps).

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of load Vusers generate, in terms of the number of server and client throughput. The graph shows metrics for input and output traffic for both the server and client machines. Use the legend below the graph to determine the line color for each metric. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The rate of data passing to and from the virtual location, in kBps for the following metrics per group or load generator:<br>• Input to the client machine<br>• Output from the client machine<br>• Input to the server machine<br>• Output from the server machine |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |

| | |
|---|---|
| **Tips** | For LoadRunner Analysis (not applicable to monitoring graphs):<br>To view information for a specific location:<br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Total Throughput Monitor Graph" on page 1216 |

**Example**

In the following example, the average server input throughput was the lowest for the **Ukraine** group.




If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Monitor Graph" on page 1210.

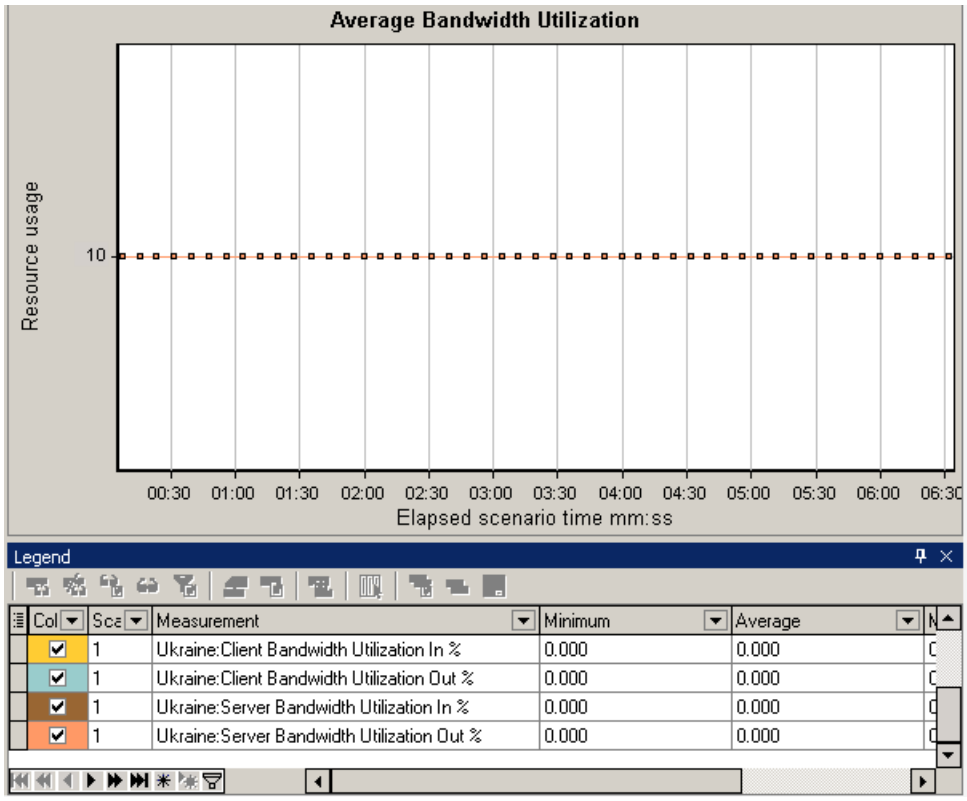
## Average Bandwidth Utilization Monitor Graph

This graph shows the average bandwidth utilized by a virtual user or a virtualized location from the maximal available bandwidth allocated for it during the last second, measured in percentages.

| | |
|---|---|
| **Purpose** | Helps you evaluate the bandwidth used over your network. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The percentage of bandwidth utilization. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **Tips** | For LoadRunner Analysis (not applicable to monitoring graphs):<br>To view information for a specific location:<br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Network Virtualization Monitoring" on page 1209 |

**Example**

In the following example, you can see that the bandwidth utilization for all locations and measurements, was constant at 10%.



If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Monitor Graph" on page 1210.

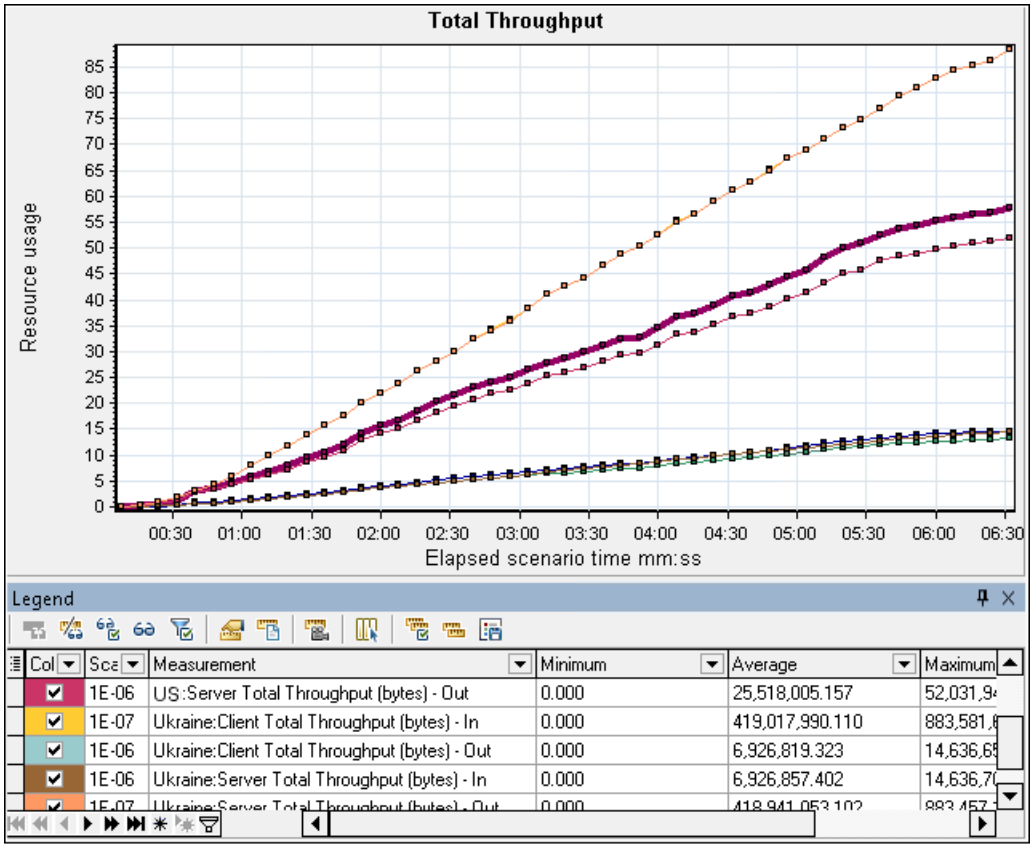
## Total Throughput Monitor Graph

Displays the total data traffic passing to or from the virtualized location, measured in kilobytes.

| | |
|---|---|
| **Purpose** | Helps you evaluate the total amount of load that Vusers generate while running a scenario with network virtualization.<br><br>The graph shows metrics for input and output traffic for both the server and client machines. The legend below the graph indicates the line color for each of these metrics. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Throughput of the server, in kilobytes per second (kBps). |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the **General** tab of the Options dialog box. |
| **Tips** | For LoadRunner Analysis (not applicable to monitoring graphs):<br><br>To view information for a specific location:<br><br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Average Throughput Monitor Graph" on page 1213 |

**Example**

In the following example, the highest throughput level was for the input data to the client, for the **Ukraine** group.




If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Monitor Graph" on page 1210.

# Runtime and Transaction Monitoring

## Runtime Graphs Overview

The **Runtime** monitor provides information about the status of the Vusers participating in the scenario, and the number and types of errors that the Vusers generate. In addition, the Runtime monitor provides the User-Defined Data Points graph, which displays the real time values for user-defined points in a Vuser script.

The Runtime monitor is enabled by default—it automatically begins monitoring Vusers at the start of a scenario.

You can view the following Runtime monitor graphs during a scenario run.

## Running Vusers Graph

The monitor's **Running Vusers** graph provides information about the status of the Vusers (or active JMeter threads) running in the current scenario on all load generator machines. The graph shows the number of running Vusers, while the information in the legend indicates the number of Vusers in each state.

| Color | Scale | Status | Max | Min | Avg | Std | Last |
|---|---|---|---|---|---|---|---|
| | 1 | Running | 14 | 2 | 7.632653... | 3.783389... | 14 |
| | 1 | Error | 0 | 0 | 0 | 0 | 0 |
| | 1 | Finished | 0 | 0 | 0 | 0 | 0 |

The Status field of each Vuser displays the current status of the Vuser. The following table describes each Vuser status.

| Status | Description |
|---|---|
| **Running** | The total number of Vusers currently running on all load generators. |
| **Ready** | The number of Vusers that completed the initialization section of the script and are ready to run. |
| **Finished** | The number of Vusers that have finished running. This includes both Vusers that passed and failed. |
| **Error** | The number of Vusers whose execution generated an error. Check the Status field in the Vuser view or the Output window for a complete explanation of the error. |

## User-Defined Data Points Graph

The **User-Defined Data Points** graph displays the real-time values of user-defined data points. You define a data point in your Vuser script by inserting an **lr_user_data_point** function at the appropriate place (**user_data_point** for GUI Vusers and **lr.user_data_point** for Java Vusers).

```
Action1()
{
    lr_think_time(1);
    lr_user_data_point ("data_point_1",1);
    lr_user_data_point ("data_point_2",2);
    return 0;
}
```

For Vuser protocols that support the graphical script representations such as Web, you insert a data point as a user-defined step. Data point information is gathered each time the script executes the function or step. For more information about data points, see the Function Reference.

By default, LoadRunner displays all of the data points in a single graph. The legend provides information about each data point. If desired, you can hide specific data points using the legend below the graphs.

You can also view data points offline, after the completion of the scenario. For more information, see the *LoadRunner Analysis User Guide*.

### Error Statistics Graph

The monitor's **Error Statistics** graph provides details about the number of errors that accrue during each second of the scenario run. The errors are grouped by error source—for example, the location in the script or the load generator name.

### Vusers with Errors Graph

The **Vusers with Errors** graph provides details about the number of Vusers that generate errors during scenario execution. The errors are grouped by error source.

## Transaction Monitor Graphs Overview

The Transaction monitor displays the transaction rate and response time during a scenario run. The Transaction monitor is enabled by default—it automatically begins monitoring Vuser transactions at the start of a scenario run. To conserve resources, you can disable the Transaction monitor from Controller.

You can view the following Transaction monitor graphs during a scenario run:

- The **Transaction Response Time** graph shows the average response time of transactions (y-axis) as a function of the elapsed time in the scenario (x-axis).
- The **Transaction Response Time by Location** graph shows the average response time for each of the virtual locations with which the transactions are performed (y-axis) as a function of the elapsed time in the scenario (x-axis).
- The **Transactions per Second (Passed)** graph shows the number of successful transactions performed per second (y-axis) as a function of the elapsed time in the scenario (x-axis).
- The **Transactions per Second (Passed) by Location** graph shows the number of successful transactions performed per second per virtual location (y-axis) as a function of the elapsed time in the scenario (x-axis).
- The **Transactions per Second (Failed, Stopped)** graph shows the number of failed and stopped transactions per second (y-axis) as a function of the elapsed time in the scenario (x-axis).
- The **Transactions per Second (Failed, Stopped) by location** graph shows the number of failed and stopped transactions per second per virtual location (y-axis) as a function of the elapsed time in the scenario (x-axis).
- The **Total Transactions per Second (Passed)** graph shows the total number of completed, successful transactions per second (y-axis) as a function of the elapsed time in the scenario (x-axis).

> **Note:**

- If there are no transactions defined in your Vuser script or if no transactions are being executed, no data will be displayed in the online monitor graphs.
- To generate Web Page diagnostics for each transaction, configure the Diagnostics options from Controller.

# SiteScope Server Monitoring

## SiteScope Resource Monitoring

The SiteScope Resources monitor graph shows the SiteScope resources measured during the scenario run. The SiteScope monitor can measure server, network, and processor performance counters. For detailed information on the performance counters that SiteScope can monitor, refer to the relevant SiteScope documentation.

Before setting up the SiteScope monitor, ensure that SiteScope has been installed on a server. You can install SiteScope on the same machine as Controller, or on a dedicated server. If SiteScope is installed on a machine other than Controller, verify that the SiteScope machine is accessible from the Controller machine.

The information you need to provide for monitoring SiteScope machines differs slightly from other monitors. The enhanced integration lets you set up SiteScope monitoring on machines that require user authentication and Secure HTTP connections. For details, see the "Add Machine Dialog Box" on page 1175.

For details on setting up the SiteScope monitor, see "Set up the SiteScope Integration" below.

## Set up the SiteScope Integration

This task describes how to set up your LoadRunner machine to integrate with the SiteScope monitor. For an overview of the SiteScope monitor, see "SiteScope Resource Monitoring" above.

You can configure SiteScope to connect anonymously without authentication or with authentication.

### Set up SiteScope without authentication

1. Open SiteScope and configure its monitors.
2. Open the URL: **http://localhost:8888/SiteScope/cgi/go.exe/SiteScope?page=topazVerify** and confirm that you can see Topaz XML data.
3. Open the **xmlmonitorshared.ini** file, in the <LoadRunner root>\dat\monitors folder and edit the **SiteScope** section as follows:

**Example:** [SiteScope]
;ExtensionDll=SiteScopeMonExt.dll
MetricDataURL=SiteScope/cgi/go.exe/SiteScope?page=topaz
MetricListURL=SiteScope/cgi/go.exe/SiteScope?page=topaz&operation=config
DefaultPort=8888
DlgTitle=SiteScope Monitor
RefreshMetricList=1
EnableAccount=1

### Set up SiteScope with authentication as a non-anonymous user

1. Open SiteScope and configure its monitors.
2. Modify the SiteScope administrator to have a user ID and password.
3. Add a new SiteScope user with a user ID and password.
4. Verify that you can see Topaz XML data at the URL http://localhost:8888/SiteScope/cgi/go.exe/SiteScope?page=topaz&account=login1 (note that Topaz requires a user index rather than a user name to view XML data)
5. Open C:\Program Files (x86)\Micro Focus\LoadRunner\dat\monitors\xmlmonitorshared.ini and edit the **SiteScope** section as follows (changes noted in bold type):

**Example:** [SiteScope]
;ExtensionDll=SiteScopeMonExt.dll
MetricDataURL=SiteScope/cgi/go.exe/SiteScope?page=topaz&account=**login1**
MetricListURL=
SiteScope/cgi/go.exe/SiteScope?page=topaz&operation=config&account=login1
DefaultPort=8888
DlgTitle=SiteScope Monitor
RefreshMetricList=1
EnableAccount=1
QueryLoginInfo=1

For more tips and guidelines, refer to the LoadRunner forums.

## Streaming State Statistics Monitoring

The Streaming State Statistics monitor provides statistics about the HTTP streaming videos in your scenario run.

## Streaming State Statistics Monitors

| Monitor | Description |
|---|---|
| **Streaming Time** | Monitors the start and buffering times of videos at any given moment in the scenario run. |
| **Streaming Network** | Monitors the download speed and video quality of the videos at any given moment in the scenario run. |
| **Streaming Events** | Monitors the play, pause, and seek events at any given moment in the scenario run. |
| **Streaming Video** | Monitors the number of videos playing or buffering at any given moment in the scenario run. |

## Streaming State Statistics Measurements

| Measurement | Description |
|---|---|
| Time to start (Can play) | The amount of time (in seconds) waited before the video started playing.<br>When multiple videos are running simultaneously, this value is an average for all the videos. |
| Buffering pause time | The amount of time (in seconds) waited due to video buffering.<br>When multiple videos are running simultaneously, this value is an average for all the videos. |
| Buffering seek time | The amount of time (in seconds) waited due to seeking a part of the video that has not yet been downloaded<br>When multiple videos are running simultaneously, this value is an average for all the videos. |
| Buffering rate | Percentage of wait/total time. This is calculated according to:<br>1 - (playing_time)/(playing_time+open_time+lag_time+seek_time)<br>When multiple videos are running simultaneously, this value is an average for all the videos. |
| Video Quality (Bandwidth) | The user/auto selected video quality . This is the bandwidth for the selected quality. If the protocols do not support multi-quality, the quality is 0.<br>When multiple videos are running simultaneously, this value is an average for all the videos. |

| Measurement | Description |
|---|---|
| Bandwidth | This is the same as the Bandwidth defined in the **Speed Simulation** runtime settings. Maximum bandwidth is 0.<br><br>When multiple videos are running simultaneously, this value is an average for all the videos. |
| Download speed | Streaming download speed (in Kb/sec).<br><br>When multiple videos are running simultaneously, this value is an average for all the videos. |
| Number of play events | Total number of "play" events. This value increments by 1 for each "play" event. e.g. When a video plays, pauses, and then plays again, the value is 2. |
| Number of seek events | Total number of "seek" events. This value increments by 1 for each "seek" event. |
| Number of pause events | Total number of "pause" events. This value increments by 1 for each "pause" event. |
| Number of download retries | Total number of retry events. If downloading of streaming data failed, streaming is not aborted immediately. Streaming is retried according to the number of retries defined in the runtime settings. |
| Number of buffering video | Total number of buffering videos. Increments by 1 when a video is buffering, and decrements by one when the video stops buffering. |
| Number of lagging video | Total number of lagging videos. Increments by 1 when a video is lagging, and decrements by one when the video stops lagging. |
| Number of playing video | Total number of playing videos. |

### See also:

- "Streaming State Statistics Graphs" on page 1467

# System Resource Monitoring

## System Resource Monitors Overview

You use LoadRunner's System Resource monitors to monitor a machine's system resource usage during a scenario run and isolate server performance bottlenecks.

A primary factor in a transaction's response time is its system resource usage. Using the System Resource monitors, you can monitor the Windows, UNIX, and SNMP resources on a machine during a scenario run, and determine why a bottleneck occurred on a particular machine.

The resource monitors are automatically enabled when you execute a scenario. However, you must specify the machine you want to monitor and which resources to monitor for each machine. You can also add or remove machines and resources during the scenario run.

## Windows Resource Monitoring

The Windows Resources monitor shows the Windows resources measured during the scenario run. Windows measurements correspond to the built-in counters available from the Windows Performance Monitor.

By default, LoadRunner monitors Windows resources using the native LoadRunner monitor engine.

If you are using the SiteScope monitor engine, ensure that SiteScope has been installed on a server. You can install SiteScope on the same server as Controller, or on a dedicated server.

If you want to monitor a remote Windows server that does not use Windows domain security, you must authenticate Controller on the remote Windows server. To authenticate Controller, create an account, or change the password of the account used to log on to Controller so that it matches the password and user name used to log on to the remote monitored Windows machine. When the remote Windows machine requests another machine's resources, it sends the logged-in user name and password of the machine requesting the resources.

> **Note:** Limited (non-admin) users cannot activate the monitoring using Windows Resource counters from a remote machine. This is due to a Windows limitation described in the following document: http://support.microsoft.com/kb/300702. You must log in with administrator permissions and verify that the Remote Registry service is running on the remote machine.

## UNIX Resource Monitoring

The UNIX Resources monitor shows the UNIX resources measured during the scenario. This graph helps you determine the impact of Vuser load on the various system resources. For a description of the measurements, see "UNIX Resources Performance Counters" on page 1226.

> **Note:** Before monitoring UNIX resources, you must configure all UNIX machines being monitored. For information, see "How to Set up the UNIX Monitoring Environment" on the next page.

## SNMP Resource Monitoring

The SNMP Resource monitor shows statistics for a Windows or UNIX machine using the Simple Network Management Protocol (SNMP). The SNMP Resources monitor is available for monitoring any machine that runs an SNMP agent.

## How to Set up the UNIX Monitoring Environment

This task describes how to configure the UNIX environment before setting up the UNIX Resources monitor.

1. Make sure that authentication by password is enabled on your server:
   a. Open **/etc/ssh/sshd_config**.
   b. Find the line that defines the password authentication property, and enable it.
      For example: **PasswordAuthentication yes**
      If this property is not defined, add it.
   c. Save the file and restart the **sshd** service.
2. Verify that the following commands run on the UNIX machine without errors:

```
/bin/cat /proc/stat| /bin/grep cpu | /usr/bin/head -n 1
/bin/cat /proc/stat | /bin/grep btime
/bin/cat /proc/uptime
/bin/cat /proc/stat | /bin/grep ctxt
/bin/cat /proc/vmstat | /bin/grep pgpgin -A 3
/bin/cat /proc/loadavg
/bin/cat /proc/net/dev | /usr/bin/head -n3 | /usr/bin/tail -n1
/bin/cat /proc/stat | /bin/grep intr | /usr/bin/awk '{print $2}'
/bin/cat /proc/meminfo | /bin/grep MemTotal -A 2
```

3. Configure the monitor measurements in the Controller
   For task details, see "How to Set Up a Monitoring Environment" on page 1171.
   In the **Resource Measurements on <machine>** section of the UNIX Resources dialog box, click **Add** to open the UNIX Kernel Statistics dialog box, and then select the available measurements and server properties.
   For a description of the available UNIX monitor measurements, see "UNIX Resources Performance Counters" on the next page.

> **Note:** In the UNIX Kernel Statistics dialog box, selecting the **Recover Connection** check box

enables the Controller to try to recover a broken connection to a monitored UNIX server. If the **Recover Connection** check box is not selected, a broken connection will be maintained until the end of the Controller session, and as a result server details will not be received from the monitored server. By default, the **Recover Connection** check box is selected.

## UNIX Resources Performance Counters

The following default measurements are available for the UNIX machine:

| Measurement | Description |
|---|---|
| **Average load** | Average number of processes simultaneously in Ready state during the last minute |
| **Collision rate** | Collisions per second detected on the Ethernet |
| **Context switches rate** | Number of switches between processes or threads, per second |
| **CPU utilization** | Percent of time that the CPU is utilized |
| **Disk rate** | Rate of disk transfers |
| **Incoming packets error rate** | Errors per second while receiving Ethernet packets |
| **Incoming packets rate** | Incoming Ethernet packets per second |
| **Interrupt rate** | Number of device interrupts per second |
| **Outgoing packets errors rate** | Errors per second while sending Ethernet packets |
| **Outgoing packets rate** | Outgoing Ethernet packets per second |
| **Page-in rate** | Number of pages read to physical memory, per second |
| **Page-out rate** | Number of pages written to pagefile(s) and removed from physical memory, per second |
| **Paging rate** | Number of pages read to physical memory or written to pagefile(s), per second |
| **Swap-in rate** | Number of processes being swapped |
| **Swap-out rate** | Number of processes being swapped |

| | |
|---|---|
| **System mode CPU utilization** | Percent of time that the CPU is utilized in system mode |
| **User mode CPU utilization** | Percent of time CPU is utilized in user mode |

**Note:** In the UNIX Kernel Statistics dialog box, selecting the **Recover Connection** check box enables the Controller to try to recover a broken connection to a monitored UNIX server. If the **Recover Connection** check box is not selected, a broken connection will be maintained until the end of the Controller session, and as a result server details will not be received from the monitored server. By default, the **Recover Connection** check box is selected.

## Add Windows Resources Measurements Dialog Box

This dialog box enables you to select the Windows resources to monitor. The Windows resources correspond to the built-in counters available from the Windows Performance Monitor.

| | |
|---|---|
| **To access** | Right-click a graph **>** **Add Measurements**<br>In the **Resource Measurements** section of the **Windows Resources** dialog, click **Add**. |
| **Relevant tasks** | "How to Set Up a Monitoring Environment" on page 1171 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Explain>> | Displays a description of the selected counter. |
| **Counters/Measurements** | The resource counter/measurement to monitor. Select multiple counters using the CTRL key. For a description of the measurements, see the Description box in the lower section of the dialog box. |
| **Instances** | If multiple instances of the selected counter are running, select one or more instances to monitor for the selected counter. |
| **Object** | The object to monitor on the specified Windows machine. |

# TruClient - Native Mobile Monitors

## CPU Utilization Percentage Graph

This monitor graph displays the percentage of the CPU being utilized during the test run for TruClient Native Mobile Vuser scripts.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of CPU utilized by an application. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | The percentage of the CPU utilized during the test run. |

## Total Free Memory In Device Monitor

This monitor displays the free memory on a mobile device as a function of time, for TruClient Native Mobile scripts.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of memory available on the device during the test run. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | The amount of free memory in KBs. |

## Total Memory Consumed by Application Monitor

This graph displays the memory consumed by the application, as a function of time for TruClient Native Mobile scripts.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of memory used by the application. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | The memory consumed by the application in KBs. |

# Web Application Server Monitoring

You use LoadRunner's Web Application Server Resource monitors to monitor Microsoft Active Server Pages during a scenario run and isolate application server performance bottlenecks.

The Microsoft Active Server Pages (ASP) monitor displays statistics about the resource usage on the ASP server during the scenario run.

## MS Active Server Pages Performance Counters

The following table describes the default counters that can be monitored:

| Measurement | Description |
|---|---|
| **Errors per Second** | The number of errors per second. |

| | |
|---|---|
| **Requests Wait Time** | The number of milliseconds the most recent request was waiting in the queue. |
| **Requests Executing** | The number of requests currently executing. |
| **Requests Queued** | The number of requests waiting in the queue for service. |
| **Requests Rejected** | The total number of requests not executed because there were insufficient resources to process them. |
| **Requests Not Found** | The number of requests for files that were not found. |
| **Requests/sec** | The number of requests executed per second. |
| **Memory Allocated** | The total amount of memory, in bytes, currently allocated by Active Server Pages. |
| **Errors During Script Runtime** | The number of failed requests due to runtime errors. |
| **Sessions Current** | The current number of sessions being serviced. |
| **Transactions/sec** | The number of transactions started per second. |

## Microsoft Active Server Pages Dialog Box

This dialog box enables you to select the items to monitor on the MS Active Server Pages application server.

| | |
|---|---|
| **To access** | Right-click a graph **>** **Add Measurements**<br>Click **Add** in the **Resource Measurements** section of **Microsoft Active Server Pages** dialog. |
| **Relevant tasks** | "How to Set Up a Monitoring Environment" on page 1171 |
| **See also** | "MS Active Server Pages Performance Counters" on the previous page |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Add ... | Adds the selected measurement to the list of measurements in the **Measurements on <machine>** section of the Microsoft Active Server Pages dialog box. |

| | |
|---|---|
| **Counters** | Select a resource counter to monitor. Select multiple counters using the **CTRL** key. For a definition of each counter, click **Explain**. |
| **Instances** | If multiple instances of the selected counter are running, select one or more instances to monitor for the selected counter. |
| **Object** | Select the object being monitored on the specified machine. |

# Web Resource Monitors

## Web Resource Monitoring Overview

The Web Resource monitor enables you to analyze the following resources on the Web server during a scenario run: throughput, HTTP requests, downloaded pages, server retries, TCP/IP connections, and SSL Connections.

You can view the following resource monitor graphs during a scenario run.

### Hits per Second Graph

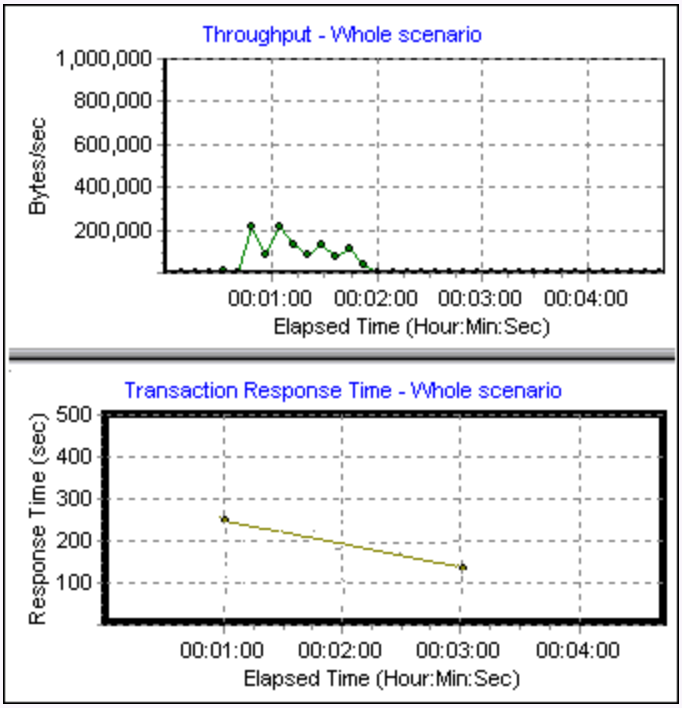
The **Hits Per Second** graph shows the number of hits (HTTP requests) to the Web server (y-axis) as a function of the elapsed time in the scenario (x-axis). This graph can display the whole step, or the last 60, 180, 600, or 3600 seconds. You can compare this graph to the Transaction Response Time graph to see how the number of hits affects transaction performance.

### Throughput Graph

The **Throughput** graph shows the amount of throughput (y-axis) on the Web server during each second of the scenario run (x-axis). Throughput is measured in bytes and represents the amount of data that the Vusers received from the server at any given second. You can compare this graph to the Transaction Response Time graph to see how the throughput affects transaction performance.

In the following example, the Transaction Response time graph is compared with the Throughput graph. It is apparent from the graph that as the throughput decreases, the transaction response time also decreases. The peak throughput occurred at approximately 1 minute into the step. The highest response time also occurred at this time.

**Example:**




## HTTP Responses per Second Graph

The **HTTP Responses per Second** graph shows the number of HTTP status codes (y-axis)—which indicate the status of HTTP requests, for example, "the request was successful" or "the page was not found"—returned from the Web server during each second of the scenario run (x-axis).

The HTTP responses are grouped by status code. You can also group the results shown in this graph by script (using the "Group By" function) to locate scripts which generated error codes.

For a list of status codes and their explanations, see "HTTP Status Codes" on page 1234.

## WebSocket Statistics Graph

The **WebSocket Statistics** graph provides you with WebSocket statistics for your test run: the number of new connections, the number of bytes sent and received, and the number of failed connections.

For details, see "WebSocket Statistics Monitor" on page 1233.

## Pages Downloaded per Second Graph

The **Pages Downloaded per Second** graph shows the number of Web pages (y-axis) downloaded from the server during each second of the scenario run (x-axis). This graph helps you evaluate the amount of load Vusers generate, in terms of the number of pages downloaded.

**Note:** To view the Pages Downloaded per Second graph, you must select **Pages per second**

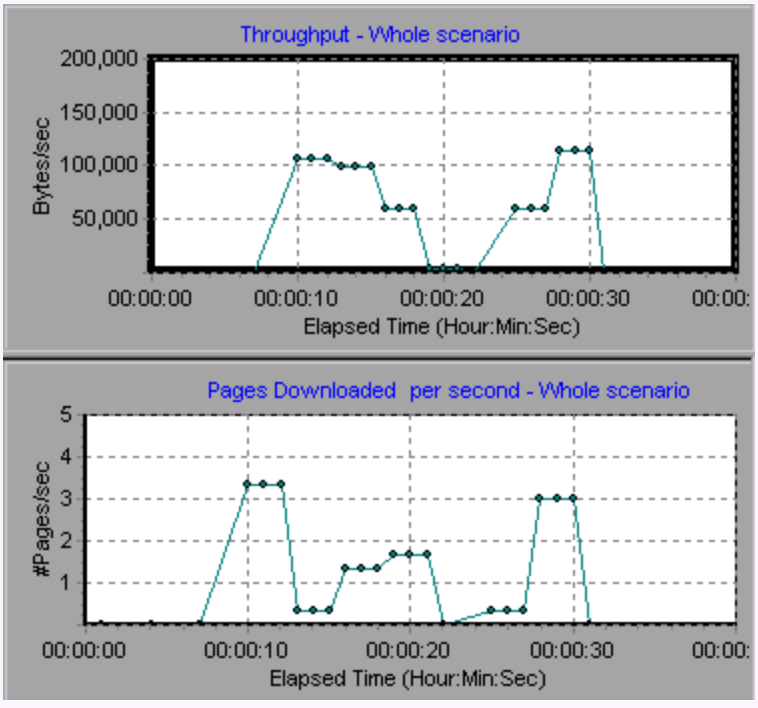
**(HTML Mode only)** from the script's runtime settings Preferences tab before running your scenario.

Like throughput, downloaded pages per second is a representation of the amount of data that the Vusers received from the server at any given second.

- The Throughput graph takes into account each resource and its size (for example, the size of each **.gif** file, the size of each Web page).
- The Pages Downloaded per Second graph takes into account simply the number of pages.

In the following example, the Throughput graph is compared with the Pages Downloaded per Second graph. It is apparent from the graph that throughput is not proportional to the number of pages downloaded per second. For example, between 15 and 16 seconds into the scenario, the throughput decreased while the number of pages downloaded per second increased.

**Example:**




## Retries per Second Graph

The **Retries Per Second** graph shows the number of attempted Web server connections (y-axis) as a function of the elapsed time in the scenario (x-axis).

A server connection is retried when:

- The initial connection was unauthorized

- Proxy authentication is required
- The initial connection was closed by the server
- The initial connection to the server could not be made
- The server was initially unable to resolve the load generator's IP address

### Connections Graph

The **Connections** graph shows the number of open TCP/IP connections (y-axis) at each point in time of the scenario (x-axis). One HTML page may cause the browser to open several connections, when links on the page go to different Web addresses. Two connections are opened for each Web server.

This graph is useful in indicating when additional connections are needed. For example, if the number of connections reaches a plateau, and the transaction response time increases sharply, adding connections would probably cause a dramatic improvement in performance (reduction in the transaction response time).

### Connections per Second Graph

The **Connections Per Second** graph shows the number of new TCP/IP connections (y-axis) opened and the number of connections that are shut down each second of the scenario (x-axis).

This number should be a small fraction of the number of hits per second, because new TCP/IP connections are very expensive in terms of server, router and network resource consumption. Ideally, many HTTP requests should use the same connection, instead of opening a new connection for each request.

### SSLs per Second Graph

The **SSLs per Second** graph shows the number of new and reused SSL Connections (y-axis) opened in each second of the scenario (x-axis). An SSL connection is opened by the browser after a TCP/IP connection has been opened to a secure server.

Because creating a new SSL connection entails heavy resource consumption, you should try to open as few new SSL connections as possible; once you have established an SSL connection, you should reuse it. There should be no more than one new SSL connection per Vuser.

If you set your runtime settings to simulate a new Vuser at each iteration (using the runtime settings **Browser Emulation** node), you should have no more than one new SSL connection per Vuser per iteration. Ideally, you should have very few new TCP/IP and SSL connections each second.

## WebSocket Statistics Monitor

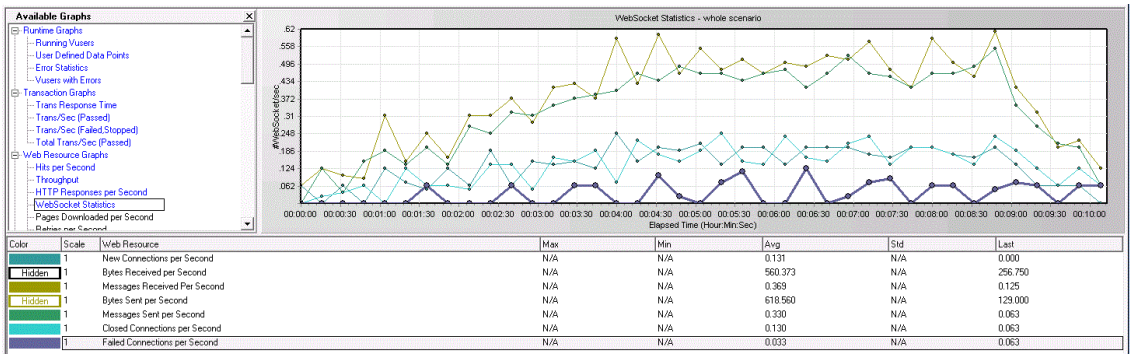
The **WebSocket Statistics** monitor provides you with statistics for the WebSocket data during the scenario run, such as byte rate, connection status, and the number of messages.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |

| | |
|---|---|
| **Y-axis** | WebSockets per second throughout the whole scenario. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the **General** tab of the Options dialog box. |

**Example - WebSocket Statistics**

In the following example, you can see that over 3000 bytes were sent per second.




**Example - WebSocket Statistics**

The measurements for this monitor are:

- New Connections per second
- Bytes Received per second
- Messages Received per second
- Bytes Sent per second
- Messages Sent per second
- Closed Connections per second
- Failed Connections per second

## HTTP Status Codes

The following table displays a list of HTTP status codes.

| Code | Description | Code | Description |
|---|---|---|---|
| 200 | OK | 406 | Not Acceptable |
| 201 | Created | 407 | Proxy Authentication Required |
| 202 | Accepted | 408 | Request Timeout |
| 203 | Non-Authoritative Information | 409 | Conflict |
| 204 | No Content | 410 | Gone |

| | | | |
|---|---|---|---|
| 205 | Reset Content | 411 | Length Required |
| 206 | Partial Content | 412 | Precondition Failed |
| 300 | Multiple Choices | 413 | Request Entity Too Large |
| 301 | Moved Permanently | 414 | Request - URI Too Large |
| 302 | Found | 415 | Unsupported Media Type |
| 303 | See Other | 416 | Requested range not satisfiable |
| 304 | Not Modified | 417 | Expectation Failed |
| 305 | Use Proxy | 500 | Internal Server Error |
| 307 | Temporary Redirect | 501 | Not Implemented |
| 400 | Bad Request | 502 | Bad Gateway |
| 401 | Unauthorized | 406 | Not Acceptable |
| 402 | Payment Required | 407 | Proxy Authentication Required |
| 403 | Forbidden | 503 | Service Unavailable |
| 404 | Not Found | 504 | Gateway Timeout |
| 405 | Method Not Allowed | 505 | HTTP Version not supported |

For more information on the status codes above, see http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html#sec10

# Web Server Resource Monitoring

## Web Server Resource Monitoring Overview

Web Server Resource monitors provide you with information about the resource usage of the Microsoft IIS and Apache Web servers during performance test execution. To obtain this data, you need to activate the online monitor for the server and specify which resources you want to measure before executing the test.

**Note:** Certain measurements or counters are especially useful for determining server performance and isolating the cause of a bottleneck during an initial stress test on a Web server.

## How to change the Apache default server properties

This task describes how to modify the Apache default server properties that are defined in the monitor configuration file.

1. Open the **apache.cfg** file in the **<performance center root folder>\dat\monitors** folder.
2. Edit the following parameters after the **Delimiter=:** statement:

   **InfoURL.** Server statistics information URL

   **ServerPort.** Server port number

   **SamplingRate.** Rate (milliseconds) at which the LoadRunner monitor will poll the server for the statistics information. If this value is greater than 1000, LoadRunner will use it as its sampling rate. Otherwise, it will use the sampling rate defined in the Monitors tab of the Options dialog box.
3. Save and close the file.

## Microsoft IIS Performance Counters

The following table describes the measurements and server properties that can be monitored on the Microsoft IIS Web server during the test run:

| Object | Measurement | Description |
|---|---|---|
| **Web Service** | **Bytes Sent/sec** | The rate at which the data bytes are sent by the Web service |
| **Web Service** | **Bytes Received/sec** | The rate at which the data bytes are received by the Web service |
| **Web Service** | **Get Requests/sec** | The rate at which HTTP requests using the GET method are made. Get requests are generally used for basic file retrievals or image maps, though they can be used with forms. |
| **Web Service** | **Post Requests/sec** | The rate at which HTTP requests using the POST method are made. Post requests are generally used for forms or gateway requests. |
| **Web Service** | **Maximum Connections** | The maximum number of simultaneous connections established with the Web service |
| **Web Service** | **Current Connections** | The current number of connections established with the Web service |
| **Web Service** | **Current NonAnonymous Users** | The number of users that currently have a non-anonymous connection using the Web service |

| | | |
|---|---|---|
| **Web Service** | **Not Found Errors/sec** | The rate of errors due to requests that could not be satisfied by the server because the requested document could not be found. These are generally reported to the client as an HTTP 404 error code. |
| **Process** | **Private Bytes** | The current number of bytes that the process has allocated that cannot be shared with other processes. |

### Apache Performance Counters

The following table describes the measurements and server properties that can be monitored on the Apache Web server during the test run:

| Measurement | Description |
|---|---|
| **# Busy Servers** | The number of servers in the Busy state |
| **# Idle Servers** | The number of servers in the Idle state |
| **Apache CPU Usage** | The percentage of time the CPU is utilized by the Apache server |
| **Hits/sec** | The HTTP request rate |
| **KBytes Sent/sec** | The rate at which data bytes are sent from the Web server |

# Network Virtualization Integration

LoadRunner integrates with Network Virtualization (NV) to help you test point-to-point performance of network-deployed products under real-world conditions.

By incorporating NV into your scenario, you can generate more realistic results by configuring Vuser groups with specific network effects, depending on the actual routes and characteristics of the networks. For example, you could define a location from New York to London and another one from Los Angeles to New York. As a result of this, your scenario performs the test in a more realistic environment that better represents the actual deployment of your application.

During the scenario run, you can run the NV monitors to verify that your scenario is performing network virtualization. For details, see "Network Virtualization Monitoring" on page 1209.

After the scenario run, you can view the results in Analysis or the NV Insights report. For details, see "Network Virtualization Graphs" on page 1456, "Transaction Response Time by Location Graph" on page 1485, or "NV Insights Report" on page 1239.

Network virtualization can be used in conjunction with IP spoofing.

For details about Network Virtualization, see the Network Virtualization for LoadRunner and Performance Center Help (select the relevant version).

# Installation

The LoadRunner setup wizard prompts you to install NV at the conclusion of the installation. If you did not install the NV components as part of the LoadRunner installation, you can run the NV installation manually at any time.

- The installation files for Windows are located in the **<LoadRunner installation package>Additional Components\Network Virtualization** folder.
- The installation files for load generators on Linux are located in the **<LoadRunner installation package>\Standalone Applications\linux_x64\NV4LG** folder.

Refer to the installation guide in the **Network Virtualization** folder to review the system requirements and to determine which NV components to install on your machine. You can also find this information in the installation section of the Network Virtualization for LoadRunner and Performance Center Help (select the relevant version).

**Note:**

- The latest LoadRunner version must be installed before you can install the NV components.
- If you have a Shunra or a Network Virtualization version prior to 12.50 installed on the machine, you must uninstall the earlier version and reboot the computer before installing NV. Otherwise, the NV installation will fail. (For Network Virtualization 12.50 and later, the installation process automatically uninstalls NV components.)

## See also:

- "How to Run a Scenario with Network Virtualization" on page 1240
- Network Virtualization product page

# Network Virtualization Locations

Network Virtualization comes with pre-defined virtual location settings for your network virtualization testing, which emulate conditions unique to specific geographic areas. In addition, you can define your own custom locations.

You view and add locations in the "Virtual Locations Settings Dialog Box" on page 1243. You can define global virtualization settings or configure the settings for each of the locations separately. For task details, see "How to Run a Scenario with Network Virtualization" on page 1240.

After the scenario run, using LoadRunner Analysis, you can group the scenario results by the virtual location name. For information on grouping metrics, see "Filtering and Sorting Graph Data" on page 1342 in the Analysis section. You can also use the NV Insights Report to analyze the results of a scenario run. For details, see "NV Insights Report" on the next page.

# NV Insights Report

The NV Insights report is a comprehensive network analysis report that provides information about how your application performs over various networks, during a scenario run.

> **Tip:** The NV Insights report replaces the NV Analytics report.

To generate the NV Insights report, you must configure Network Virtualization functionality for your scenario, so that Controller can emulate the conditions required for each of the networks. After you run a scenario, Controller analyzes and compares the behavior of the application over the various networks, and identifies problematic transactions in the scenario. The resulting NV Insights Report can help to pinpoint root causes for performance issues, and provide optimization recommendations to resolve the issues, thereby improving the performance of your application.

For information about how to incorporate NV Insights functionality into a scenario, see "How to Run a Scenario with Network Virtualization" on the next page.

For details on the NV Insights report, see the Network Virtualization for Performance Testing Help Center.

## Considerations for Working with NV Insights

- NV Insights functionality is available for the following protocols only: Flex, SAP Web, Siebel Web, TruClient-Web, and Web HTTP/HTML.
- When scheduling time slots for running performance tests, allow additional time (of approximately 15 minutes) for each NV Insights-enabled test.
- Your Controller and load generator machines may require additional disk space for processing the data for the NV Insights Report.
- When working with a limited user account on the Controller machine, ignore the error message displayed when creating an NV Insights Report. Launch the report directly from the scenario results folder.
- If your test includes several Vuser groups, you may need to increase the time needed to generate the NV Insights Report. Edit <LoadRunner root>\config\wlrun7.ini. Under the [General] section, increase the value for the NVAnalyticsTimeout parameter. By default, the value is set to 10 minutes (in seconds).
- When using NV Insights functionality, Controller creates "hidden" Vusers (2 hidden Vusers for each Vuser group) in addition to the Vusers that are defined in the scenario scheduler. If the scenario includes parameters with unique data, make sure that the list of parameter values includes extra values for the hidden Vusers.

> **Note:** The "hidden" Vusers appear at the bottom of the Vuser Group pane while a scenario is running.

- The NV Insights Report does not generate when running a TruClient script on a remote load generator, using Service mode.

  **Resolution:** Run the remote agent in Process mode instead. For details on how to restart the agent as a process, see "Run SAP GUI Scripts from the Controller" on page 686.

# Excluding Machines from Network Virtualization

In some situations, you may need to exclude certain machines that may affect the virtualization emulation, from the network virtualization . A typical example is a software update server.

To exclude a machine, you configure the IP Filter settings of your network virtualization software. When you exclude a machine, their network effects will not be included in the network virtualization results.

Which machines should you exclude? Any machine that if emulated, may affect the results of the actual scenario during its run, (for example, Controller) should be excluded. The following machines are excluded by default:

- MI Listener, and proxy server machines
- The Diagnostics Commander server
- A machine running SiteScope

The following are situations to consider excluding a machine from network virtualization:

- In a Multiprotocol scenario that includes a Web server and a database server; where information from the database server is not required as a part of the load test. In such a case, you would exclude the database server.
- Deployment and software upgrade servers.
- Servers that run and store scripts on a shared network drive.

# How to Run a Scenario with Network Virtualization

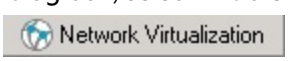
This task describes how to run a scenario using network virtualization and view the metrics in Analysis and the "NV Insights Report" on the previous page.

1. Prerequisites

   Make sure that you have the current version of Network Virtualization for LoadRunner installed on your Controller and load generator machines. For details, see the LoadRunner Installation Guide.

2. Open the Virtual Locations Settings dialog box.

   a. On the Controller toolbar, click the **Show Virtual Locations Settings** button to open the dialog box. For details, see "Virtual Locations Settings Dialog Box" on page 1243.

   b. In the Virtual Locations Settings dialog box, select **Enable Network Virtualization**. After you enable network virtualization, the 

icon appears in the Controller status bar.

3. Choose a virtualization mode.
   - Select the default **Per Group** mode to enable you to run multiple Vuser groups with different virtual locations on the same load generator.

     For NV insights, select the **Per Group** mode.
   - Select the **Per Load Generator** mode to enable you to run multiple Vuser groups on each load generator, where all the Vuser groups on any particular load generator use the same virtual location.

   For more information on the different modes, see "Per Group versus Per Load Generator" on page 1246.
4. Click **Common Settings** and, in the dialog box that opens, set the global network virtualization settings that apply to all virtual locations in the scenario.

   > **Note:** This step provides general information. For more details, see the Network Virtualization for LoadRunner and Performance Center Help (select the relevant version).

   a. Define the packet capture settings, as needed.

      For NV Insights, set **Enable Packet Capture** to **Yes**.

   b. In the **Exclude these IPs** area, specify any machines that you want to exclude from network virtualization for all locations. For details on which machines to exclude, see "Excluding Machines from Network Virtualization" on the previous page.

   c. Click **OK** to apply your changes to all virtual locations in the scenario and close the dialog box.
5. Add virtual locations. (Optional)

   If you do not want to use one of the built-in virtual locations, you can define your own custom locations. In the Virtual Locations Settings dialog box, click in the **Virtual Locations** list and add one or more location names. For location names, use the ANSI standard format. You cannot use the following characters:
   \ / : " ? ' < > | * % ^ , ! { } ( ) ; = #

   To add a new location, enter the name of the location in the next available line.

   For more details, see " Network Virtualization Locations" on page 1238.
6. Configure the network virtualization settings per location.
   a. In the Virtual Locations Settings dialog box, select a location in the **Virtual Locations** list.

   b. Click **Configure** to open the Virtual Location Editor. For details on how to configure your virtual location, see the Network Virtualization for LoadRunner and Performance Center Help (select the relevant version).

   > **Note:** When defining bandwidth settings for your scenario, bandwidth can either be shared between all Vusers or set per Vuser. The following protocols support shared bandwidth only:

- Citrix ICA
- COM/DCOM
- Java Record Replay
- Java Vuser
- MAPI

If you define a scenario with any of these protocols with individual bandwidth settings, a warning is displayed when you run the scenario.

7. Close the Virtual Locations Settings dialog box.
8. Do one of the following:
   - For **Per Group** mode, configure each Vuser group.
     i. In the Scenario Groups pane, click the **Virtual Location** box for the Vuser group and select a location from the dropdown list. If you do not want to use network virtualization for a specific group, select **None**.
     ii. Select the **NV Insights** check box if you want to include the Vuser group in the NV Insights Report that is produced after the scenario run.
     iii. Repeat the above steps for each Vuser group.

     **Note:** If the script runs on a protocol that does not support the Per Group mode, the icon appears next to the virtual location showing that the script will run according to the Load Generator settings.

   - For **Per Load Generator** mode, set a location for each load generator.
     i. On the main Controller toolbar, click the **Load Generator** button to open the Load Generators dialog box.
     ii. Select the load generator and click **Details**.
     iii. Click the **Network Virtualization** tab, click the **Default Virtual Location** drop-down and select a virtual location for the Load Generator. If you do not want to use network virtualization for a specific load generator, select **None**.
     iv. Repeat the above steps for each load generator.
9. Initiate the network virtualization monitors.
   a. In the bottom section of the Controller window, select the **Run** tab.
   b. In the **Available Graphs** pane, locate the **Network Virtualization** node. Double-click the metrics to monitor them. For details, see "Network Virtualization Monitoring" on page 1209.

**Note:** If a load generator is connected over a firewall, add monitors manually using the

> Monitor Over Firewall component. For details, see "Set Up Your LoadRunner System Over Firewalls" on page 1136.

10. Filter the measurements by location. (Optional)

    To view graphs for a specific location:

    a. Click within a graph.

    b. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.

    c. In the **Filter condition** section, select the **Location** row, and specify the desired location.

    To group the measurements by location:

    a. Click within a graph and select **Set Filter/ Sort By** from the right-click menu.

    b. In the **Group by** section, select **Location** in the **Available groups** pane.

    c. Click the right arrow to move it into the **Selected groups** pane.

11. Export the virtual locations settings. (Optional)

    In the Virtual Locations Settings dialog box, click **Export** to save the settings to a file for future use.

12. Save and run the scenario.

    Complete all other steps required to set up your scenario and save it. The network virtualization settings are saved together with the scenario. Run the scenario in the normal way. Network virtualization starts and stops automatically with the scenario. View the metrics in the monitors that you added earlier.

> **Note:** If the scenario includes NV Insights, the system-generated "hidden" Vusers appear at the bottom of the Vuser Group pane while the scenario is running.

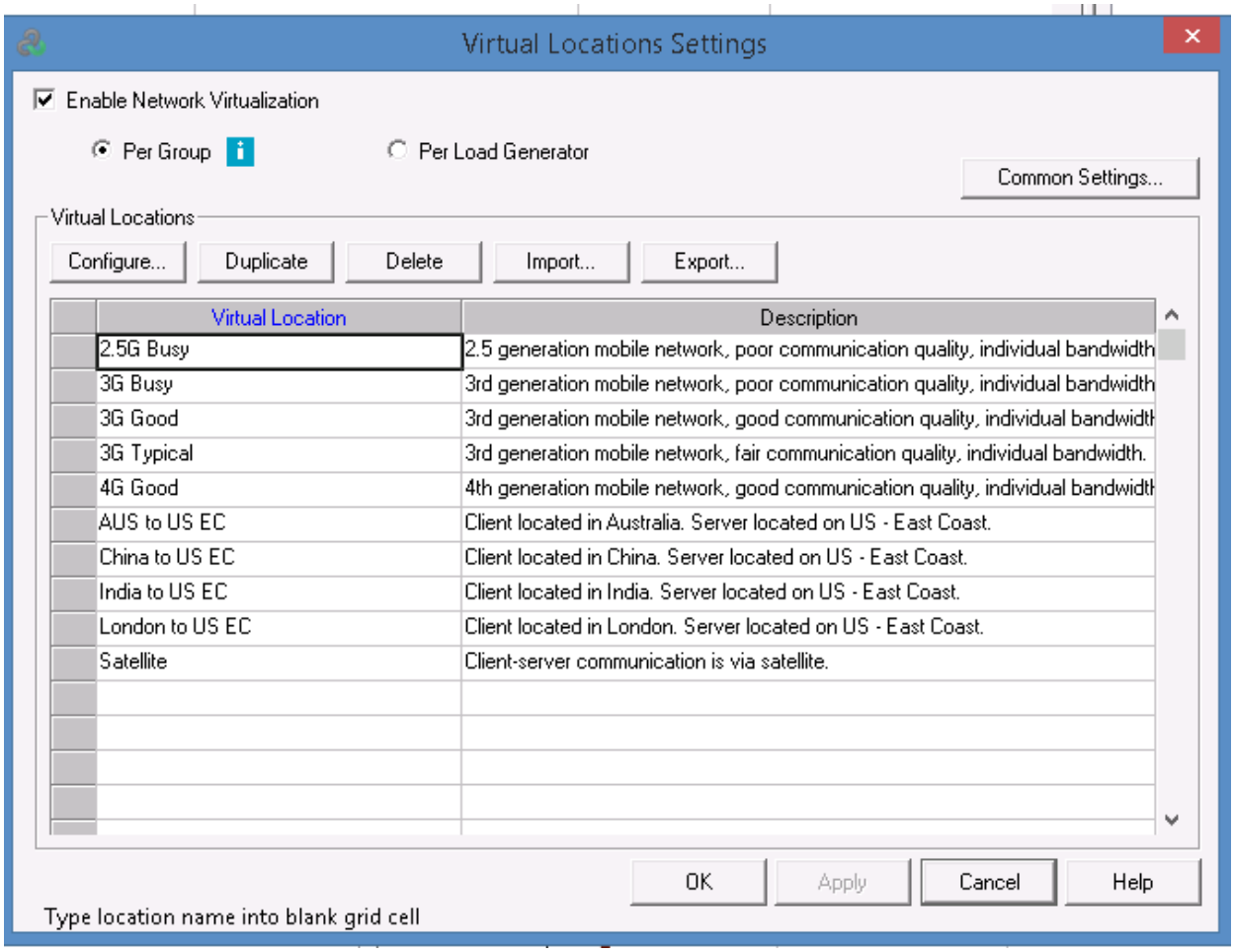
13. View the metrics in LoadRunner Analysis and the NV Insights Report.

    Network virtualization metrics are automatically collected during the scenario run. Open Analysis and view the network virtualization metrics. You can group them by location, and correlate the metrics with other data, such as response time.

    If the scenario includes NV Insights, click the **Display NV Insights Report** button on the Controller toolbar to generate and display the NV Insights Report. For details, see "NV Insights Report" on page 1239.

# Virtual Locations Settings Dialog Box

This dialog box allows you to configure your virtual locations.




| | |
|---|---|
| **To access** | **Controller toolbar >** **Show Virtual Locations Settings** button |
| **Important information** | To access this dialog box, you must have Network Virtualization installed on your machine. |
| **Relevant tasks** | • "How to Run a Scenario with Network Virtualization" on page 1240<br>• "Add Load Generators to a Scenario" on page 1002<br>• "How to Modify Load Generator Settings" on page 1004 |
| **See also** | • "Network Virtualization Integration" on page 1237<br>• "Network Virtualization Monitoring" on page 1209 |

## User interface elements

| UI Element | Description |
|---|---|
| **Enable Network Virtualization** | Enables network virtualization when running a scenario.<br>Choose between running Network Virtualization:<br>• **Per Group.** Enables you to assign multiple virtual locations for each Vuser group, when several Vuser groups run on the same Load Generator. (Default)<br>• **Per Load Generator.** Lets you assign a single virtual location to each load generator.<br>For more details, see "Per Group versus Per Load Generator" on the next page. |
| **Common Settings** | Opens the Virtual Location Editor that lets you specify the global settings for network virtualization. This includes packet capture details and a list of IP addresses to exclude from network virtualization, applied to all virtual locations. For details, see the Network Virtualization for LoadRunner and Performance Center Help (select the relevant version). |
| **Virtual Location list** | A list of the virtual locations for network virtualization. To add a new virtual location, enter a name and optional description in the next available line.<br>**Note:**<br>• Locations names cannot contain non-English characters.<br>• The following Vuser protocols are not supported by the Per Group mode: Citrix ICA, Java Record/Replay, COM/DCOM and MAPI. For these protocols, specify the virtual default locations in the Load Generators dialog box. For details, see "Load Generators Dialog Box" on page 1019. |
| **Select a virtual location in the grid to use the following:** | |
| **Configure** | Opens the Virtual Location Editor that lets you set the properties of the selected virtual location.<br>For details on using this editor, see the Network Virtualization for LoadRunner and Performance Center Help (select the relevant version). |
| **Duplicate** | Adds a new virtual location with settings identical to the selected entry. |
| **Delete** | Removes the selected virtual location from the list. |
| **Import/Export** | Imports or exports network virtualization settings to or from an XML file. |

## Per Group versus Per Load Generator

You can choose to assign *different* virtual locations to Vuser groups running on a load generator or to assign the *same* virtual location to Vuser groups running on the load generator.

| | |
|---|---|
| **Per Load Generator** mode | Applies the same virtual location to all groups running on the load generator.<br><br>**Note:** Not supported for NV Insights |
| **Per Group** mode | • Enables you to assign several virtual locations to the various Vuser groups available.<br>• Required for NV insights. For more details, see "NV Insights Report" on page 1239.<br>• Does **not** support the following protocols:<br>  • Citrix ICA<br>  • COM/DCOM<br>  • Java Record Replay<br>  • Java Vuser<br>  • MAPI<br>  • user-defined protocols based on the Protocol SDK |

# Troubleshooting and Limitations for Network Virtualization

This section describes troubleshooting and limitations for running Vusers with Network Virtualization.

- When installing the Network Virtualization driver on Windows Server 2008 R2 or Windows 7 SP1, and network adapters are configured as **Teaming** (also known as LBFO), the NV Agent crashes and the active adapter cannot be set.

  **Resolution:** See this knowledge base article: Unable to retrieve Network Virtualization Active Adapter
- The bandwidth utilization measurement is represented in the graphs only when the bandwidth is configured as **Shared Bandwidth**. If the bandwidth is configured as **Individual Bandwidth**, bandwidth utilization data does not appear.
- If you upgraded from a LoadRunner version with no Network Virtualization installed, to the current version with Network Virtualization installed and enabled, you will not see the virtual locations needed to work with Network Virtualization. To enable this feature, do the following:
  a. Click the highlighted globe **Show Virtual Locations Settings** on the Controller toolbar.
  b. In the Virtual Locations Setting dialog box, select the **Enable Network Virtualization** check box, choose **Per Group**, and click **OK**.
  c. In the Virtual Location column, where it says **none** for the virtual location, open the dropdown list and click **Browse**. A window without locations opens.

d. Click **Import**.

e. Browse to **C:\Program Files (x86)\Micro Focus\Network Virtualization\DefaultLocations** and select **DefaultLocations.xml**.

f. You can now see the virtual locations, and can apply them to the scenario.

> **Note:** Once you add a virtual location, and you have modified the **.lrs** file; make a backup of the file if you want to keep the original.

- For very large tests, the report generation time may be slow.
- If the Network Virtualization service is restarted during a scenario run, the network virtualization may fail. Check the service and restart the scenario run. For Network Virtualization specific limitations and system requirements, see the Network Virtualization for LoadRunner and Performance Center Help (select the relevant version).
- Monitoring over a firewall is not supported for scenarios with network virtualization.
- If WinInet replay is enabled in the runtime settings for a Web Vuser (**Replay > Runtime Settings > Internet Protocol > Preferences > Advanced**), you must run the Vuser in **per Load Generator** mode when using Network Virtualization—running in **per Group** mode will cause the script to fail.
- Network Virtualization integration does not comply with all of the accepted Internationalization (I18N) conventions.
- The Network Virtualization software may consume large amounts of memory, since the technology delays traffic and captures traffic for later analysis. To verify that the load generator machine has sufficient memory, compare the load generator memory consumption with and without the virtualization.
- When running scenarios with virtual locations, Vusers must communicate with the application servers over IPv4. Network Virtualization emulation is not supported for IPv6 network traffic.
- You cannot run a scenario on the same load generator from two different Controller machines, if they both have Network Virtualization enabled.
- Locations names cannot contain non-English characters.
- In Windows 7, if the number of open filters is exceeded, the Network Virtualization setup may fail. **Solution:** Increase the value of the registry key:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Network\MaxNumFilters
```

# Service Virtualization Integration

As part of your system-wide load test, you will need to test all of the services that partake in your business process. Some business processes contain services that are not available. For example, your business process might include a service that is still in development or incurs a cost, such as processing a credit card through a third-party vendor.

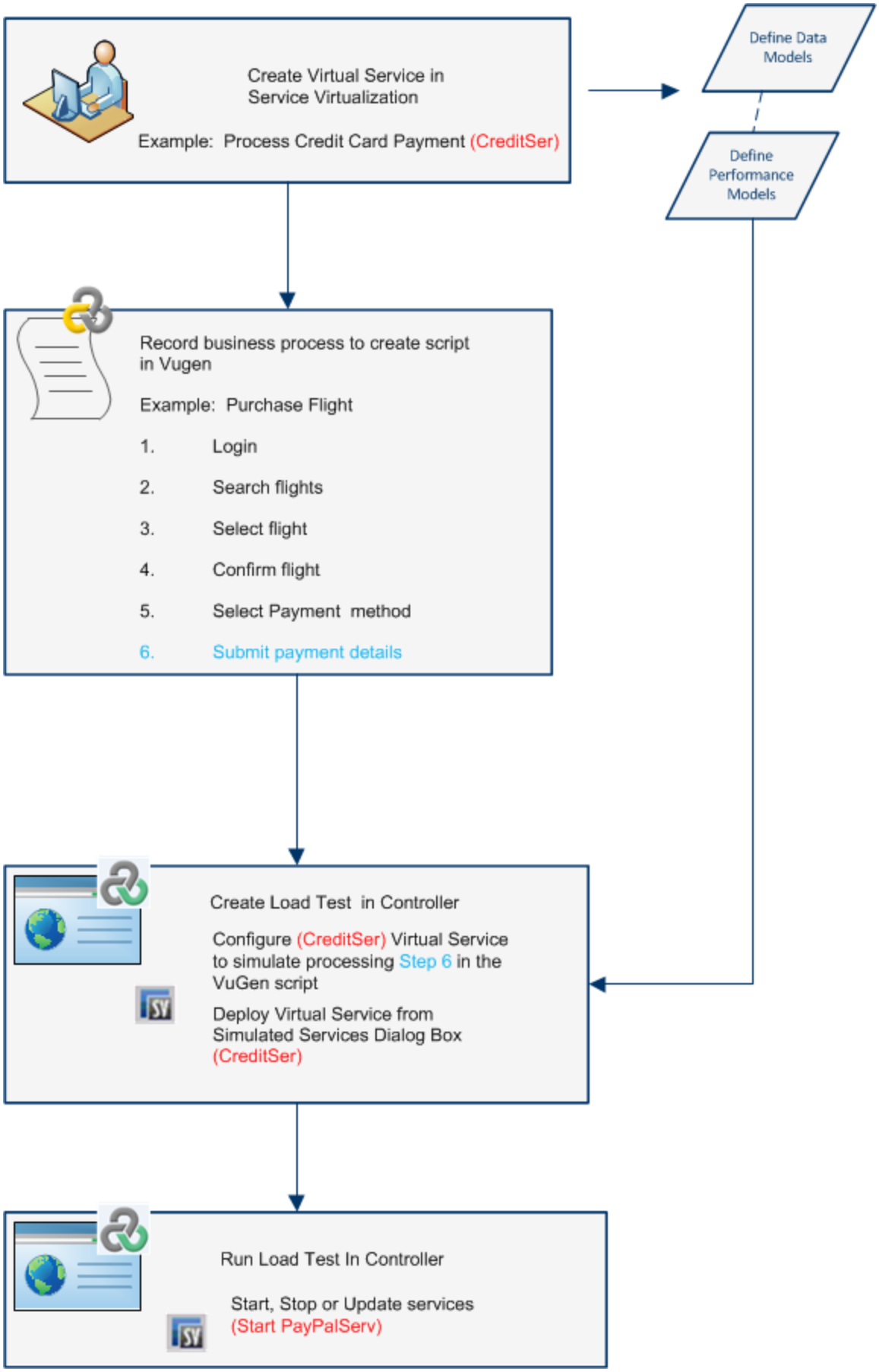
To facilitate load testing these business processes, LoadRunner Controller integrates with Service Virtualization. This integration enables LoadRunner users to define services that will be virtualized during test execution so that the tested business process is performed as required.

By using a virtual service, you can load test and replace actual services with virtual services that mirror both the data model and the performance model.

To learn more about Service Virtualization, see the Service Virtualization site.

# Workflow

The workflow below explains how Service Virtualization integrates with LoadRunner processes:


Create Virtual Service in Service Virtualization
Example: Process Credit Card Payment (CreditSer)
Define Data Models
Define Performance Models
Record business process to create script in Vugen
Example: Purchase Flight
1. Login
2. Search flights
3. Select flight
4. Confirm flight
5. Select Payment method
6. Submit payment details
Create Load Test in Controller
Configure (CreditSer) Virtual Service to simulate processing Step 6 in the VuGen script
Deploy Virtual Service from Simulated Services Dialog Box (CreditSer)
Run Load Test In Controller
Start, Stop or Update services (Start PayPalServ)

# Virtual Service Locks

When your scenario contains virtualized services, these services are automatically locked and cannot be used by other users while the scenario is running. Virtualized services may locked by you or other users for editing, simulation, or deployment.

- If your scenario includes a service that has been locked by another user, the run will fail and the scenario will send an error message to the Output pane.
- If your scenario is running and you update the scenario to include a service that has been locked by another user, the scenario continues running and sends a warning message to the Output window.

# Virtual Service Modes

The following modes apply to the virtual services.

| | |
|---|---|
| **Learning mode** | The virtual service works as a proxy to record and learn the behavior of a real service. The virtual service forwards the real communication between a client and a service. In this mode, any communication through the virtual service is added to the virtual service's simulation models. |
| **Standby mode** | The virtual service redirects requests to the real service, and redirects responses from the real service back to the client. The virtual service is not learning, and not simulating. |
| **Simulating mode** | The virtual service responds to client requests according to learned behavior. This is the main use of the virtual service, and the mode you use for testing purposes. |

### See also:

- "Use Virtual Services in a Scenario" below
- "Service Virtualization Setup Dialog Box" on page 1253
- "Service Virtualization Runtime Dialog Box" on page 1254
- "Service Virtualization Monitors" on page 1255

# Use Virtual Services in a Scenario

This topic describes how to incorporate virtual services into your load test scenario, so that you can run tests with virtual services instead of actual services.

## Prepare the environment

In order to work with virtual services in LoadRunner, you must first install Service Virtualization and define the virtual services you need. You may also need to perform some configuration for the application that you are testing.

### To prepare the environment:

1. If Service Virtualization is not installed in your LoadRunner environment, download it from the Service Virtualization site (free trial available). After installation, see the Service Virtualization Help Center for setup information.
2. Create your virtual services in Service Virtualization Designer. For details, see Virtual Services in the Service Virtualization Help Center.
3. If you need a virtual service to be running while you record your script, open Service Virtualization and start the virtual service before recording.
4. If the application under test has embedded components in the code or configuration files, referring to the Web service, update them to point to the virtual service URL in place of the actual service.

   For example:

   - You have a .NET desktop application that calls a Web service, whose URL is set using a constant:

     ```
     stringURL = http://microfocus.com
     ```

   - You have a service or back-end component that uses the Web service, and the URL is configured in the app.config file.
5. If both of the following apply for the application under test:
   - It is using services that are consumed by accessing UDDI or another registry component (Systinet).
   - The URL is retrieved during runtime.

   Then update the end point URL in UDDI/Systinet to point to the virtual service URL.

## Add virtual services to a scenario

The following steps describe how to work with Service Virtualization from within Controller, enabling you to incorporate virtual services into a load test scenario.

### To add virtual services to a scenario:

1. Define a manual scenario (see "How to Design a Manual Scenario" on page 953) or a goal-orientated scenario (see "How to Design a Goal-Oriented Scenario" on page 951).
2. In the **Design** tab, click the **Service Virtualization** button .
3. In the "Service Virtualization Setup Dialog Box", click the **Add Services** button.
4. In the displayed **Add New Services** dialog box, add projects and services:
   a. Select a source from which to load the services—**Project** or **Running Server**.
   b. For **Project**, browse to the project containing the simulated services you want to run with your load test.
   c. For **Running Server**, specify the URL of the server hosting Service Virtualization.

d. Click **Next**. If your project or service requires authentication or decryption, you are prompted to provide the details.

e. Click **Next**. After a short wait, the project and its corresponding services are displayed in the Service Virtualization Setup dialog box.

> **Note:** A service cannot be deployed when it is added from a running server, but you can switch the modes and deploy it at a later time, when it is no longer locked. For information on modes and locks, see "Service Virtualization Integration" on page 1247.

5. In the Service Virtualization Setup dialog box, select the check box adjacent to each virtual service you want to run.
6. Click the <**project name**> link for each relevant parent project, and verify that the address of the virtualization server is correct. If required, you can change the server address.
7. For each selected service, select the relevant **Data Model** and **Performance Model** information to associate with the virtual service.
8. For each selected service, select a **Mode**, or use the **Stand-By** or **Simulation** options to set the mode for all selected services.
9. Click **Save Setup** to save your changes.

## Manage and monitor virtual services during the test

During your load test run, you can control virtual service deployment. You can also view monitors, to analyze the status and performance of the simulated services during the test run.

### To manage and monitor virtual services:

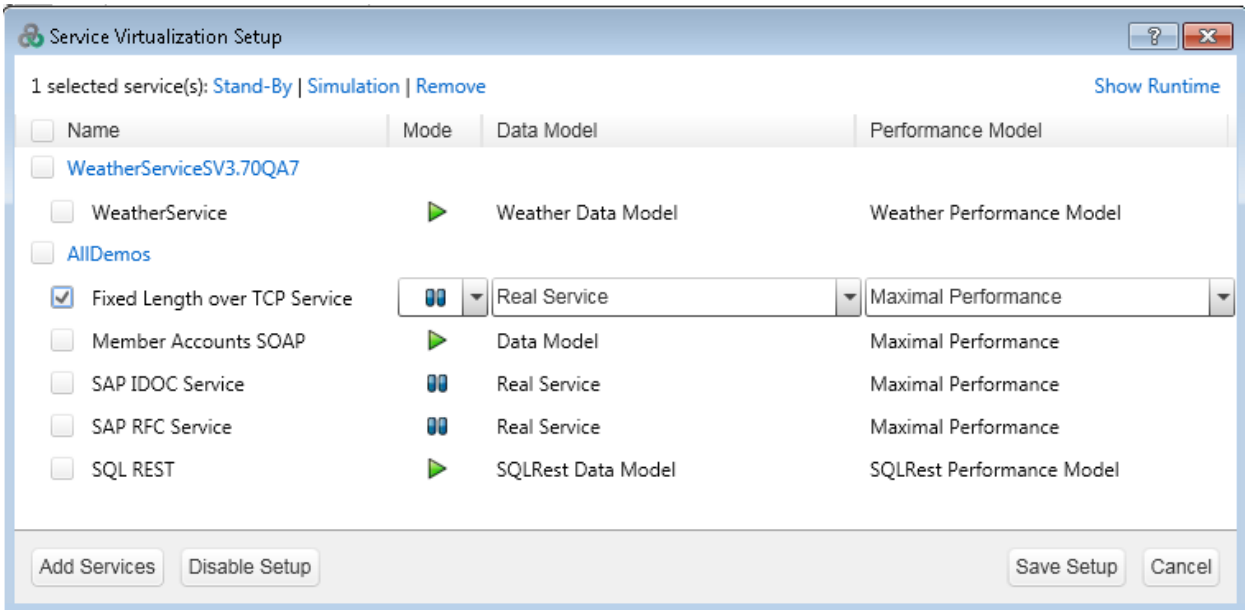
1. Run the scenario. For details, see "How to Run a Scenario" on page 1094.
2. Use the "Service Virtualization Runtime Dialog Box" to manage the virtual services. You can view information and problems related to the services, and use the toolbar links to control the services: **Simulate**, **Stop**, **Unlock**, **Undeploy**.
3. To add or update a service during your load test, return to the **Design** tab and add a new service as described above, using the "Service Virtualization Setup Dialog Box".
4. Monitor virtual service behavior using the online monitors. For details, see "Service Virtualization Monitors" on page 1255.

### See also:

- "Service Virtualization Integration" on page 1247
- "Service Virtualization Monitors" on page 1255
- "Service Virtualization Graphs" on page 1465

# Service Virtualization Setup Dialog Box

The Service Virtualization Setup dialog box allows you to add and configure the virtual services to use in your test.




| | |
|---|---|
| **To access** | Click the Virtualized Service Settings button from the Controller toolbar. |
| **Relevant tasks** | "Use Virtual Services in a Scenario" on page 1250 |
| **See also** | "Service Virtualization Runtime Dialog Box" on the next page |

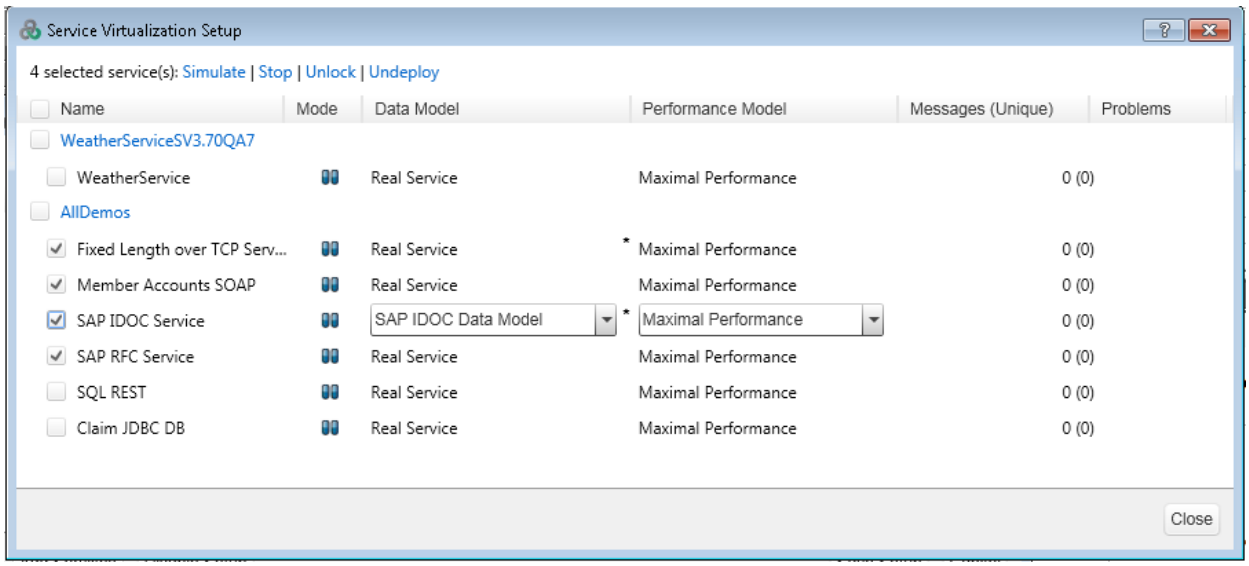
User interface elements are described below(unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Stand-By** | Places all selected services in stand-by (pass through) mode. The service remains deployed. The Mode icons change to . |
| **Simulation** | Places all selected services on in Simulation mode, deploying them when the test begins. The Mode icons change to . |
| **Remove** | Remove the selected service from the list of virtualized services. |
| **Show Runtime** | Opens the Service Virtualization Runtime dialog box, displaying the states of the services in runtime. |

| | |
|---|---|
| **<Virtualization services>** | Lists the virtualized services displaying the following information:<br>• **Name.** The name of the virtual service.<br>• **Mode.** The mode of the service—Stand-By (non-active) or Simulation (active).<br>• **Data Model.** The defined set of data sent from the client (request) to the application server and the expected data sent back to the client (response).<br>• **Performance Model.** The model that defines how fast the response and request are processed. |
| Add Services | Opens the Add New Services dialog box, allowing you to add services defined in a project or located on a running server. |
| Disable Setup<br>Enable Setup | Disables or enables the setting up of services. This is useful if you do not want them to be accidentally simulated during your test run. |
| Save Setup | Saves the current setup and closes this window. When you reopen this dialog box, it will show your most recent list of services, their selections, and their modes. |

# Service Virtualization Runtime Dialog Box

The Service Virtualization Runtime dialog box lets you interact with the services during the test.



| | |
|---|---|
| **To access** | Do one of the following:<br>1. From the **Design** tab: Click the Service Virtualization button on the Controller toolbar. In the Service Virtualization Setup dialog box, click **Show Runtime** in the top right corner.<br>2. From the **Run** tab, during a scenario, view the **Scenario Status** pane and click the **ON** or **OFF** link in the Service Virtualization row. |
| **Relevant tasks** | "Use Virtual Services in a Scenario" on page 1250 |
| **See also** | "Service Virtualization Setup Dialog Box" on page 1253 |

User interface elements are described below(unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<#> selected service(s)** | The number of active services selected in the list below. |
| **Simulate** | Includes the selected services in the test. The Mode icons change to . |
| **Stop** | Removes the selected services from the test. The Mode icons change to . |
| **Unlock** | Removes the lock from the selected services. |
| **Undeploy** | Undeploys the selected services. |
| **<Virtualization services>** | Lists the virtualized services displaying the following information:<br>• **Name.** The name of the virtual service.<br>• **Mode.** The mode of the service—Stand-By (non-active) or Simulation (active).<br>• **Data Model.** The defined set of data sent from the client (request) to the application server and the expected data sent back to the client (response).<br>• **Performance Model.** The model that defines how fast the response and request are processed.<br>• **Messages.** Indications about the service.<br>• **Problems.** A short description of the problems related to the service. |

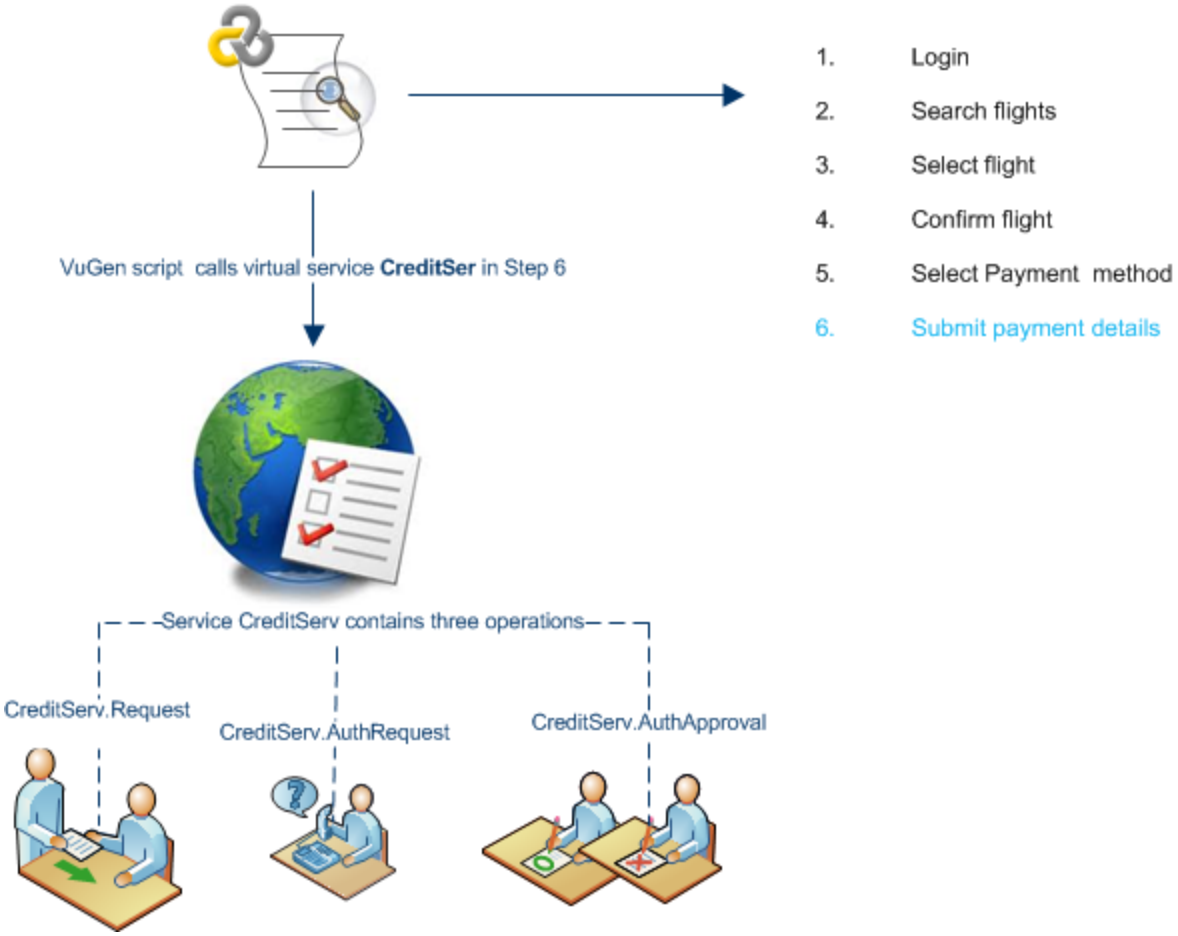
# Service Virtualization Monitors

The Service Virtualization monitors enable you to analyze the status and performance of the simulated services during the load test run.

For example, you may integrate a payment process service containing the following services and operations into your script:

| Type | Example Name | Example Description |
|---|---|---|
| Service | CreditServ | Process online credit card payments |
| Operation | CreditServ.PurchaseRequest | Cardholder requests product/service |
| Operation | CreditServ.AuthorizationRequest | Merchant request authorization for payment |
| Operation | CreditServ.AuthorizationApproval | Credit card company authorizes or denies payment |

The online monitors measure the overall behavior of the service and each operation. The following flowchart illustrates using a service and operations for credit card payment.



| Monitor | Measurements | Description |
|---|---|---|
| Operations | Average Response Time | Average response time of virtual service in milliseconds. |
| | Hit Rate | Number of requests per second of the virtual service operation. |
| | Throughput | Data sent and received by virtual service operation measured in megabytes. |
| Services | Average Response Time | Average response time of virtual service in milliseconds. |
| | Data Simulation Accuracy | Accuracy of data model emulation on virtual service, displayed as a percentage. |
| | Hit Rate | The number of requests per second of the virtual service. |
| | Performance Simulation Accuracy | Accuracy of performance model emulation on virtual service, displayed as a percentage. |
| | Throughput | Data sent and received on virtual service measured in megabytes. |

**Note:** If you intend to use the monitor credentials together with SSL credentials, the security policy may prevent the project from being added to Controller. To overcome this issue, add the **SV Operators** group to **Performance Monitor Users**.

### See also:

- "Service Virtualization Graphs" on page 1465

# J2EE/.NET Diagnostics

**Supported for the following protocols:** Web - HTTP/HTML, Java over HTTP, Oracle - Web, SAP - Web, Siebel - Web, TruClient - Web, Web Services, and Flex.

The J2EE/.NET Diagnostics integration with LoadRunner allows you to monitor and analyze the performance of Java 2 Enterprise Edition (J2EE), .NET-connected, and other complex environments.

By configuring a LoadRunner scenario to use J2EE/.NET Diagnostics, you can instruct LoadRunner to capture server requests which occur outside the context of a Vuser transaction.

The benefit of this functionality is that it can capture calls into a back-end virtual machine even when:

- The probe is not capturing RMI calls
- RMI calls cannot be captured (perhaps because an unsupported application container is being used)
- The application uses some other mechanism for communications between multiple virtual machines

For more information about working with diagnostics for J2EE and .NET, see the *Diagnostics User Guide* available on the Diagnostics installation media.

### Next steps:

- Install the Diagnostics Add-in
- Configure a scenario to use J2EE/.NET Diagnostics
- View Diagnostics data during a scenario run
- View Diagnostics results offline

# How to Install the LoadRunner J2EE/.NET Diagnostics Add-in

The integration between J2EE/.NET Diagnostics and LoadRunner requires the following installations:

- Diagnostics Server
- An Application server with diagnostic probes, connected to the Diagnostics server, with diagnostic probes on the application to monitor
- LoadRunner Controller and load generator to run the scenario
- LoadRunner Virtual User Generator (optional)
- LoadRunner Analysis (optional)
- LR Add-In for J2EE/.NET Diagnostics installed on the Controller machine (described below)

For details about installing the J2EE/.NET Diagnostics components, see the *Diagnostics Server Installation and Administration Guide* on the Diagnostics media.

After you have the relevant products installed, you complete the integration with LoadRunner by doing the following:

1. Close the Controller if it is open.
2. Copy the **LR_Addin** folder from the Diagnostics media to the LoadRunner Controller machine, and run the **setup.exe** file.
3. If this is the first time you are running a scenario with Diagnostics, or if you changed the server or port information do the following:

   On the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > Diagnostics for J2EE and .NET Setup**.

   On machines with icon-based desktops, such as Windows 8, search for **J2EE** and select the **Diagnostics for J2EE and .NET Setup**.

   Indicate the server and port for the Diagnostics server and the authentication information. You can use a server name or IP address.

Click **Test** to verify that the connection is valid.

For details, see "Diagnostics for J2EE/.NET Setup Dialog Box" on page 1261.

# How to Configure a LoadRunner Scenario to use J2EE/.NET Diagnostics

This task describes how to capture J2EE/.NET diagnostics metrics in a LoadRunner scenario and how to select the probes that will be included in the scenario.

1. Prerequisites
   - Make sure that the application server you are monitoring is started.
   - Make sure that the load test scenario is not already running.
   - Make sure to assign a unique transaction name for each scenario.
2. Prepare a VuGen script and scenario
   - Prepare a VuGen script that emulates user's behavior on your application on the application server.
   - Prepare a LoadRunner scenario that runs the script.
3. Enable J2EE/.NET Diagnostics

   In the Controller, select **Diagnostics** > **Configuration** to open the Diagnostics Distribution dialog box. Then select **Enable the following diagnostics**. For details, see "Diagnostics Distribution Dialog Box" on page 1261.
4. Set a distribution percentage

   Specify the percentage of Vusers for which you want to collect J2EE/.NET Diagnostics data.
5. Select the Probes
   a. In the Diagnostics Distribution dialog box, locate the **Online & Offline Diagnostics** section and click **Configure**. The Enable J2EE/.NET Diagnostics Configuration dialog box opens. For details, see "J2EE/.NET Configuration Dialog Box" on page 1263.
   b. Select the **Enable J2EE/.NET Diagnostics** check box.
   c. Select the probes that you want to include in the scenario run. Click **OK**.
6. View the graphs
   a. Begin the scenario run. In the Controller, click the **Diagnostics for J2EE/.NET** tab to view the online graphs.
   b. After the scenario run is complete, open LoadRunner Analysis to view the **J2EE & .NET Diagnostics** Graphs. If the graph is not visible, select **Graph > Add New Graph**. For details, see "J2EE and .NET Diagnostics Graphs" on page 1417.

# How to View J2EE/.NET Diagnostics Data in LoadRunner During a Scenario Run

This task describes how to view diagnostics data for J2EE/.NET Diagnostics in LoadRunner for the whole scenario or for a specific transaction during a scenario run.

### To view diagnostics data for the whole scenario:

In the Controller, select the **Diagnostics for J2EE/.NET** tab. J2EE/.NET Diagnostics opens, displaying the **Scenario Summary** dashboard view.

The **Scenario Summary** dashboard view displays monitoring versions of the transactions, server requests, load, and probe views for the current run.

> **Note:** If you move to another tab during the scenario run and then return to the **Diagnostics for J2EE/.NET** tab, the last screen that you viewed will be displayed.

### To view Diagnostics Data for a Specific Transaction:

1. Select one of the Transaction graphs (for example, **Transaction Response Time**), to open the graph.
2. Right-click the relevant transaction in the graph legend and select **Show J2EE/.NET server side**.

   J2EE/.NET Diagnostics opens, displaying the Transactions view, which contains performance metrics and drill-down options for the selected transaction.

   For more information about interpreting data in the Diagnostics Transactions view, see the *Diagnostics User Guide*.

# How to View Offline J2EE/.NET Diagnostics Results

1. Open Analysis: In the Run tab of the Controller, select **Results > Analyze Results**, or click the **Analyze Results** button .
2. View results in the Analysis diagnostics graphs.

   You can use the Analysis diagnostics graphs and reports to view the performance data and drill down to pinpoint problem areas in any layer of the application.

   For specific information about J2EE/.NET diagnostics graphs, see "J2EE and .NET Diagnostics Graphs Overview" on page 1417.

# Diagnostics for J2EE/.NET Setup Dialog Box

This dialog box enables you to update the LoadRunner configuration settings for J2EE/.NET Diagnostics.

| | |
|---|---|
| **To access** | On the LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Tools > Diagnostics for J2EE and .NET Setup**. On machine with icon-based desktops, such as Windows 8, search for **J2EE** and select the **Diagnostics for J2EE and .NET Setup**. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| Test | Click to verify that you entered the correct information for the Diagnostics Server in Commander mode and that there is connectivity between the server and LoadRunner. |
| **Login** | The user name with which you log in to Diagnostics.<br>**Default:** admin<br><br>**Note:** The user name that you specify should have **view**, **change**, and **execute** privileges. For more information about user privileges, see the *Diagnostics Server Installation and Administration Guide*. |
| **Password** | Enter the password with which you log in to Diagnostics.<br>**Default:** admin |
| **Port** | Enter the port number used by the Diagnostic server in Commander mode.<br>**Default:** 2006<br><br>**Note:** LoadRunner does not support communication with the Diagnostics Server in Commander mode using HTTPS. |
| **Server Name** | Enter the name of the machine that is to host the Diagnostics Server in Commander mode. |

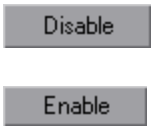
# Diagnostics Distribution Dialog Box

This dialog box enables you to enable and customize the Diagnostics integration.

| | |
|---|---|
| **To access** | In the Controller, select **Diagnostics** > **Configuration** |

| | |
|---|---|
| **Important information** | • The Diagnostics Distribution dialog box is disabled during scenario execution. You must enable and configure the diagnostics modules before running the scenario.<br>• The settings that you configure are per scenario. All scripts in the scenario will run under the same diagnostics configuration. |
| **Relevant tasks** | "How to Configure a LoadRunner Scenario to use J2EE/.NET Diagnostics" on page 1259 |

User interface elements are described below:

| UI Element | Description |
|---|---|
|  | Enables/Disables Web Page Diagnostics graphs. |
| | Click to enable and configure the relevant diagnostics graphs. |
| | Indicates that the diagnostics type is disabled. |
| | Indicates that the diagnostics type is enabled. |
| **Enable the following diagnostics** | Enables LoadRunner to generate offline Web Page Diagnostics graphs, and online and offline J2EE & .NET Diagnostics graphs. |

| | |
|---|---|
| **For X% of all the relevant Vusers in the current scenario** | Specify the percentage of Vusers for which you want to collect diagnostics data. This value determines how many of the transactions on the application server are reported to the Controller. Reducing this percentage will reduce the overhead on the application server for Web Page, and J2EE & .NET Diagnostics.<br><br>**Example:** If you enter a sampling value of 25% and run 12 Vusers in **group1**, 8 Vusers in **group2**, and 1 Vuser in **group3**, diagnostics data will be collected for 3 Vusers in **group1**, 2 Vusers in **group2**, and 1 Vuser in **group3**.<br><br>**Note:** The minimum percentage of Vuser sampling allowed is 1%, or 1 Vuser per group, whichever is more.<br><br>The maximum percentage allowed is the lowest of the **Max. Vuser Sampling** values of all the selected diagnostics types.<br><br>**Example:** If you enable **Web Page** (max 10%) and J2EE/.NET (max 100%) diagnostics, the percentage of Vuser participation for J2EE/.NET Diagnostics cannot exceed 5%. |
| **Offline Diagnostics** | Generates offline Web Page Diagnostics graphs.<br>The maximum percentage of Vusers for which diagnostics data can be collected is 10%.<br>**Default:** Enabled<br>**See also:** "Web Resources Graphs Overview" on page 1515 |
| **Online & Offline Diagnostics** | Generates online and offline J2EE/.NET Diagnostics graphs.<br>The maximum percentage of Vusers for which J2EE/.NET Diagnostics data can be collected is 100% of the amount of Vusers selected in the **For X% of all the relevant Vusers in the current scenario** setting.<br>To enable and configure J2EE/.NET Diagnostics, click **Configure**. For details, see "J2EE/.NET Configuration Dialog Box" below. |

# J2EE/.NET Configuration Dialog Box

This dialog box enables you to set up the J2EE/.NET Diagnostics module.

| | |
|---|---|
| **To access** | Select **Diagnostics** > **Configuration.** In the **Online & Offline Diagnostics** section, click **Configure**. |
| **Important information** | The dialog box is read- only while a scenario is running. |

| | |
|---|---|
| **Relevant tasks** | "How to Configure a LoadRunner Scenario to use J2EE/.NET Diagnostics" on page 1259 |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Enable J2EE/.NET Diagnostics** | Enables J2EE/.NET Diagnostics and allows you to configure the J2EE/.NET Diagnostics settings |
| **Select probes table** | Selects a probe for monitoring. At least one probe must be selected. Clear the check box to disable a probe for the duration of the scenario.<br>**Name.** The name of the probe.<br>**Group.** The probe group.<br>**Host Name.** The host the probe is running on (or the application server on which the probe is installed) la.<br><br>**Note:** If you upgraded your Diagnostics installation, probes from existing scenarios may appear with a red status. Clear any probes that appear in red. |
| **MI Listener server** | Enter the name of the MI Listener server when the Diagnostics server is located behind a firewall. |
| **Monitor server requests** | Select to capture a percentage of server requests which occur outside the context of any Vuser transaction. For more information, see "How to Install the LoadRunner J2EE/.NET Diagnostics Add-in" on page 1258.<br><br>**Note:**<br>• The server requests will be captured at the same percentage that was selected for the percentage of Vusers in the Diagnostics Distribution dialog box.<br>• Enabling this option imposes an additional overhead on the probe. |
| **Troubleshoot Diagnostics for J2EE/.NET connectivity** | Click to open the Diagnostics System Health Monitor to enable you to investigate any connectivity issues between the Diagnostics components. |

# Troubleshooting and Limitations for Controller

> **Note:** To troubleshoot error messages, see Troubleshooting Error Messages in the online Help Center.

## Vusers do not stop with scenario

If your script contains code that is not part of the LoadRunner API, the Vusers may not stop running immediately after you stop the scenario.

## Secure Communication with TLS (SSL)

LoadRunner uses an version of OpenSSL which enforces additional constraints for secure communication.

OpenSSL (from 1.1.x) requires that CA certificates include the following basic constraint:**CA (Certificate Authority)**

**Upgrade:** If you are upgrading from an earlier version than 12.60, and your CA certificate was created in a previous version of LoadRunner using any of the LoadRunner tools, you can no longer use these certificates because they do not include the required constraints. Use the LoadRunner tools to create and install a new CA certificate and new TLS certificates on each LoadRunner machine. For details, see Secure Communication with TLS (SSL).

**Organization's CA:** If your CA certificate is provided by your organization, verify that the certificate includes the constraints described above. If the required constraints are not defined, your IT department should provide updated CA and TLS certificates to install on the LoadRunner machines.

**To verify on Windows:**

1. Double-click the CA certificate and select the **Details** tab.
2. Select **Extensions only** from the **Show** list, then select **Basic Constraints** and check that the constraints are defined in the properties area.

**To verify on Linux:** Use the following command to view certificate information: **openssl x509 -in <CA-filename> -noout -text**.

## Linux Machine Issues

This section describes how to troubleshoot problems with a test run on a Linux machine. For more information, see the Linux section in the *LoadRunner Installation Guide*.

### Linux system limits exceeded

Each VUser may open many files and TCP connections during a run. When running many VUsers, a run may fail because Linux system-wide limits are exceeded. To address this, you can adjust Linux settings. For example:

- You can increase the soft and hard file limits in `/etc/security/limits.conf`.
  The following values are recommended, but you should experiment to find the best values for your needs.
  - soft: 65535
  - hard: 131070
- You can expand the port range. The recommended maximum range is:
  `sysctl -w net.ipv4.ip_local_port_range="1024 65535"`
- You can also increase available ports by decreasing the minimum time that a socket remains in `TIME_WAIT` state. (Default: `net.ipv4.tcp_fin_timeout = 60`)

### Using Web-based protocols on Linux

When running web-based scripts on Linux, considering the following:

- WinInet replay does not support Linux
- You cannot use Windows native NTLM authentication
- The **web_set_certificate_ex** does not work with the **certIndex**/**certThumbprint** parameters
- When working with Click and Script protocols, ActiveX and Applets are not supported.
- The **web_set_certificate** function is not supported.
- BinaryXML DFE is not supported.
- RDP and Silverlight protocols are not supported.
- Other web-based protocols have the same limitations as described above for Web.

# Shellshock Vulnerability

This section describes how to check if your load generator machine is vulnerable to the Shellshock issue, and guides you high to apply a fix.

This fix applies to Linux load generator machines, and Azure cloud machines, for which you create custom images. For details, see "Create Custom Image for Load Generators on the Cloud" on page 992.

1. Check the machine for vulnerability

   Run the following command:

   ```
   env 'VAR=() { :;}; echo Bash is vulnerable!' 'FUNCTION()=() { :;}; echo Bash is vulnerable!' bash -c "echo Bash Test"
   ```

- If the result is: `Bash is vulnerable!`, then you need to apply the fix.
- If the result is: `Bash Test`, your machine is not vulnerable, and no further action is required.

2. Apply the fix

   To apply the fix, you update Bash to latest version. Run the following command:

   ```
   apt-get:sudo apt-get update && sudo apt-get install --only-upgrade bash
   ```

   - If the result is: `Bash is vulnerable!`, then you need to apply the fix.
   - If the result is: `Bash Test`, your machine is not vulnerable, and no further action is required.

   For more information about the affected releases, see the following websites:

   - https://wiki.ubuntu.com/Releases
   - https://www.debian.org/releases
   - https://www.digitalocean.com/community/tutorials/how-to-protect-your-server-against-the-shellshock-bash-vulnerability

3. Recheck the vulnerability

   Check the vulnerability again as described in the first step.

# Scenario run automation

**Problem:** I set up the CLI tool to automate my scenario run but it did not run/is not running.

**Possible solutions:**

- Only one instance of Controller can run at a time on the machine. Make sure that no instance of Controller is running on the machine.
- Make sure the credentials of the admin user that will be used to run Controller are defined in the CLI tool. For details, see "Running Scenarios using the Command Line Interface (CLI) Tool" on page 1114.

# Analysis

Analysis is a component of LoadRunner, enabling you to create graphs and reports for analyzing system performance after a test run.

To learn more, see "Introducing Analysis" on page 1269.

# Introducing Analysis

Welcome to LoadRunner Analysis, the tool for gathering and presenting load test data. When you execute a load test scenario, Vusers generate result data as they perform their transactions.

The Analysis tool provides graphs and reports enabling you to view and understand the data, and analyze system performance after a test run.

### What do you want to do?

- Set up Analysis
- Create graphs
- Generate reports
- Define a Service Level Agreement

### See also:

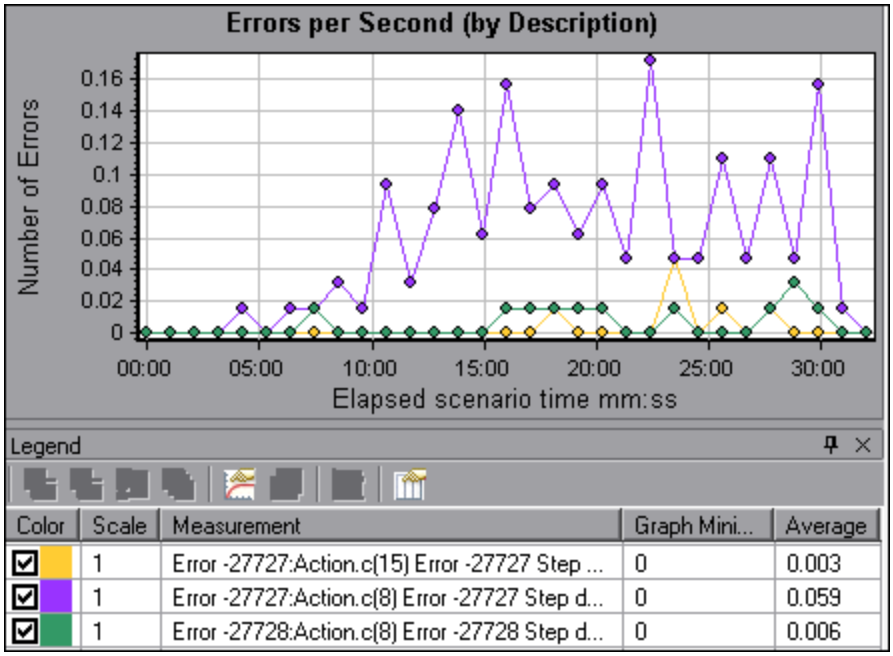
- Results overview
- Analysis API

# Results Overview

To monitor the scenario performance *during* test execution, use the online monitoring tools described in "Monitoring Load Test Scenarios" on page 1171.

To view a summary of the results *after* test execution, use one or more of the following tools:

- **Vuser log files.** These files contain a full trace of the load test scenario run for each Vuser. These files are located in the scenario results folder. (When you run a Vuser script in standalone mode, these files are stored in the Vuser script folder.)
- **Controller Output window.** The output window displays information about the load test scenario run. If your scenario run fails, look for debug information in this window.
- **Analysis Graphs.** Standard and protocol-specific graphs help you determine system performance and provide information about transactions and Vusers.
  - You can compare multiple graphs by combining results from several load test scenarios or merging several graphs into one.
  - Each graph has a legend which describes the metrics in the graph. You can also filter your data and sort it by a specific field.




- **Analysis Graph Data and Raw Data Views.** These views display the actual data used to generate the graph in a spreadsheet format. You can copy this data into external spreadsheet applications for further processing.
- **Analysis Reports.** This utility enables you to generate a summary of each graph. The report summarizes and displays the test's significant data in graphical and tabular format. You can generate reports based on customizable report templates.

# Analysis Toolbars

This section describes the buttons that you access from the main Analysis toolbars.

## Common Toolbar

This toolbar is always accessible from the toolbar at top of the page and includes the following buttons:

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Create a new session. |
| | Open an existing session. |
| | Generate a Cross Result graph. |

| UI Element | Description |
|---|---|
| | Save a session. |
| | Print item. |
| | Create an HTML report. |
| | View runtime settings. |
| | Set global filter options. |
| | Configure SLA rules |
| | Analyze a transaction. |
| | Undo the most recent action. |
| | Reapply the last action that was undone. |
| | Apply filter on summary page |
| | Export Summary to Excel |

## Graph Toolbar

This toolbar is accessible from the top of the page when you have a graph open and includes the following buttons

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Set filter settings. |
| | Clear filter settings. |
| | Set granularity settings. |

| UI Element | Description |
|---|---|
| | Merge graphs. |
| | Configure auto correlation settings. |
| | View raw data. |
| | Add comments to a graph. |
| | Add arrows to a graph. |
| | Set display options. |

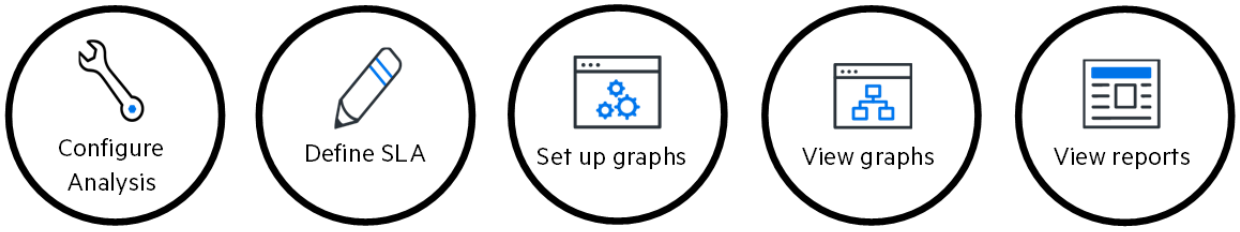
# Analysis API

The LoadRunner Analysis API enables you to write programs to perform some of the functions of the Analysis user interface, and to extract data for use in external applications. Among other capabilities, the API allows you to create an analysis session from test results, analyze raw results of an Analysis session, and extract key session measurements for external use. You can also use the API to launch an application from the LoadRunner Controller at the completion of a test.

To view this help from a LoadRunner machine, from the Windows Start menu, go to **Micro Focus > LoadRunner > Documentation > Analysis API Reference**.

# Workflow

The workflow below displays the key configuration and analysis tasks.




## See also:

- Analysis Basics
- Troubleshooting Analysis

# Analysis Basics

## Creating Analysis Sessions

When you run a load test scenario, LoadRunner stores the runtime data in a result file with an **.lrr** extension. LoadRunner **Analysis** is the utility that processes this data and generates graphs and reports.

When you work with the LoadRunner Analysis, you work within an Analysis *session*. This session contains one or more sets of scenario results (**.lrr** file). Analysis stores the display information and layout settings for the active graphs in a file with an .**lra** extension.

## Starting Analysis

You can open Analysis as an independent application or directly from Controller. To open Analysis as an independent application, choose one of the following:

- From the Windows Start menu, go to **Micro Focus > LoadRunner > Analysis**
- The Analysis shortcut on the desktop

To open Analysis directly from Controller, click the **Analysis** button on the toolbar or select **Results > Analyze Result**. This option is only available after running a load test scenario. Analysis takes the latest result file from the current scenario, and opens a new session using these results. You can also instruct Controller to automatically open Analysis after it completes scenario execution by selecting **Results > Auto Load Analysis**.

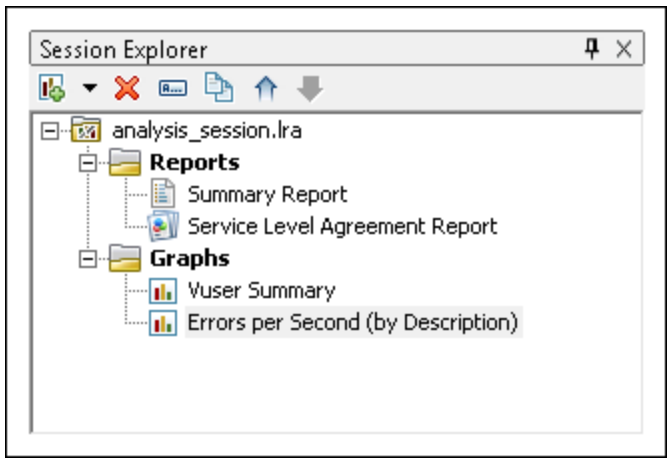
## Collating Execution Results

When you run a load test scenario, by default all Vuser information is stored locally on each Vuser host. After scenario execution, the results from all of the hosts are automatically collated or consolidated in the results folder.

You disable automatic collation by choosing **Results** > **Auto Collate Results** in Controller, and clearing the check mark adjacent to the option. To manually collate results, choose **Results** > **Collate Results**. If your results have not been collated, Analysis will automatically collate the results before generating the analysis data.

# Session Explorer Window

This window displays a tree view of the items (graphs and reports) that are open in the current session. When you click an item in the Session Explorer, it is activated in the main Analysis window.




| | |
|---|---|
| **To access** | Use one of the following:<br>• **Session Explorer**<br>• **Session Explorer > Reports > Summary Report**<br>• **Session Explorer > Reports >Service Level Agreement Report**<br>• **Session Explorer >** **> Analyze Transaction**<br>• **Session Explorer > Graphs** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Add a new graph or report to the current Analysis session. Opens the Open a New Graph dialog box. For details, see "Open a New Graph Dialog Box" on page 1371. |
| | Delete the selected graph or report. |
| | Rename the selected graph or report. |
| | Create a copy of the selected graph. |

# Analysis Window Layouts

This section describes ways to customize the layout of the windows of the Analysis session.

## Open Windows

You can open a window or restore a window that was closed by selecting the name of the relevant window from the **Windows** menu.

## Lock/Unlock the Layout of the Screen

Select **Windows > Layout Locked** to lock or unlock the layout of the screen.

## Restore the Window Placement to the Default Layout

Select **Windows > Restore Default Layout** to restore the placement of the Analysis windows to their default layout.

> **Note:** This option is available only when no Analysis session is open.

## Restore the Window Placement to the Classic Layout

Select **Windows > Restore Classic Layout** to restore the placement of the Analysis windows to their classic layout. The classic layout resembles the layout of earlier versions of Analysis.

> **Note:** This option is available only when no Analysis session is open.

## Reposition and Dock Windows

You can reposition any window by dragging it to the desired position on the screen. You can dock a window by dragging the window and using the arrows of the guide diamond to dock the window in the desired position.

> **Note:**
>
> - Only document windows (graphs or reports) can be docked in the center portion of the screen.
> - **Windows > Layout Locked** must not be selected when repositioning or docking windows.

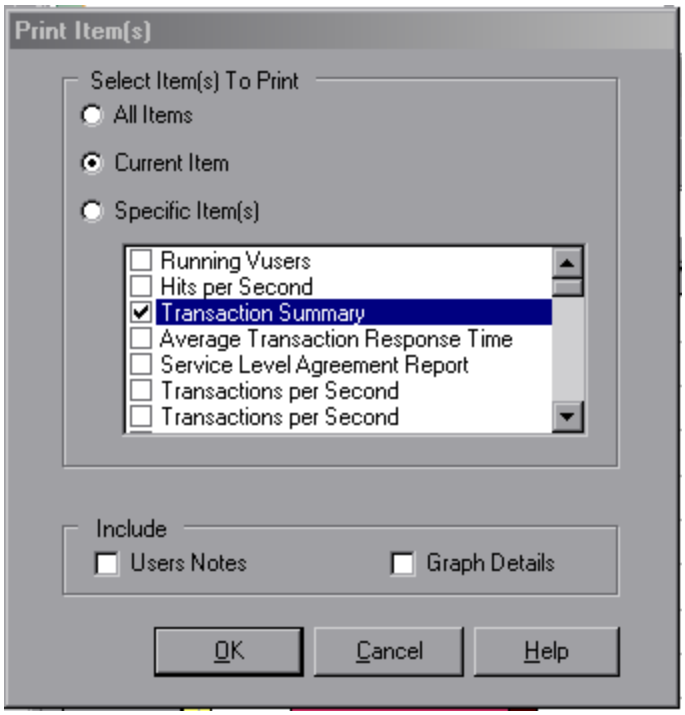
## Using Auto Hide

You can use the Auto Hide feature to minimize open windows that are not in use. The window is minimized along the edges of the screen.

Click the **Auto Hide** button on the title bar of the window to enable or disable Auto Hide.

# Printing Graphs or Reports

This dialog box enables you to print graphs or reports

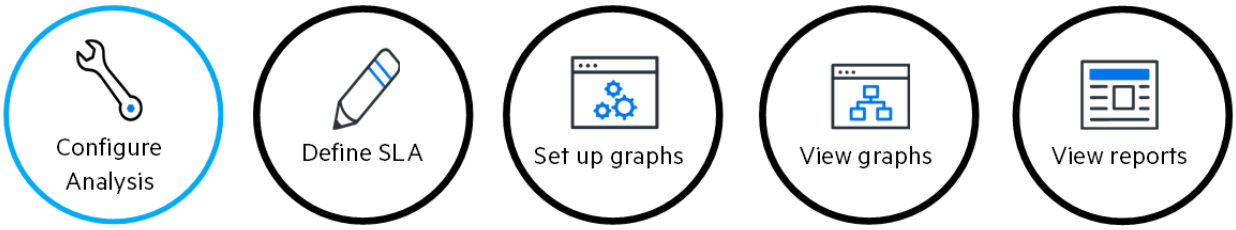



| | |
|---|---|
| **To access** | Do one of the following:<br>• **File > Print**<br>• **Main toolbar >** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Select Items to Print** | • **All Items.** Prints all graphs and reports in the current session.<br>• **Current Item.** Prints the graph or report currently selected in the Session Explorer.<br>• **Specific Item(s).** Select the graphs or reports to print. |
| **Include** | • **User Notes.** Prints the notes in the User Notes window.<br>• **Graph Details.** Prints details such as graph filters and granularity settings. |

# Configuring Analysis




## Summary Data Versus Complete Data

In large load test scenarios, with results exceeding 100 MB, it can take a long time for Analysis to process the data. When you configure how Analysis generates result data from load test scenarios, you can choose to generate complete data or summary data.

**Complete data** refers to the result data after it has been processed for use within Analysis.

**Summary data** refers to the raw, unprocessed data. The summary graphs contain general information such as transaction names and times. Some fields are not available for filtering when you work with summary graphs.

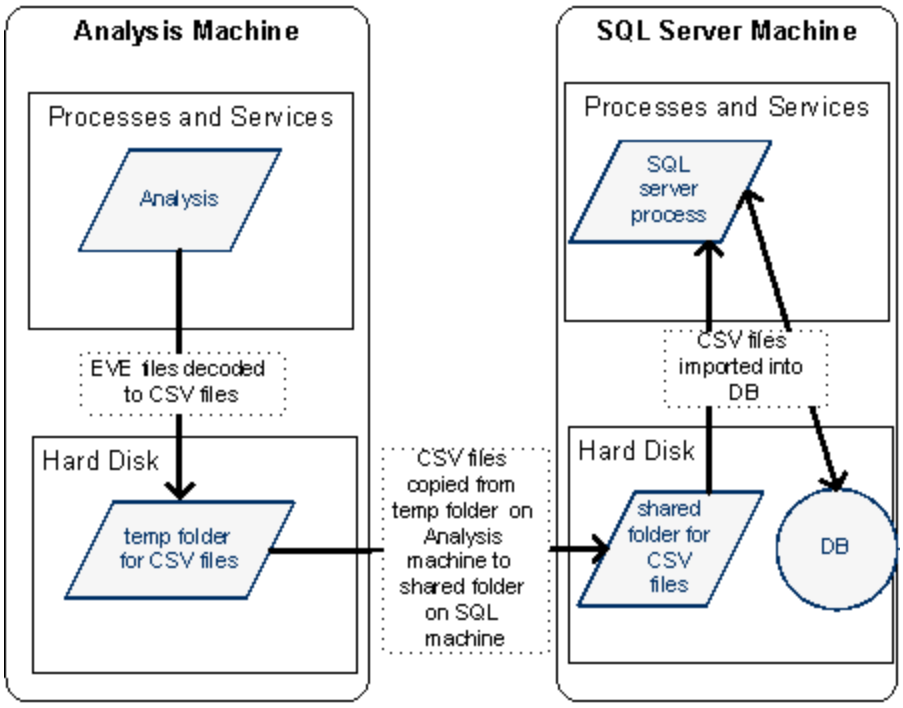
Note that some graphs will not be available when viewing only the summary data.

## Importing Data Directly from the Analysis Machine

If you are using an SQL server to store Analysis result data, you can configure Analysis to import data directly from the Analysis machine.

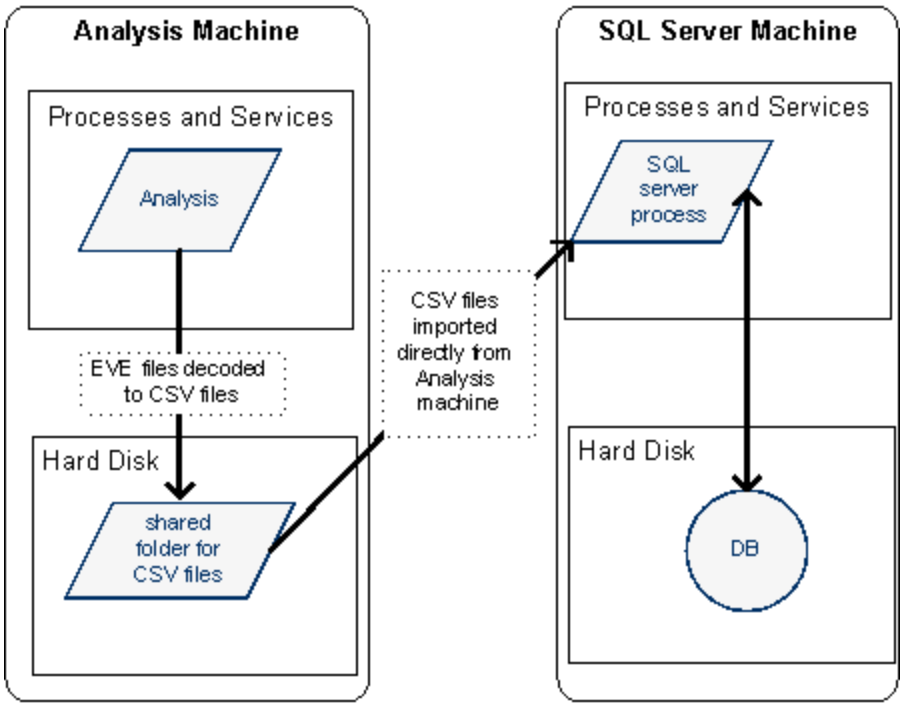
### Importing Data from the SQL Server

If you do not select the option to import data directly from the Analysis machine, Analysis creates CSV files in a local temp folder. The CSV files are copied to a shared folder on the SQL Server machine. The SQL server engine then imports the CSV files into the database. The following diagram illustrates the data flow:




## Importing Data from the Analysis Machine

If you selected the option to import data directly from the Analysis machine, Analysis creates the CSV files in a shared folder on the Analysis machine and the SQL server imports these CSV files from the Analysis machine directly into the database. The following diagram illustrates the data flow:



# How to Configure Settings for Analyzing Load Test Results

The following steps describe how to configure certain Analysis settings that significantly impact the way in which Analysis analyzes load test results.

## Configure how Analysis processes result data

You define how Analysis processes result data from load test scenarios in the **Tools > Options > Result Collection** tab. For example, you can configure how Analysis aggregates result data, to what extent the data is processed, and whether output messages are copied from the Controller. For details on the user interface, see "Result Collection Tab (Options Dialog Box)" on page 1282.

## Configure template settings

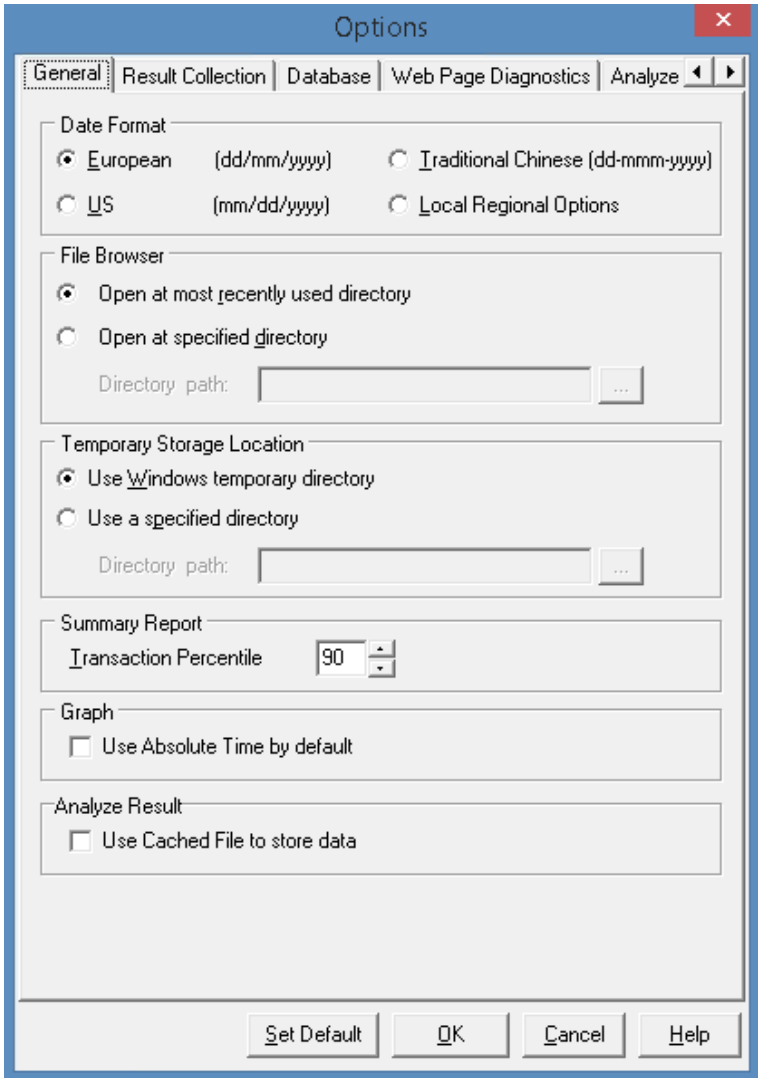
For details on the user interface, see "Apply/Edit Template Dialog Box" on page 1323.

## Configure analysis of transactions

You configure how transactions are analyzed and displayed in the summary report in the **Summary Report** area of the **Tools > Options > General** tab. For details, see the description of "General Tab (Options Dialog Box)" below.

# General Tab (Options Dialog Box)

This tab enables you to configure general Analysis options, such as date formats, temporary storage location, and transaction report settings.




| | |
|---|---|
| **To access** | **Tools > Options > General** tab. |
| **See Also** | "How to Configure Settings for Analyzing Load Test Results" on the previous page |

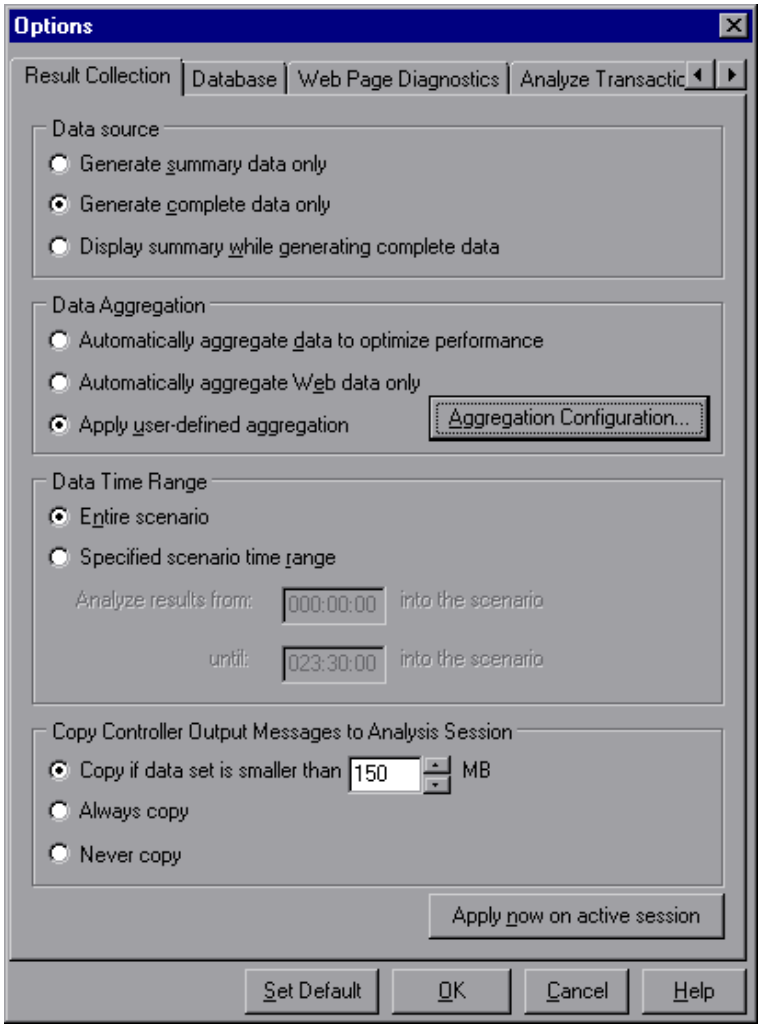
User interface elements are described below:

| UI Element | Description |
|---|---|
| **Date Format** | Select a date format for storage and display. (For example, the date displayed in the Summary report)<br>• **European.** Displays the European date format.<br>• **US.** Displays the U.S. date format.<br>• **Traditional Chinese.** Displays the Traditional Chinese date format.<br>• **Local Regional Options.** Displays the date format as defined in the current user's regional settings.<br>**Note:** When you change the date format, it only affects newly created Analysis sessions. The date format of existing sessions is not affected. |
| **File Browser** | Select the directory location at which you want the file browser to open.<br>• **Open at most recently used directory.** Opens the file browser at the previously used directory location.<br>• **Open at specified directory.** Opens the file browser at a specified directory.<br>In the **Directory path** box, enter the directory location where you want the file browser to open. |
| **Temporary Storage Location** | Select the directory location in which you want to save temporary files.<br>• **Use Windows temporary directory.** Saves temporary files in your Windows temp directory.<br>• **Use a specified directory.** Saves temporary files in a specified directory.<br>In the **Directory path** box, enter the directory location in which you want to save temporary files. |
| **Summary Report** | Set the following transaction settings in the Summary Report:<br>• **Transaction Percentile.** The Summary Report contains a percentile column showing the response time of 90% of transactions (90% of transactions that fall within this amount of time). To change the value of the default 90 percentile, enter a new figure in the **Transaction Percentile** box.<br>The Transaction Percentile value is only applied to newly created templates . To create a new template, select **Tools >Templates**. For details, see "Apply/Edit Template Dialog Box" on page 1323. |

| UI Element | Description |
|---|---|
| **Graph** | Select the way in which graphs shows the Elapsed Scenario Time on the x-axis.<br><br>**Use Absolute time by default.** Shows an elapsed time based on the absolute time of the machine's system clock. If not checked, the graphs show the elapsed time relative to the start of the scenario. The default is unchecked. |
| **Analyze Result** | **Use cached file to store data.** Uses a cached file to store the analysis data.<br><br>This option should only be used when analyzing a large result file. Enabling this option may increase the time required to analyze and open the results. |

## Result Collection Tab (Options Dialog Box)

This tab enables you to configure how Analysis processes result data from load test scenarios.



| | |
|---|---|
| **To access** | **Tools > Options > Result Collection** tab. |
| **Important information** | The options in this tab are pre-defined with default settings. It is recommended to use these default settings unless there is a specific need to change them. Changing some of the settings, such as default aggregation, can significantly impact the amount of data stored in the Analysis database. |
| **See Also** | "How to Configure Settings for Analyzing Load Test Results" on page 1279 |

User interface elements are described below:

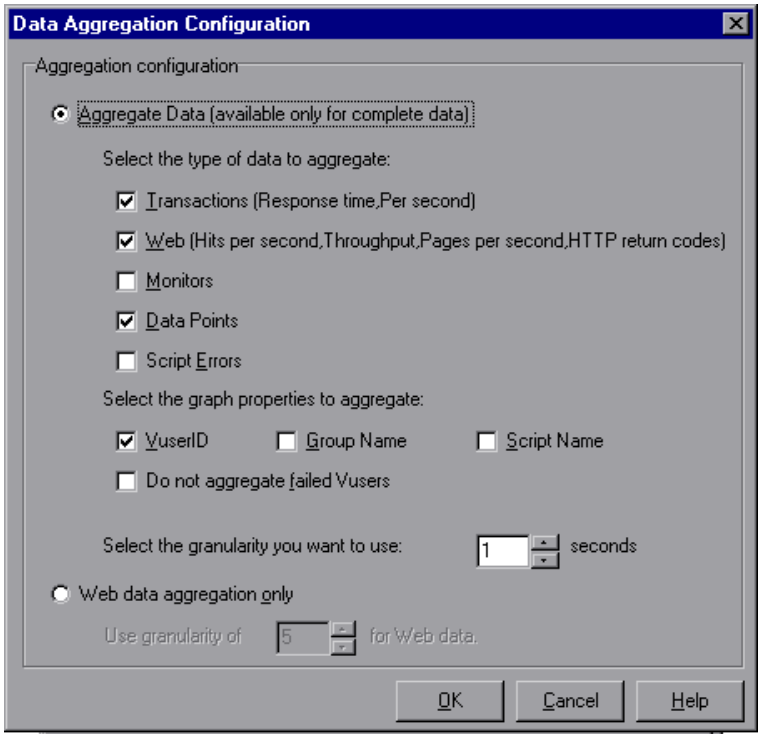
| UI Element | Description |
|---|---|
| **Data Source** | In this area, you configure how Analysis generates result data from load test scenarios.<br><br>**Complete data** refers to the result data after it has been processed for use within Analysis. **Summary data** refers to the raw, unprocessed data. The summary graphs contain general information such as transaction names and times. For more details on summary data versus complete data, see "Summary Data Versus Complete Data" on page 1277.<br><br>Select one of the following options:<br><br>• **Generate summary data only.** If this option is selected, Analysis will not process the data for advanced use with filtering and grouping.<br>• **Generate complete data only.** If this option is selected, the graphs can then be sorted, filtered, and manipulated.<br>• **Display summary data while generating complete data.** Enables you to view summary data while you wait for the complete data to be processed.<br><br>**Note:** If you selected one of the options to generate complete data, you can define how Analysis aggregates the complete data in the **Data Aggregation** area. |

| UI Element | Description |
|---|---|
| **Data Aggregation** | If you chose to generate complete data in the **Data Source** area, you use this area to configure how Analysis aggregates the data.<br><br>Data aggregation is necessary in order to reduce the size of the database and decrease processing time in large scenarios.<br><br>Select one of the following options:<br><br>• **Automatically aggregate data to optimize performance.** Aggregates data using built-in data aggregation formulas.<br>• **Automatically aggregate Web data only.** Aggregates Web data only using built-in data aggregation formulas.<br>• **Apply user-defined aggregation.** Aggregates data using settings you define.<br><br>Click the **Aggregation Configuration** button to open the Data Aggregation Configuration Dialog Box and define your custom aggregation settings. For details on the user interface, see "Data Aggregation Configuration Dialog Box (Result Collection Tab)" on the next page. |
| **Data Time Range** | In this area you specify whether to display data for the complete duration of the scenario, or for a specified time range only. Select one of the following options:<br><br>• **Entire scenario.** Displays data for the complete duration of the load test scenario<br>• **Specified scenario time range.** Specify the time range using the following boxes:<br>  • **Analyze results from.** Enter the amount of scenario time you want to elapse (in hh:mm:ss format) before Analysis begins displaying data.<br>  • **until.** Enter the point in the scenario (in hh:mm:ss format) at which you want Analysis to stop displaying data.<br><br>**Note:** The **Specified scenario time range** settings are not applied to the Connections and Running Vusers graphs. |

| UI Element | Description |
|---|---|
| **Copy Controller Output Messages to Analysis Session** | Controller output messages are displayed in Analysis in Controller's Output Messages window. Select one of the following options for copying output messages generated by Controller to the Analysis session.<br>• **Copy if data set is smaller than X MB.** Copies the Controller output data to the Analysis session if the data set is smaller than the amount you specify.<br>• **Always Copy.** Always copies the Controller output data to the Analysis session.<br>• **Never Copy.** Never copies the Controller output data to the Analysis session. |
| Apply now on active session | Click this button to apply the settings in the Result Collection tab to the current session. The Controller output data is copied when the Analysis session is saved. |

## Data Aggregation Configuration Dialog Box (Result Collection Tab)

If you choose to generate the complete data from the load lest scenario results, Analysis aggregates the data using either built-in data aggregation formulas, or aggregation settings that you define. This dialog box enables you to define custom aggregation settings.



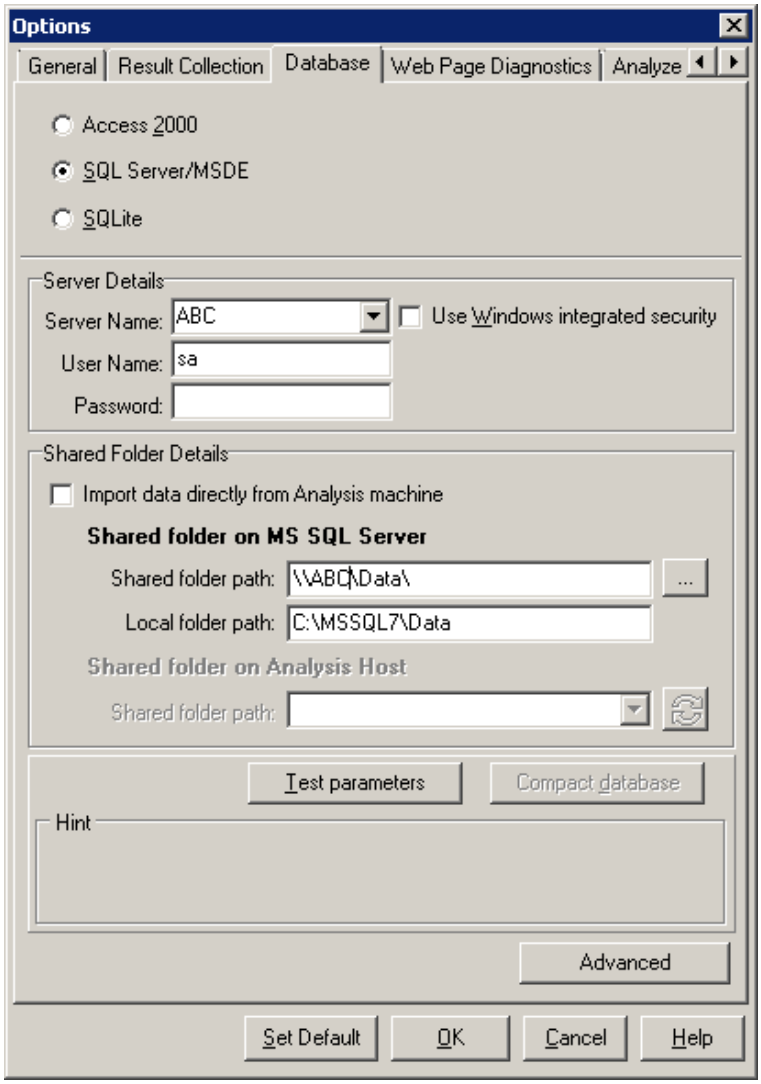
| | |
|---|---|
| **To access** | Select **Tools > Options > Result Collection**. Select the **Apply user-defined aggregation** option and click the **Aggregation Configuration** button. |
| **Important information** | In this dialog box, you can select granularity settings. To reduce the size of the database, increase the granularity. To focus on more detailed results, decrease the granularity. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Aggregate Data** | Select this option to define your custom aggregation settings using the following criteria:<br>• **Select the type of data to aggregate.** Use the check boxes to select the types of graphs for which you want to aggregate data.<br>• **Select graph properties to aggregate**. Use the check boxes to select the graph properties you want to aggregate.<br>To exclude data from failed Vusers, select **Do not aggregate failed Vusers**.<br>**Note:** You will not be able to drill down on the graph properties you select in this list.<br>• **Select the granularity you want to use.** Specify a custom granularity for the data. The minimum granularity is 1 second. |
| **Web data aggregation only** | Select this option to aggregate Web data only. In the **Use Granularity of X for Web data** box, specify a custom granularity for Web data.<br>The minimum granularity is 1 second. By default, Analysis summarizes Web measurements every 5 seconds. |

## Database Tab (Options Dialog Box)

This tab enables you to specify the database in which to store Analysis session result data and to configure the way in which CSV files will be imported into the database.




| | |
|---|---|
| **To access** | **Analysis > Tools > Options > Database** tab. |
| **See also** | "Importing Data Directly from the Analysis Machine" on page 1277 |

## Main Area

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Access 2000** | Instructs LoadRunner to save Analysis result data in an Access 2000 database format. This setting is the default.<br><br>Recommended if the Analysis session file is less than 2 GB. |

| UI Element | Description |
|---|---|
| **SQL Server** | Instructs LoadRunner to save Analysis result data on an SQL server. If you select this option, you have to complete the **Server Details** and **Shared Folder Details**, described below.<br><br>Use this format if you need to work in multithread mode.<br><br>Recommended if the Analysis session file is 2 GB to 10 GB. |
| **SQLite** | Instructs LoadRunner to save Analysis result data in an SQLite database format.<br><br>Allows you to store up to 32 terabytes of data.<br><br>If you choose this format, you will not be able to work in multithread mode.<br><br>Recommended if the Analysis session file is more than 10 GB. |
| **Server Details area** | SQL server details. See description below. |
| **Shared Folder Details area** | SQL server shared folder details. See description below. |
| Test parameters | Depending on which database you are using, this button performs the following action:<br><br>• **For Access.** Checks the connection parameters to the Access database and verifies that the **delimiter** on your machine's regional settings matches the Microsoft JET delimiter on the database machine.<br>• **For SQL server.** Checks the connection parameters, the existence of a shared server directory, whether there are write permissions on the shared server directory, and whether the shared and physical server directories are synchronized.<br>• **For SQLite.** This button is disabled. |
| Compact database | When you configure and set up your Analysis session, the database containing the results may become fragmented. As a result, it will use excessive disk space. For Access databases, the **Compact database** button enables you to repair and compress your results and optimize your database. This button is disabled if you choose **SQLite**.<br><br>**Note:** Long load test scenarios (duration of two hours or more) will require more time for compacting. |
| Advanced | Opens the Advanced Options dialog box, allowing you to increase performance when processing LoadRunner results or importing data from other sources.This button is disabled if you choose **SQLite**. For user interface details see "Advanced Options Dialog Box (Database Tab)" on page 1290. |

**Note:** Both the Access 2000 database format and the SQLite format are embedded databases. The session directory contains both the database and the analysis data.

## Server Details Area

If you choose to store Analysis result data on an SQL server, you need to complete the server details. User interface elements are described below:

| UI Element | Description |
|---|---|
| **Server Name** | The name of the machine on which the SQL server is running. |
| **Use Windows integrated security** | Enables you to use your Windows login, instead of specifying a user name and password. By default, the user name "sa" and no password are used for the SQL server. |
| **User Name** | The user name for the master database. |
| **Password** | The password for the master database. |

## Shared Folder Details Area

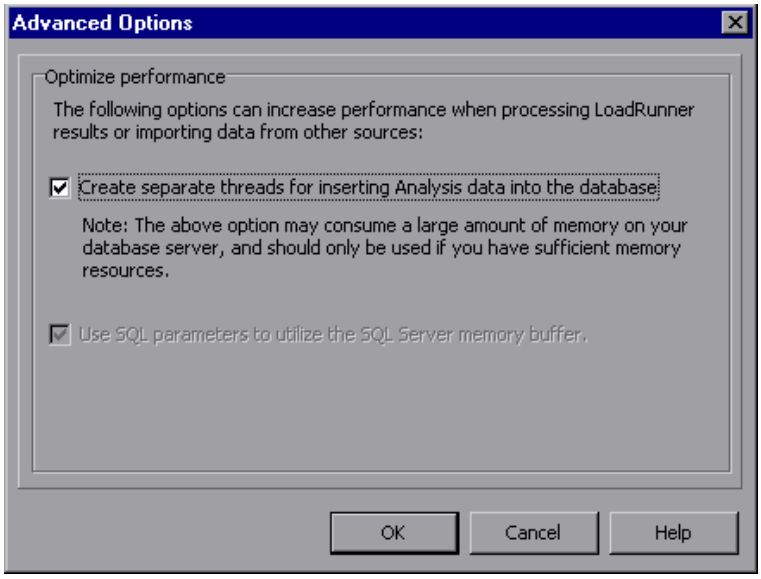
If you store Analysis result data on an SQL server, you need to provide the shared folder details. User interface elements are described below:

| UI Element | Description |
|---|---|
| **Import Data Directly from Analysis machine** | Select this option to import data directly from the Analysis machine. For details on this option, see "Importing Data Directly from the Analysis Machine" on page 1277. |

| UI Element | Description |
|---|---|
| **Shared Folder on MS SQL Server** | • **Shared folder path.** Enter a shared folder on the SQL server. For example, if your SQL server's name is `fly`, enter `\\fly\<Analysis database folder>\`.<br>This folder has different functions, depending on how you import the Analysis data:<br>  • **If you did not select the option to import data directly from the Analysis machine**, this folder stores permanent and temporary database files. Analysis results stored on an SQL server can only be viewed on the machine's local LAN.<br>  • **If you selected the option to import data directly from the Analysis machine**, this folder is used to store an empty database template copied from the Analysis machine.<br>• **Local folder path.** Enter the real drive and folder path on the SQL server that correspond to the above shared folder path. For example, if the Analysis database is mapped to an SQL server named `fly`, and `fly` is mapped to drive D, enter `D:\<Analysis database folder>`.<br>If the SQL server and Analysis are on the same machine, the logical storage location and physical storage location are identical. |
| **Shared Folder on Analysis Host** | If you selected the option to import data directly from the Analysis machine, the **Shared folder path** box is enabled. Analysis detects all shared folders on your Analysis machine and displays them in a drop-down list. Select a shared folder from the list.<br>**Notes:**<br>• Ensure that the user running the SQL server (by default, SYSTEM) has access rights to this shared folder.<br>• If you add a new shared folder on your machine, you can click the refresh button to display the updated list of shared folders.<br>• Analysis creates the CSV files in this folder and the SQL server imports these CSV files from the Analysis machine directly into the database. This folder stores permanent and temporary database files. |

## Advanced Options Dialog Box (Database Tab)

This dialog box enables you to increase performance when processing LoadRunner results or importing data from other sources.



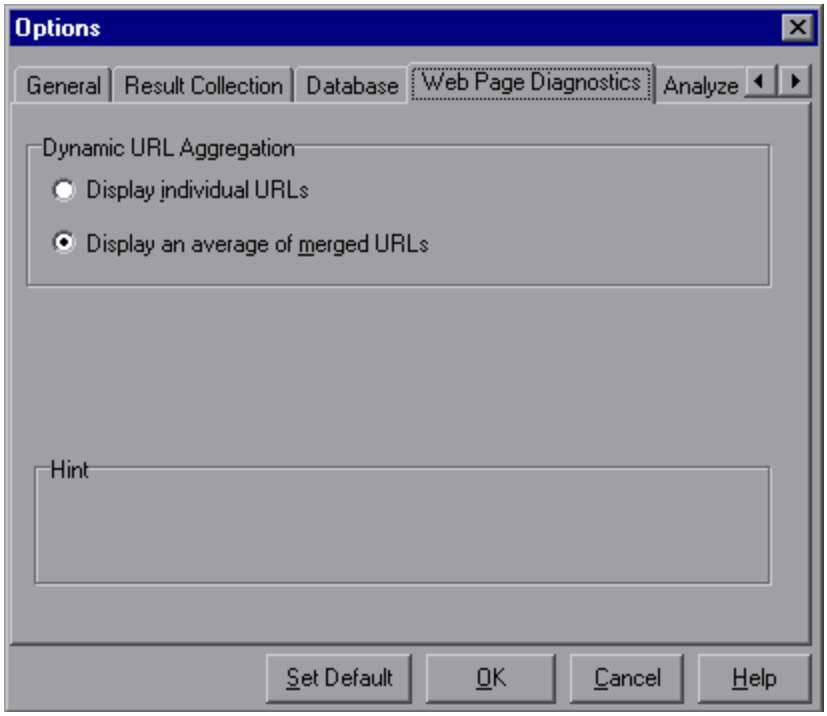

| | |
|---|---|
| **To access** | **Analysis > Tools > Options > Database** tab **> Advanced** button |
| **See also** | "Database Tab (Options Dialog Box)" on page 1286 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Create separate threads for inserting Analysis data into the database.** | This option may consume a large amount of memory on your database server, and should only be used if you have sufficient memory resources. |
| **Use SQL parameters to utilize the SQL Server memory buffer.** | This option is only enabled when you store Analysis result data on an SQL server. |

## Web Page Diagnostics Tab (Options Dialog Box)

This tab enables you to set Web page breakdown options. You can choose how to aggregate the display of URLs that include dynamic information, such as a session ID. You can display these URLs individually, or you can unify them and display them as one line with merged data points.




| | |
|---|---|
| **To access** | **Tools > Options > Web Page Diagnostics** tab |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Display individual URLs** | Displays each URL individually |
| **Display an average of merged URLs** | Merges URLs from the same script step into one URL, and displays it with merged (average) data points. |

## Session Information Dialog Box (Options Dialog Box)

This dialog box enables you to view a summary of the configuration properties of the current Analysis session.

| To access | File > Session Information |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| Aggregation Properties... | Displays the type of data aggregated, the criteria according to which it is aggregated, and the time granularity of the aggregated data. |
| Server Properties... | Displays the properties of the SQL server. |
| **Aggregation** | Indicates whether the session data has been aggregated. |
| **Data Collection Mode** | Indicates whether the session displays complete data or summary data. |
| **Data Time Filter** | Indicates whether a time filter has been applied to the session. |
| **Database Name** | Displays the name and directory path of the database. |
| **Database Type** | Displays the type of database used to store the load test scenario data. |
| **Results** | Displays the name of the LoadRunner result file. |

| UI Element | Description |
|---|---|
| **Session Name** | Displays the name of the current session. |
| **Web Granularity** | Displays the Web granularity used in the session. |

# Viewing Load Test Scenario Information

## Viewing Load Test Scenario Information

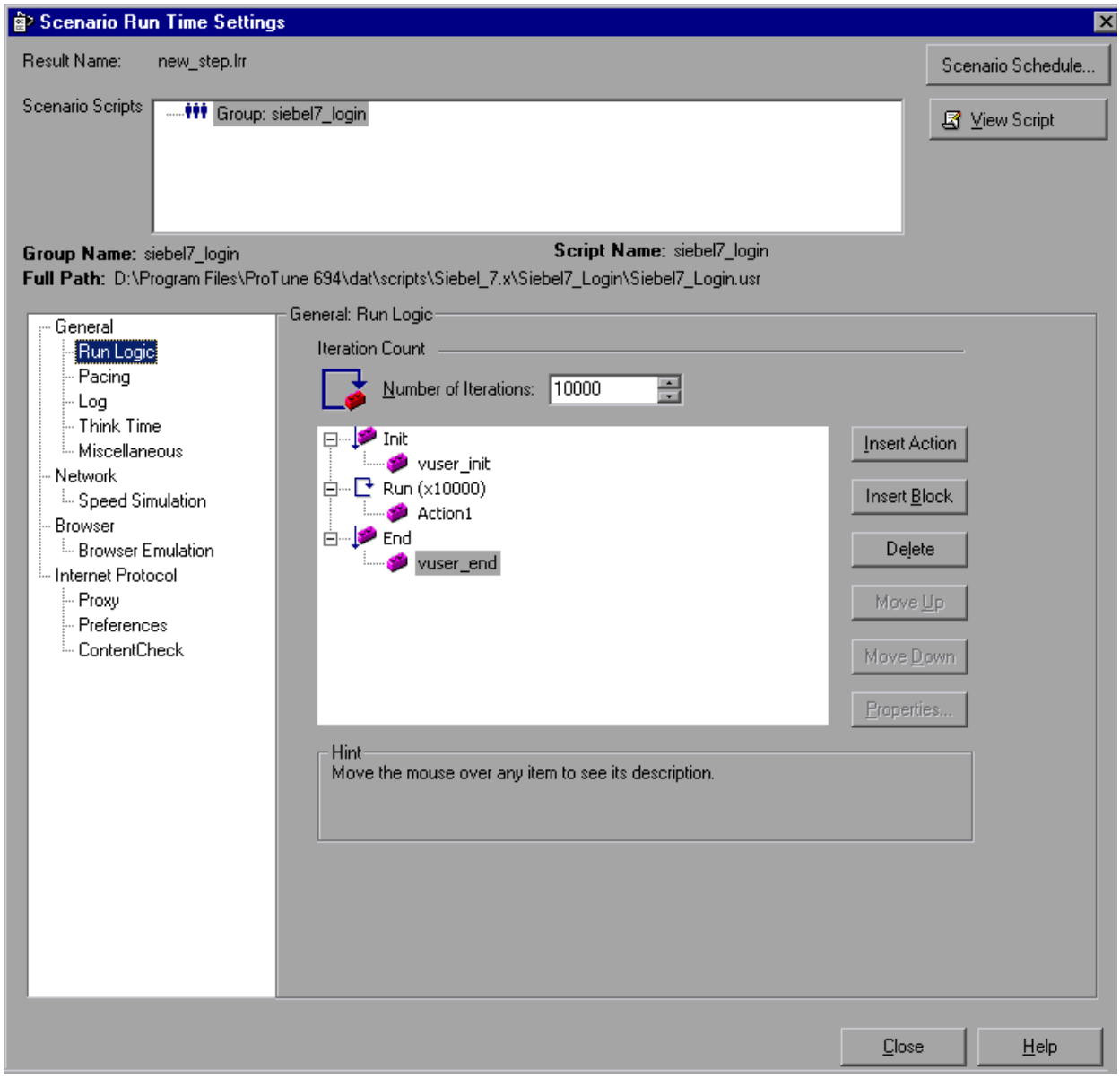
In Analysis, you can view information about the load test scenario which you are analyzing. You can view the scenario runtime settings and output messages that were generated by Controller during the scenario.

You can view information about the Vuser groups and scripts that were run in each scenario, as well as the runtime settings for each script in a scenario, in the Scenario runtime settings dialog box.

> **Note:** The runtime settings allow you to customize the way a Vuser script is executed. You configure the runtime settings from Controller or Virtual User Generator (*VuGen*) before running a scenario. For information on configuring the runtime settings, refer to the online help in those products.

Select **File > View Scenario Runtime Settings**, or click the **View runtime settings** button on the toolbar.

The Scenario runtime settings dialog box opens, displaying the Vuser groups, scripts, and scheduling information for each scenario. For each script in a scenario, you can view the runtime settings that were configured in Controller or VuGen before scenario execution.




## Configuring Controller Output Messages Settings

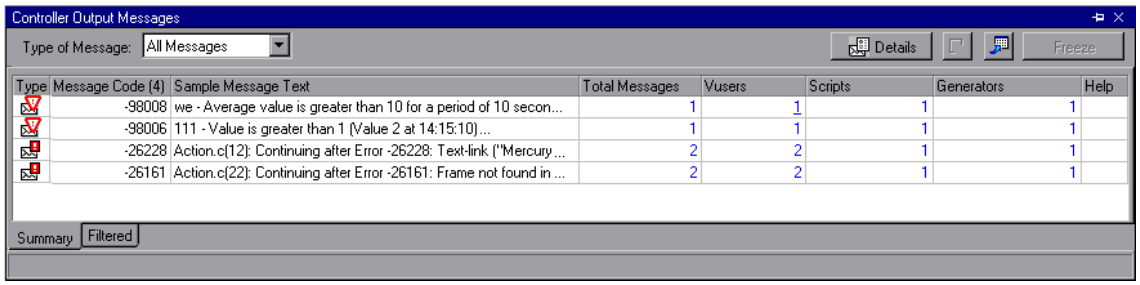
This task describes how to configure settings for output messages.

1. Choose **Tools > Options** and select the **Result Collection** tab.
2. In the **Copy Controller Output Messages to Analysis Session** area, choose one of the following options:
   - **Copy if data set is smaller than X MB.** Copies Controller output data to the Analysis session if the data set is smaller than the amount you specify.
   - **Always Copy.** Always copies Controller output data to the Analysis session.
   - **Never Copy.** Never copies Controller output data to the Analysis session.

3. Apply your settings.
   - To apply these settings to the current session, click **Apply now to active session**.
   - To apply these settings after the current session is saved, click **OK**.

## Controller Output Messages Window

This window displays error, notification, warning, debug, and batch messages that are sent to Controller by the Vusers and load generators during a scenario run.




| | |
|---|---|
| **To access** | **Windows > Controller Output Messages** |
| **Important information** | • The Summary tab is displayed by default when you open this window.<br>• Analysis searches for the output data in the current Analysis session. If the data is not found, it searches in the scenario results folder. If Analysis cannot locate the results folder, no messages are displayed. |

### Summary Tab

This tab displays summary information about the messages sent during a scenario run.

| UI Element | Description |
|---|---|
| Details | Displays the full text of the selected output message in the Detailed Message Text area at the bottom of the Output window. |
| | **Remove all messages.** Clears all log information from the Output window. |
| | **Export the view.** Saves the output to a specified file. |
| Freeze<br>Resume | • **Freeze.** Stops updating the Output window with messages.<br>• **Resume.** Resumes updating the Output window with messages. The newly updated log information is displayed in a red frame. |

| UI Element | Description |
|---|---|
| **Detailed Message Text** | Displays the full text of the selected output message when you click the **Details** button. |
| **Generators** | Displays the number of load generators that generated messages with the specified message code. |
| **Help** | Displays an icon if there is a link to troubleshooting for the message. |
| **Message Code** | Displays the code assigned to all similar messages. The number in parentheses indicates the number of different codes displayed in the Output window. |
| **Sample Message Text** | Displays an example of the text of a message with the specified code. |
| **Scripts** | Displays the number of scripts whose execution generated messages with the specified code. |
| **Total Messages** | Displays the total number of sent messages with the specified code. |
| **Type** | The type of message being displayed. The following icons indicate the various message types.<br>• Batch<br>• Debug<br>• Errors<br>• Notifications<br>• Warnings<br>• Alerts<br>For more information about each type, see "Type of Message" on the next page. |

| UI Element | Description |
|---|---|
| **Type of Message** | Filters the output messages to display only certain message types. Select one of the following filters:<br>• **All messages.** Displays all message types.<br>• **Batch.** Sent instead of message boxes appearing in Controller, if you are using automation.<br>• **Debug.** Sent only if the debugging feature is enabled in Controller. (Expert mode: **Tools > Options > Debug Information**). For more information, see "Options > Debug Information Tab" on page 242.<br>• **Errors.** Usually indicate that the script failed.<br>• **Notifications.** Provides runtime information, such as message sent using **lr_output_message**.<br>• **Warnings.** Indicates that the Vuser encountered a problem, but the scenario continued to run.<br>• **Alerts**. Indicates a warning. |
| **Vusers** | Displays the number of Vusers that generated messages with the specified code. |

## Filtered Tab

This tab displays a drilled down view by message, Vuser, script, or load generator. For example, if you drill down on the Vuser column, the Filtered tab displays all the messages with the code you selected, grouped by the Vusers that sent the messages.

| UI Element | Description |
|---|---|
| | **Previous/Next View.** Enables you to move between the various drill down levels. |
| | Displays the full text of the selected output message in the Detailed Message Text area at the bottom of the Output window. |
| | **Export the view.** Saves the output to a specified file. |
| | Refreshes the Filtered tab with new log information that arrived in the Output window updated in the Summary tab. |
| **<Message icon>** | Displays an icon indicating the type of message by which the current Output view is filtered. |
| **Active Filter** | Displays the category or categories by which the current Output view is filtered. |

| UI Element | Description |
|---|---|
| **Viewed By** | Displays the name of the column on which you selected to drill down. The following icons indicate the various message types:<br>• Batch<br>• Debug<br>• Errors<br>• Notifications<br>• Warnings<br>• Alerts |
| **Detailed Message Text** | Displays the full text of the selected output message when the **Details** button is selected. |
| **Message** | Displays all instances of the sample message text. |
| **Script** | The script on which the message was generated. If you click the blue link, VuGen opens displaying the script. |
| **Action** | The action in the script where the message was generated. If you click the blue link, VuGen opens the script to the relevant action. |
| **Line #** | The line in the script where the message was generated. If you click the blue link, VuGen opens the script and highlights the relevant line. |
| **# Lines** | The total number of lines in the script where the Vuser failed. |
| **Time** | The time the message was generated. |
| **Iteration** | The iteration during which the message was generated. |
| **Vuser** | The Vuser that generated the message. |
| **Generator** | The load generator on which the message was generated. If you click the blue link, the Load Generator dialog box opens. |
| **# Messages** | The number of messages generated by a specific Vuser. |

## Scenario Runtime Settings Dialog Box

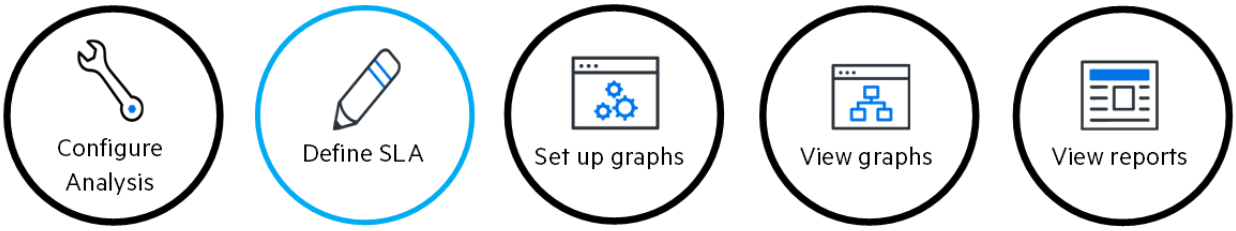
This dialog box enables you to view information about executed load test scenarios, as well as the runtime settings for each script in a scenario.

| | |
|---|---|
| **To access** | Toolbar > |
| **See also** | "Viewing Load Test Scenario Information" on page 1294 |

User interface elements are described below

| UI Element | Description |
|---|---|
| **Result Name** | The name of the result file. |
| **Scenario Scripts** | Displays the result set for each executed scenario, as well as the Vuser groups and scripts that were run in the scenario. |
| **Group Name** | Displays the name of the group to which the selected script belongs. |
| **Full Path** | Displays the script's full directory path. |
| **Script Name** | Displays the name of the selected script. |
| **Scenario Schedule** | Displays goal-oriented or manual scenario scheduling information for the selected scenario. |
| **View Script** | Opens the Virtual User Generator, so that you can edit the script. |

# Defining Service Level Agreements




## Service Level Agreements Overview

**Service level agreements (SLAs)** are specific goals that you define for your load test scenario. After a scenario run, LoadRunner Analysis compares these goals against performance related data that was gathered and stored during the course of the run, and determines whether the SLA passed or failed.

Depending on the measurements that you are evaluating for your goal, LoadRunner determines the SLA status in one of the following ways:

| SLA Type | Description |
|---|---|
| **SLA status determined at time intervals over a timeline** | Analysis displays SLA statuses at set time intervals over a timeline within the run. At each time interval in the timeline—for example, every 10 seconds—Analysis checks to see if the measurement's performance deviated from the threshold defined in the SLA.<br>Measurements that can be evaluated in this way:<br>• **Average Transaction Response Time**<br>• **Errors per Second** |
| **SLA status determined over the whole run** | Analysis displays a single SLA status for the whole scenario run.<br>Measurements that can be evaluated in this way:<br>• **Total Hits per run**<br>• **Average Hits (hits/second) per run**<br>• **Total Throughput (bytes) per run**<br>• **Average Throughput (bytes/second) per run** |

## Tracking Period

When you define service level agreements (SLAs)an SLA for measurements that are evaluated over a timeline, Analysis determines SLA statuses at specified time intervals within that timeline. The frequency of the time intervals is called the **tracking period**.

An internally-calculated tracking period is defined by default. You can change the tracking period by entering a value in the Advanced Options dialog box which Analysis plugs into a built-in algorithm to calculate the tracking period. For details, see "Advanced Options Dialog Box (Service Level Agreement Pane)" on page 1305.

## Define Service Level Agreements

This task describes how to define service level agreements (SLAs).

You can define service level agreements (SLAs) which measure scenario goals over time intervals, or over a whole scenario run. For details, see "Service Level Agreements Overview" on the previous page.

> **Tip:** For a use-case scenario related to this task, see "Define Service Level Agreements - Use-Case Scenario" on the next page.

1. Prerequisites

   If you are defining an SLA for Average Transaction Response Time, your scenario must include a script that contains at least one transaction.

2. Run through the SLA wizard

In the Service Level Agreement pane, click **New** to open the Service Level Agreement wizard. For user interface details, see "Service Level Agreement Wizard" on page 1307.

a. Select a measurement for the SLA.

b. If you are defining an SLA for Average Transaction Response Time or Transaction Response Time (Percentile), select the transactions to include in your goal.

c. (Optional) When evaluating SLA statuses over a timeline, select load criteria to take into account and define appropriate load value ranges for the load criteria. For an example, see "Define Service Level Agreements - Use-Case Scenario" below.

d. Set thresholds for the measurements.

- If the **Average Transaction Response Time** or **Errors per Second** exceed the defined thresholds, Analysis will produce a **Failed** SLA status.
- If **Transaction Response Time(Percentile)**, **Total Hits per run**, **Average Hits (hits/second) per run**, **Total Throughput (bytes) per run**, or **Average Throughput (bytes/second) per run** are lower than the defined threshold, Analysis will produce a **Failed** SLA status.

3. Define a tracking period - optional

   For measurements whose SLA statuses are determined over time intervals, you need to define the frequency of the time intervals, that is, the **tracking period**. For details, see "Tracking Period" on the previous page.

   For user interface details, see "Advanced Options Dialog Box (Service Level Agreement Pane)" on page 1305.

4. Results

   When analyzing your scenario run, LoadRunner Analysis compares the data collected from the scenario run against the SLA settings, and determines SLA statuses which are included in the default Summary Report.

## Define Service Level Agreements - Use-Case Scenario

This use-case scenario describes how to define a service level agreement (SLA) for Average Transaction Response Time.

1. Background

   The administrator of Web Tours would like to know when the average transaction response time for booking a flight and searching for a flight exceeds a certain value. Assume that your scenario includes a script that includes the following transactions: **book_flight** and **search_flight**.

2. Start the SLA wizard

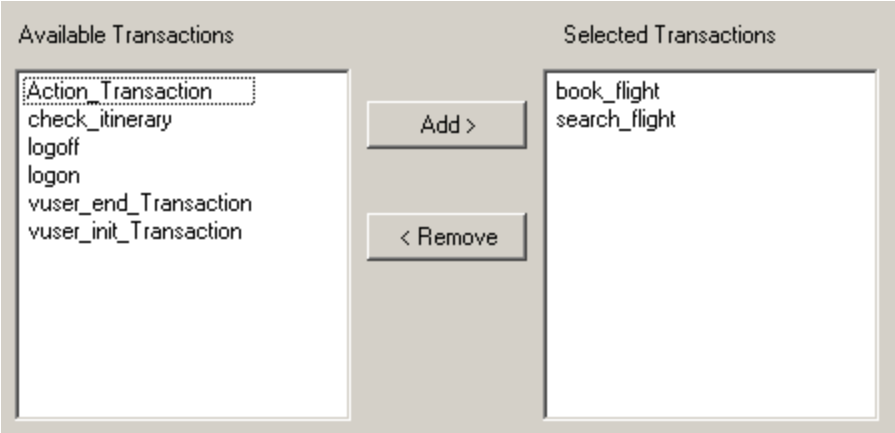
   In the Service Level Agreement pane, click **New** to open the Service Level Agreement wizard.

3. Select the measurement for the SLA

   On the Select a Measurement page, under **Select a Measurement for Your Goal,** in the **Transaction Response Time** box, select **Average**.

4. Select the transactions to evaluate in your goal

On the Select a Transaction page, select the transactions to be evaluated: **book_flight** and **search_flight**.




5. Select a load criterion and define appropriate ranges of load - optional

   On the Select Load Criteria page, select the load criterion to take into account when evaluating the average transaction response time.

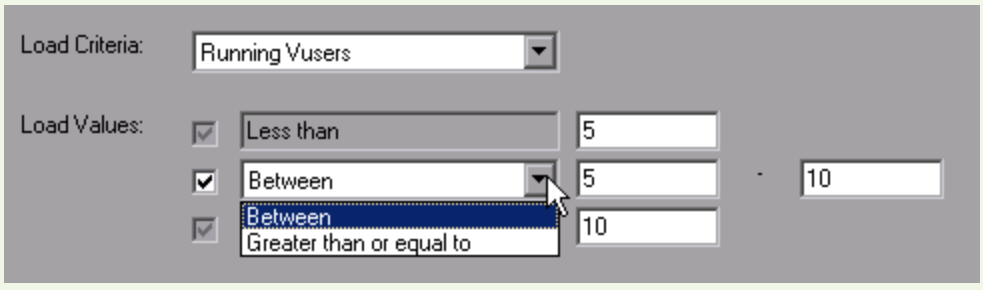
   In this case, to see the effect that various quantities of Vusers running on the system has on the average transaction response time of each transaction, in the **Load Criteria** box, select **Running Vusers**.

   Then set the value ranges for the running Vusers:

   Consider less than 20 Vusers to be a light load, 20 – 50 Vusers an average load, and 50 Vusers or more a heavy load. Enter these values in the Load Values boxes.

   **Note:**

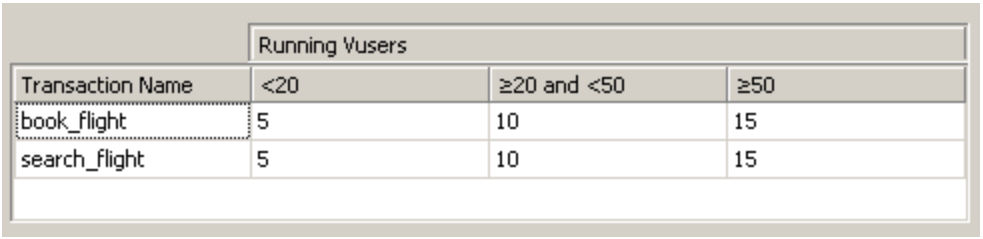
   - You can set up to three in-between ranges.
   - Valid load value ranges are consecutive—there are no gaps in the range—and span all values from zero to infinity.

6. Set thresholds

   On the Set Threshold Values page, you define the acceptable average transaction response times for the transactions, taking into account the defined load criteria.

In this case, define the same threshold values for both transactions as follows: for a light load, a reasonable average response time can be up to 5 seconds, for an average load, up to 10 seconds, and for a heavy load, up to 15 seconds.



| | Running Vusers | | |
|---|---|---|---|
| Transaction Name | <20 | ≥20 and <50 | ≥50 |
| book_flight | 5 | 10 | 15 |
| search_flight | 5 | 10 | 15 |

> **Tip:** To define the same thresholds for all the transactions, you can type the values in the table nearer the bottom of the Set Threshold Values page, and click **Apply to all transactions**.

7. Define a tracking period - optional

   When SLA statuses for a measurement are determined at time intervals over a timeline, the frequency of the time intervals is determined by the **tracking period**.

   This step is optional because an internally-calculated tracking period of at least 5 seconds is defined by default. You can change the tracking period in the Advanced Options dialog box:

   a. In the Service Level Agreement pane, click the **Advanced** button.

   b. Select **Tracking period of at least X seconds**, and select a tracking period. The time intervals are calculated by Analysis according to a built-in algorithm and as a function of the value you enter here.

      **Example:**

      If you select a tracking period of 10, and the aggregation granularity for the scenario (defined in Analysis) is 6, then the tracking period is set to the nearest multiple of 6 that is greater than or equal to 10, that is, Tracking Period = 12.

      For details, see "Tracking Period" on page 1301.

      For user interface details, see "Advanced Options Dialog Box (Service Level Agreement Pane)" on the next page.

8. Results

   When analyzing your scenario run, Analysis applies your SLA settings to the default Summary Report and the report is updated to include all the relevant SLA information.

   For example, it displays the worst performing transactions in terms of defined SLAs, how specific transactions performed over set time intervals, and overall SLA statuses.

## Service Level Agreement Pane

This pane lists all the service level agreements (SLAs) defined for the scenario.

| | |
|---|---|
| **To access** | **In Analysis: Tools** menu > **Configure SLA Rules** > **Service Level Agreement** pane<br>**In Controller: Design** tab |
| **Relevant Tasks** | • "How to Design a Goal-Oriented Scenario" on page 951<br>• "How to Design a Manual Scenario" on page 953<br>• "Define Service Level Agreements" on page 1301<br>• "Define Service Level Agreements - Use-Case Scenario" on page 1302 |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Starts the Service Level Agreement wizard where you can define new goals for the load test scenario. |
| | Opens the Goal Details dialog box which displays a summary of the details of the selected SLA. |
| | Opens the Service Level Agreement wizard where you can modify the goals defined in the SLA. |
| | Deletes the selected SLA. |
| | Opens the Advanced Options dialog box where you can adjust the tracking period for measurements that are evaluated per time interval over a timeline.<br>For more information, see "Tracking Period" on page 1301.<br>For user interface details, see "Advanced Options Dialog Box (Service Level Agreement Pane)" below. |
| **Service Level Agreement list** | Lists the SLAs defined for the scenario. |

## Advanced Options Dialog Box (Service Level Agreement Pane)

This dialog box enables you to define a tracking period for load test scenario.

| | |
|---|---|
| **To access** | **In Analysis: Tools** menu > **Configure SLA Rules** > **Service Level Agreement** pane > <br>**In Controller: Design** tab > **Service Level Agreement** pane > |

| | |
|---|---|
| **Important information** | The tracking period is calculated by Analysis according to a built-in algorithm and as a function of the value entered here. |
| **Relevant tasks** | • "Define Service Level Agreements" on page 1301<br>• "Define Service Level Agreements - Use-Case Scenario" on page 1302 |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Internally calculated tracking period** | Analysis sets the tracking period to the minimum value possible, taking into account the aggregation granularity defined for the scenario. This value is at least 5 seconds. It uses the following formula:<br>`Tracking Period = Max (5 seconds, aggregation granularity)` |
| **Tracking period of at least X seconds** | Determines the minimum amount of time for the tracking period. This value can never be less than 5 seconds.<br>Analysis sets the tracking period to the nearest multiple of the scenario's aggregation granularity that is greater than or equal to the value (X) that you selected.<br>For this option, Analysis uses the following formula:<br>`Tracking Period =`<br>`Max(5 seconds, m(Aggregation Granularity))`<br>where **m** is a multiple of the scenario's aggregation granularity such that **m (Aggregation Granularity)** is greater than or equal to **X.**<br>**Example:** If you select a tracking period of X=10, and the aggregation granularity for the scenario is 6, then the tracking period is set to the nearest multiple of 6 that is greater than or equal to 10, that is, Tracking Period = 12. |

## Goal Details Dialog Box (Service Level Agreement Pane)

This dialog box displays the thresholds that were set for the selected SLA.

| | |
|---|---|
| **To access** | **In Analysis: Tools** menu > **Configure SLA Rules** > **Service Level Agreement** pane > Details<br>**In Controller: Design** tab > **Service Level Agreement** pane > Details |
| **Important information** | If you defined load criteria as part of your SLA, the threshold values are displayed per the defined load value ranges. |

| See also | "Service Level Agreements Overview" on page 1300 |
|---|---|

## Service Level Agreement Wizard

This wizard enables you to define goals or **service level agreements** (SLAs) for your load test scenario.

| To access | **In Analysis: Tools** menu > **Configure SLA Rules** > **Service Level Agreement** pane > <br><br>**In Controller: Design** tab > **Service Level Agreement** pane > |
|---|---|
| **Important information** | There are two modes for the Service Level Agreement wizard. The pages included in the wizard depend on the measurement that is selected. See the wizard maps below. |
| **Relevant tasks** | • "Define Service Level Agreements" on page 1301<br>• "Define Service Level Agreements - Use-Case Scenario" on page 1302 |
| **Wizard map - Goal measured per time interval** | The Service Level Agreement Wizard contains:<br><br>**Welcome** > "Select a Measurement Page" below > ("Select Transactions Page" on the next page) > "Set Load Criteria Page" on page 1309 > "Set Threshold Values Page (Goal Per Time Interval)" on page 1311 |
| **Wizard map - Goal measured over whole scenario run** | The Service Level Agreement Wizard contains:<br><br>**Welcome** > "Select a Measurement Page" below > ("Select Transactions Page" on the next page) > "Set Threshold Values Page (Goal Per Whole Run)" on page 1312 |
| **See also** | "Service Level Agreements Overview" on page 1300 |

### Select a Measurement Page

This wizard page enables you to select a measurement for your goal.

| Important information | • General information about this wizard is available here: "Service Level Agreement Wizard" above.<br>• There are two modes for the Service Level Agreement wizard. The wizard pages that follow depend on the measurement that you select on this page. See the wizard maps below. |
|---|---|

| | |
|---|---|
| **Wizard map - Goal measured per time interval** | The "Service Level Agreement Wizard" on the previous page contains:<br>**Welcome** > Select a Measurement Page > ("Select Transactions Page" below) > "Set Load Criteria Page" on the next page > "Set Threshold Values Page (Goal Per Time Interval)" on page 1311 |
| **Wizard map - Goal measured over whole scenario run** | The "Service Level Agreement Wizard" on the previous page contains:<br>**Welcome** > Select a Measurement Page > ("Select Transactions Page" below) > "Set Threshold Values Page (Goal Per Whole Run)" on page 1312 |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **SLA status determined over the whole run** | Evaluates a single SLA status for the whole scenario run. Select one of the following measurements:<br>• **Transaction Response Time (Percentile)**<br>• **Total Hits per run**<br>• **Average Hits (hits/second) per run**<br>• **Total Throughput (bytes) per run**<br>• **Average Throughput (bytes/second) per run** |
| **SLA status determined per time intervals over a timeline** | Evaluates SLA statuses at set time intervals within the run. Select one of the following measurements:<br>• **Average Transaction Response Time**<br>• **Errors per Second**<br>The time intervals at which the SLA statuses are evaluated are known as the **tracking period**. For details, see "Tracking Period" on page 1301. |

## Select Transactions Page

This wizard page enables you to select transactions to evaluate as part of your goal.

| | |
|---|---|
| **Important information** | • General information about this wizard is available here: "Service Level Agreement Wizard" on page 1307.<br>• This page is displayed when creating an SLA for Transaction Response Time by Average or by Percentile.<br>• In order to define an SLA for Transaction Response Time by Average or by Percentile, at least one of the Vuser scripts participating in the scenario must include a transaction.<br>• You can select multiple transactions using the CTRL key. |
| **Wizard map - Goal measured per time interval** | The "Service Level Agreement Wizard" on page 1307 contains:<br>**Welcome** > "Select a Measurement Page" on page 1307 > (Select Transactions Page) > "Set Load Criteria Page" below > "Set Threshold Values Page (Goal Per Time Interval)" on page 1311 |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Available Transactions** | Lists the transactions in the Vuser scripts participating in the scenario.<br>To move a script to the **Selected Transaction** list, select it and click **Add**. |
| **Selected Transactions** | Lists the transactions in the Vuser scripts participating in the scenario that have been selected for the SLA.<br>To remove a script from this list, select it and click **Remove**. |

## Set Load Criteria Page

This wizard page enables you to select load criteria to take into account when testing your goal.

| | |
|---|---|
| **Important information** | • General information about this wizard is available here: "Service Level Agreement Wizard" on page 1307.<br>• This page is displayed only when defining an SLA that determines SLA statuses per time interval over a timeline**.**<br>• In the next wizard step (Set Threshold Values page), you will set different thresholds per each of the load ranges that you select here. |
| **Wizard map - Goal measured per time interval** | The "Service Level Agreement Wizard" on page 1307 contains:<br>**Welcome** > "Select a Measurement Page" on page 1307 > ("Select Transactions Page" on the previous page) > **Set Load Criteria Page** > "Set Threshold Values Page (Goal Per Time Interval)" on page 1311 |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Load Criteria** | The relevant load criteria that you want to use<br>**Example:** If you want to see the impact of running Vusers on the measurement, select **Running Vusers**.<br>To define an SLA without load criteria, select **None**. |
| **Load Values** | Valid load value ranges are consecutive—there are no gaps in the range—and span all values from zero to infinity.<br>• **Less than.** Enter the upper value for the lower range of values for the load criteria.<br>The lower range is between 0 and the value you entered. It does not include the upper value.<br>**Example:** If you enter 5, the lower range of values for the load criteria is between 0 and 5, but does not include 5.<br>• **Between.** The in-between range of values for the load criteria. Enter lower and upper values for this range. The lower range is included in this range; it does not include the upper value.<br>**Example:** If you enter 5 and 10, the in-between range of values for the load criteria is from 5 and up to, but not including, 10.<br>**Note:** You can set up to three in-between ranges.<br>• **Greater than.** Enter the lower value for the upper range of values for the load criteria.<br>The upper range includes values from the value you entered and on.<br>**Example:** If you enter 10, the upper range of values for the load criteria is from 10 and on. |
| **Selected Measurement** | The measurement selected for the goal. |

## Set Percentile Threshold Values Page

This wizard page enables you to select load criteria to take into account when testing your goal.

| | |
|---|---|
| **Important information** | • General information about this wizard is available here: "Service Level Agreement Wizard" on page 1307.<br>• The Percentile SLA enables you to measure whether the percentage of transaction samples meets the defined threshold criteria.<br>• You can enter a threshold value to 3 decimal places. |

| | |
|---|---|
| **Wizard map - Goal measured over whole scenario run** | The "Service Level Agreement Wizard" on page 1307 contains:<br>**Welcome** > "Select a Measurement Page" on page 1307 > ("Select Transactions Page" on page 1308) > **Set Percentile Threshold Values Page** |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Selected Measurement** | The measurement selected for the goal. |
| **Percentile** | Percentage of transactions to measure against the configured threshold. |
| **Provide threshold value for all transactions** | To apply one set of threshold values to all transactions selected for the goal, enter the threshold value and click **Apply to all**. These values are applied to all the transactions in the Thresholds table at the bottom of the page. |
| **Transaction name** | The transaction from the scenario run. |
| **Threshold** | The threshold value for the selected transaction. |

## Set Threshold Values Page (Goal Per Time Interval)

This wizard page enables you to set thresholds for the measurements you are evaluating in your goal.

| | |
|---|---|
| **Important information** | • General information about this wizard is available here: "Service Level Agreement Wizard" on page 1307.<br>• If you defined load criteria in the "Set Load Criteria Page" on page 1309, you must set thresholds per each of the defined load ranges. If you did not define load criteria, you set one threshold value. For Average Transaction response time, you set threshold values for each transaction.<br>• You can enter a threshold value to 3 decimal places. |
| **Wizard map - Goal measured per time interval** | The "Service Level Agreement Wizard" on page 1307 contains:<br>**Welcome** > "Select a Measurement Page" on page 1307 > ("Select Transactions Page" on page 1308) > "Set Load Criteria Page" on page 1309 > **Set Threshold Values Page (Goal Per Time Interval)** |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Thresholds table>** | The thresholds for your goal. If you defined load criteria, enter thresholds for each range of values.<br><br>**Note:** If the maximum threshold value is exceeded during a particular time interval during the run, Analysis displays an SLA status of **Failed** for that time interval. |
| **Apply to all** (Average Transaction Response Time goal only) | To apply one set of threshold values to all transactions selected for the goal, enter the threshold values in this table and click **Apply to all transactions**. These values are applied to all the transactions in the Thresholds table at the top of the page.<br><br>**Note:** Threshold values for selected transactions do not have to be the same. You can assign different values for each transaction. |
| **Selected Measurement** | The measurement selected for the goal. |

## Set Threshold Values Page (Goal Per Whole Run)

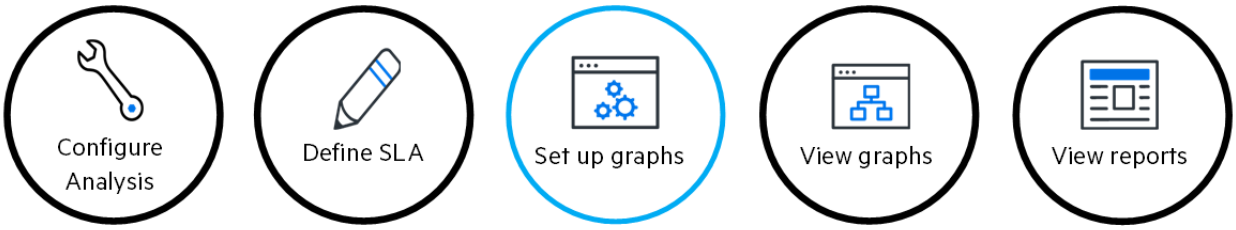
This wizard page enables you to set minimum thresholds for the measurements you are evaluating in your goal.

| | |
|---|---|
| **Important information** | General information about this wizard is available here: "Service Level Agreement Wizard" on page 1307. |
| **Wizard map - Goal measured over whole scenario run** | The "Service Level Agreement Wizard" on page 1307 contains:<br><br>**Welcome** > "Select a Measurement Page" on page 1307 > **Set Threshold Values Page (Goal Per Whole Run)** |
| **See also** | "Service Level Agreements Overview" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Selected measurement** | The measurement selected for the goal. |
| **Threshold** | The minimum threshold value for the selected measurement.<br><br>**Note:** If the value of the measurement is lower than this threshold during the run, Analysis displays an SLA status of **Failed** for the entire run. |

# Set Up Graphs




## Configuring Graph Display

Analysis allows you to customize the display of the graphs and measurements in your session so that you can view the data displayed in the most effective way possible.

### Customize the Analysis Display

You can customize the display of the graphs and measurements in your session so that you can view the data displayed in the most effective way possible.

The following steps describe how to customize the display of analysis:

1. Enlarging a section of the graph

   To zoom in or enlarge a section of the graph, move and hold down the left mouse button over the section of the graph you want to enlarge.
2. Using comments in a graph

   To add a comment to a graph, click and then click the mouse over the section of the graph where you would like to add a comment. Type your comment in the Add Comment dialog box.

   To edit, format or delete a comment from the graph, click the comment and apply your change in the Edit Comments dialog box. In the left pane, verify the relevant comment is selected before you edit, format or delete.
3. Using arrows in a graph

   To add an arrow to a graph, click and then click the mouse button within the graph to position the base of the arrow.

   To delete an arrow from a graph, select the arrow and press Delete.
4. Using the User Notes Window

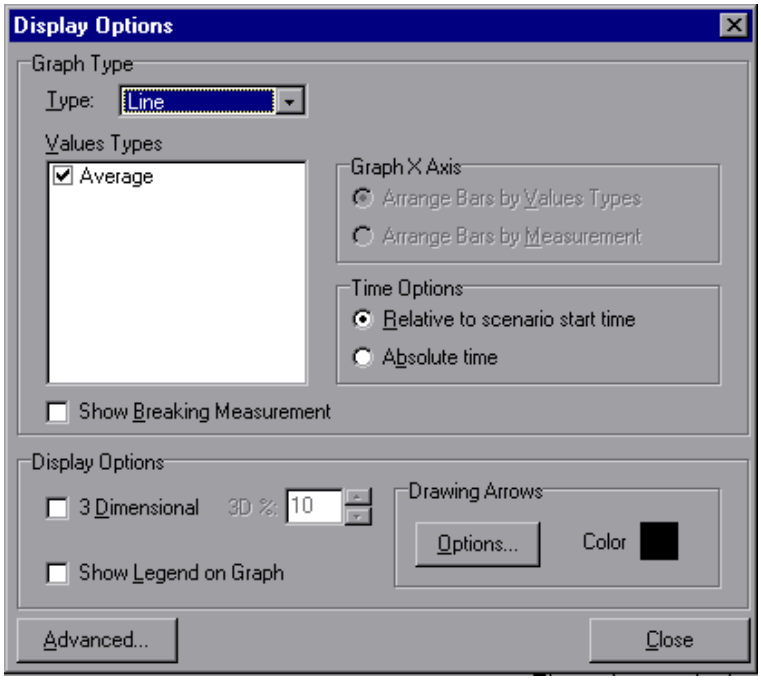
   In the User Notes window (**Windows > User Notes**), you can enter text about the graph or report that is currently open. The text in the User Notes window is saved with the session.

   To view the text that you entered for a specific graph or report, select the relevant graph or report and open the User Notes window (**Windows > User Notes**).

## Display Options Dialog Box

This dialog box enables you to select the graph type and configure the display of the graph.

> **Note:** This option is not available for all graph types.




| | |
|---|---|
| **To access** | **View > Display Options** |
| **See also** | "Editing Main Chart Dialog Box (Display Options Dialog Box)" on the next page |

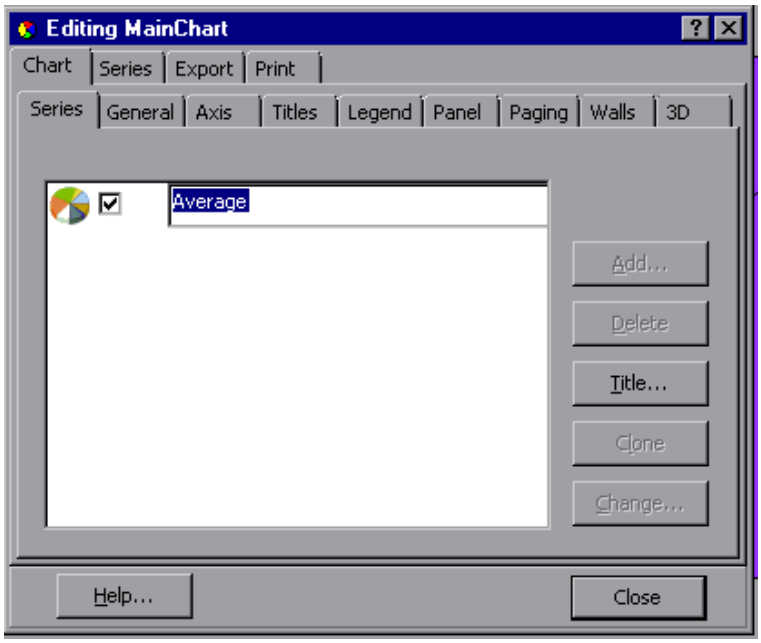
User interface elements are described below:

| UI Elements | Description |
|---|---|
| **Type** | Select the type of graph to display from the drop-down list. |
| **Values Types** | Select the type of display information from the list of available values. For example, a bar graph displaying Average Transaction Response Time can be configured to display minimum, maximum, average, STD, count, and sum averages. |
| **Graph X Axis (Bar graphs only)** | Select the bar arrangement along the x-axis. You can arrange the bars by value types or measurement. |

| UI Elements | Description |
|---|---|
| **Time Options** | Select the way in which the graph shows the Elapsed Scenario Time on the x-axis. You can choose an elapsed time relative to the beginning of the scenario or an elapsed time from the absolute time of the machine's system clock. |
| **Show Breaking Measurement** | Select this check box to display the name and properties of the breaking measurement at the top of the graph (disabled by default). |
| **3 Dimensional** | Select this check box to enable a 3-dimensional display of the graph. |
| **3D %** | Specify a percentage for the 3-dimensional aspect of lines in the graph. This percentage indicates the thickness if the bar, grid, or pie chart. |
| **Show Legend on Graph** | Select this check box to display a legend at the bottom of the graph. |
| **Drawing Arrows** | Allows you to configure the style, color, and width of arrows you draw to highlight graph information. |
| Advanced... | Opens the **Editing MainChart** dialog box. For more information, see "Editing Main Chart Dialog Box (Display Options Dialog Box)" below. |

## Editing Main Chart Dialog Box (Display Options Dialog Box)

This dialog box enables you to configure the look and feel of your graph as well as its title and the format of the data.



| | |
|---|---|
| **To access** | **View > Display Options > Advanced** button |
| **See also** | "Display Options Dialog Box" on page 1314 |

### Series Tab (Editing MainChart Dialog Box)

This page enables you to control the appearance of the individual points plotted in the graph.

| UI Element | Description |
|---|---|
| **Format tab** | Set the border color, line color, pattern, and invert property for the lines or bars in your graph. |
| **Point tab** | Set the size, color, and shape of the points that appear within your line graph. |
| **General tab** | Select the type of cursor, the format of the axis values, and show/hide settings for the horizontal and vertical axes. |
| **Marks tab** | Configure the format for each point in the graph. |

### Chart Tab (Editing MainChart Dialog Box)

This tab enables you to configure the look and feel of your entire graph.

| UI Element | Description |
|---|---|
| **Series tab** | Select the graph style (for example, bar or line), the hide/show settings, line and fill color, and the title of the series. |
| **General tab** | Select options for print preview, export, margins, scrolling, and magnification. |
| **Axis tab** | Select which axes to show, as well as their scales, titles, ticks, and position. |
| **Titles tab** | Set the title of the graph, its font, background color, border, and alignment. |
| **Legend tab** | Set all legend related settings, such as position, fonts, and divider lines. |
| **Panel tab** | Show the background panel layout of the graph. You can modify its color, set a gradient option, or specify a background image. |

| UI Element | Description |
|---|---|
| **Paging tab** | Set all page related settings, such as amount of data per page, scale, and page numbering. These settings are relevant when the graph data exceeds a single page. |
| **Walls tab** | Set colors for the walls of 3-dimensional graphs. |
| **3D** | Select the 3-dimensional settings, offset, magnification, and rotation angle for the active graph. |

### Export tab

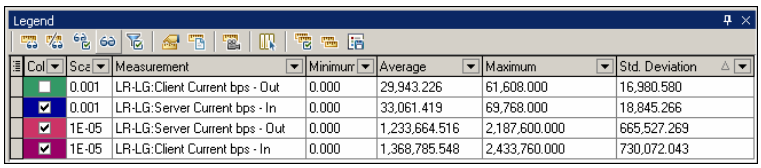
Enables you to store the current graph to an image file in the format of your choice—BMP, JPG, or EMF. You can also export the graph's data to HTML, Excel, or XML.

### Print tab

Enables you to print only the graph itself without the legend and other data such as the User Notes.

## Legend Window

This window enables you to configure the color, scale, minimum, maximum, average, median, and standard deviation of each measurement appearing in the graph. Access from **Analysis Window > Legend**.




**Tip:**

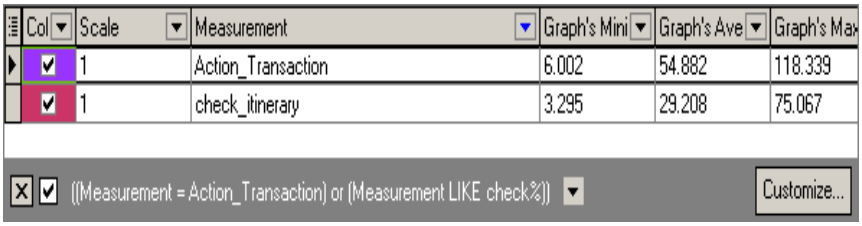
**Filtering:** To show only certain values, click the down arrow in the selected column and click **Custom**. The Custom Filter dialog box opens. For details, see "Custom Filter Dialog Box" on page 1350.

**Sorting:** To sort the measurements by a specific metrics, select a column header once to display the measurements in ascending order. Click it again to display them in descending order.

## Legend Toolbar

User interface elements are described below:

| UI Element | Description |
|---|---|
| | **Show.** Displays the selected measurements in the graph. |
| | **Hide.** Hides the selected measurements in the graph. |
| | **Show only Selected.** Displays the highlighted measurement only. |
| | **Show All.** Displays all the available measurements in the graph. |
| | **Filter.** Filters the graph by the measurements selected in the Legend window. You can select multiple measurements. To clear the filter, select **View > Clear Filter/Group By**. |
| | **Configure.** Opens the Measurement Options dialog box that enables you to configure measurement options (for example, set color and measurement scale). For more information, see "Measurement Options Dialog Box" on page 1321. |
| | **Show Description.** Opens the Measurement Description dialog box that displays the name, monitor type, and description of the selected measurement. For more information, see "Measurement Description Dialog Box" on page 1320. |
| | **Animate.** Displays the selected measurement as a flashing line. |
| | **Configure Columns.** Opens the Legend Columns Options dialog box that enables you to select the columns to display in the Legend window. |
| | **Copy Selection.** Copies the selected rows to the clipboard. You can paste the data in a text file or a spreadsheet. |
| | **Copy All.** Copies all of the legend data to the clipboard, regardless of what is selected. You can paste the data in a text file or a spreadsheet. |
| | **Export.** Saves the legend data to a CSV file. |

| UI Element | Description |
|---|---|
| **<Custom filter>** | After adding a custom filter (by expanding the down arrow in the column headers), the window shows them at the bottom of the legend. Click the **x** button to remove the filter, or clear the check box to disable it temporarily. For details, see "Custom Filter Dialog Box" on page 1350.<br>  |
| **Customize** | Opens the Filter Builder and allows you to save your filter settings to a file. |

## Legend grid shortcut menu

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Show** | Displays the selected measurements in the graph. |
| **Hide** | Hides the selected measurements in the graph. |
| **Show only Selected** | Displays the highlighted measurement only. |
| **Show All** | Displays all the available measurements in the graph. |
| **Filter** | Filters the graph by the measurements selected in the Legend window. You can select multiple measurements. To clear the filter, select **View > Clear Filter/Group By**. |
| **Configure** | Opens the Measurement Options dialog box that enables you to configure measurement options (for example, set color and measurement scale). For more information, see "Measurement Options Dialog Box" on page 1321. |
| **Show Description** | Opens the Measurement Description dialog box that displays the name, monitor type, and description of the selected measurement. For more information, see "Measurement Description Dialog Box" on the next page. |
| **Animate** | Displays the selected measurement as a flashing line. |

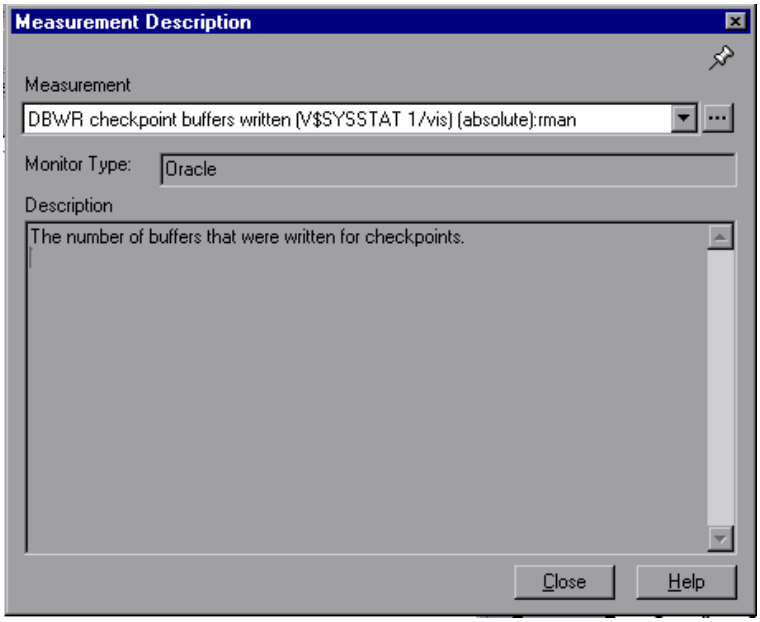
| UI Element | Description |
|---|---|
| **Auto Correlate** | Opens the Auto Correlate dialog box that enables you to correlate the selected measurement with other monitor measurements in the load test scenario. For more information on auto correlation, see "Auto Correlating Measurements" on page 1332. |
| **Configure Columns** | Opens the Legend Columns Options dialog box that enables you to select the columns to display in the Legend window. |
| **Web Page Diagnostics for <selected measurement>** | Displays a Web Page Diagnostics graph for the selected transaction measurement (only available for the Average Transaction Response Time and Transaction Performance Summary graphs). |
| **Break down** | Displays a graph with a breakdown of the selected page (only available for the Web Page Diagnostics graphs). |

## See also:

- "Measurement Description Dialog Box" below
- "Measurement Options Dialog Box" on the next page

## Measurement Description Dialog Box

This dialog box shows you additional information about the selected measurement.




| To access | **Legend Toolbar >** |
|---|---|

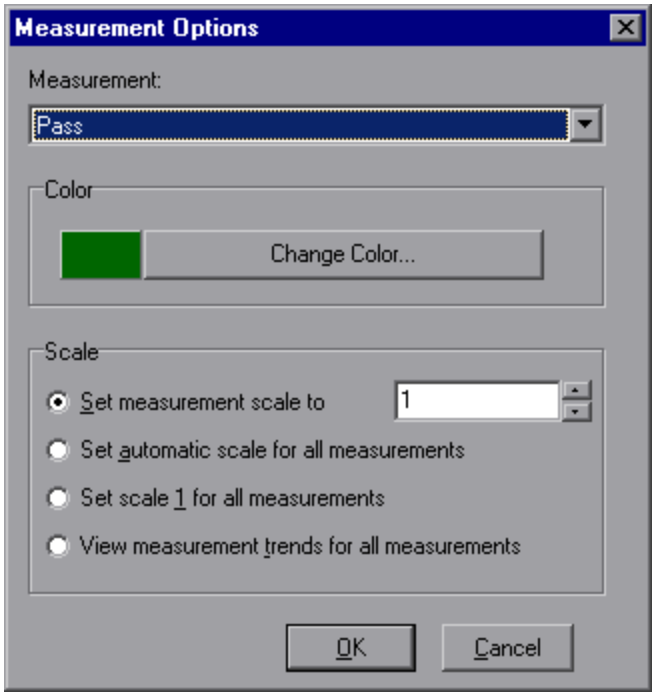
| See also | • "Legend Window" on page 1317<br>• "Measurement Options Dialog Box" below |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Measurement** | Displays the name of the selected measurement. Click the drop-down arrow to select a different measurement. |
| **Monitor Type** | Displays the type of monitor used to obtain the selected measurement. |
| **Description** | Displays a description of the selected monitored measurement. |
| **SQL** | If an SQL logical name is in use, displays the full SQL statement. |

## Measurement Options Dialog Box

This dialog box enables you to set the color and the scale for any measurement of the graph you selected.



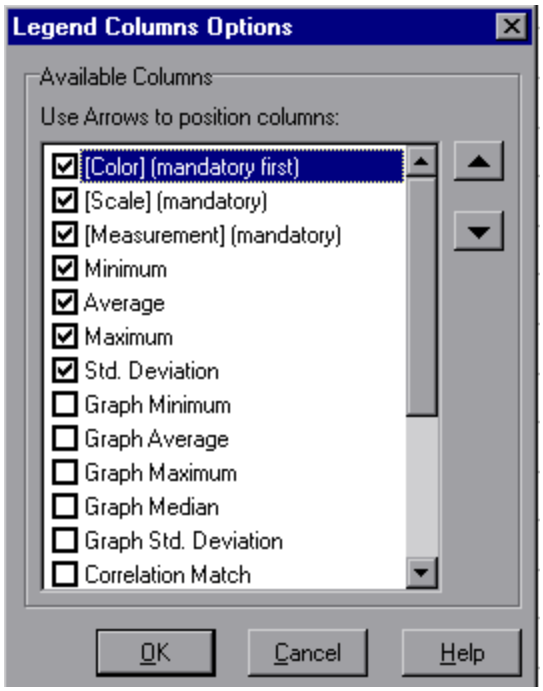

| To access | **Legend Toolbar >** |
|---|---|
| See also | • "Legend Window" on page 1317<br>• "Measurement Description Dialog Box" on the previous page |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Measurement** | Select a measurement to configure. |
| **Change Color** | Select a new color for the selected measurement. |
| **Scale** | Select the desired scale option:<br>• Set measurement scale to x. Select the scale with which you want to view the selected measurement.<br>• Set automatic scale for all measurements. Uses an automatic scale optimized to display each measurement in the graph.<br>• Set scale 1 for all measurements. Sets the scale to one for all measurements in the graph.<br>• View measurement trends for all measurements. Standardizes the y-axis values in the graph, according to the following formula: New Y value = (Previous Y Value - Average of previous values) / STD of previous values. |

## Legend Columns Options Dialog Box

This dialog box enables you to select the columns to be displayed.



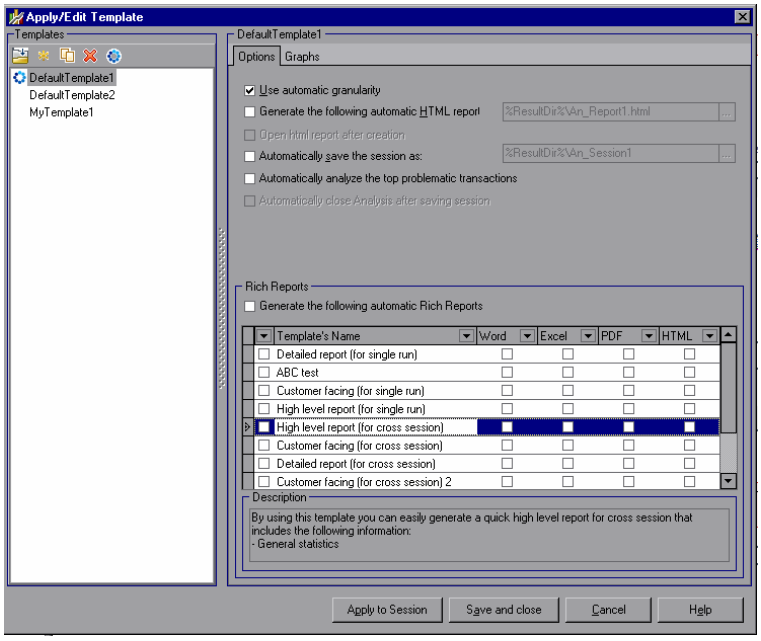

| To access | View > Legend Columns |
|---|---|
| **See also** | "Legend Window" on page 1317 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Available Columns** | Select or deselect the check boxes to the left of the column names to show or hide the columns respectively.<br><br>**Notes:**<br>• The Color, Scale, and Measurement columns are mandatory and cannot be deselected.<br>• To rearrange the order in which the columns appear (from left to right), you use the vertical arrows to the right of the Available Columns list to place the columns in the desired order. |

## Apply/Edit Template Dialog Box

This dialog box enables you to configure template settings and select report template options. Using this dialog box, you can create new templates, open existing ones, and set the default template for your sessions.




| To access | Tools > Templates |
|---|---|

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Templates** | • Browse for a template.<br>• Add a new template. Enter the title of the new template in the **Add new template** dialog box.<br>• Duplicate the selected template.<br>• Delete the selected template.<br>• Set the selected template as the default. |
| **Use automatic granularity** | Applies the default Analysis granularity (one second) to the template. For information about setting Analysis granularity, see "Changing the Granularity of the Data" on page 1330. |
| **Generate the following automatic HTML report** | Generates an HTML report using the template. Specify or select a report name. For information about generating HTML reports, see "HTML Reports" on page 1551. |
| **Open html report after creation** | If you selected the option of generating an automatic HTML report, select this option to automatically open the HTML report after it is created. |
| **Automatically save the session as** | Automatically saves the session using the template you specify. Specify or select a file name. |
| **Automatically analyze the top problematic transactions** | Automatically generates Transaction Analysis reports for the transactions with the worst SLA violations. Reports are generated for a maximum of five transactions. For more information about Transaction Analysis reports, see "Analyze Transactions Dialog Box" on page 1536. |
| **Automatically close Analysis after saving session** | Automatically closes Analysis after a session is automatically saved (using the previous option). This prevents the running of multiple instances of Analysis. |
| **Generate the following automatic Rich Reports** | The selected reports are added to the template. |

| UI Element | Description |
|---|---|
| **<check box on left of Template's Name>** | Select to add report template to selected template. The reports are added to the session. |
| **Word** | Generates a report using the selected report template to MS Word.<br>**Note:** Take into account that the content load may affect the table format within the MS Word document. |
| **Excel** | Generates a report using the selected report template to Excel. |
| **PDF** | Generates a report using the selected report template to PDF. |
| **HTML** | Generates a report using the selected report template to HTML. |
| **Graphs tab** | Displays the list of graphs that are included in the template. When the template is applied to a session, the graphs are displayed under Graphs in Session Explorer. If there is no data in the session, the graphs are not created. |
| **Apply to Session** | Applies your changes to the current analysis session without closing the dialog box. |

## Color Palettes

Color Palettes allow you to define the colors that will be used in Analysis graphs and to allocate those colors to specific series. There is a general, default palette and you can also define a Color Palette for a specific session. You can add new colors to a palette and delete existing colors from a palette, but a palette must contain at least thirty two colors.

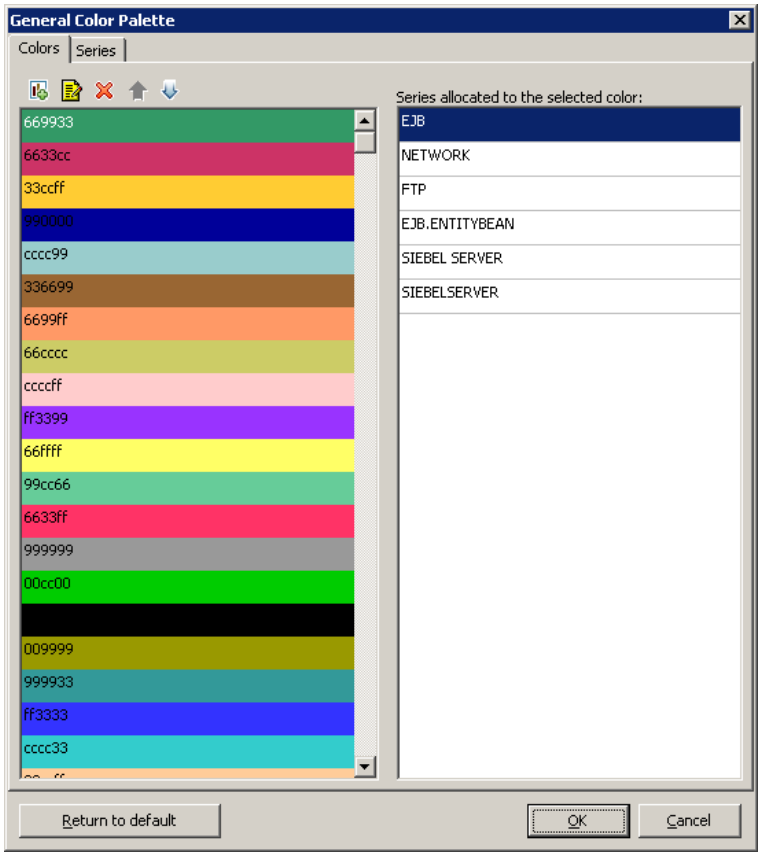
When a new session is created, or when you open an existing session that does not have a Graph Colors file, Analysis uses the general color palette. When you open an existing session that has a Graph Colors file, Analysis uses the file from the session folder.

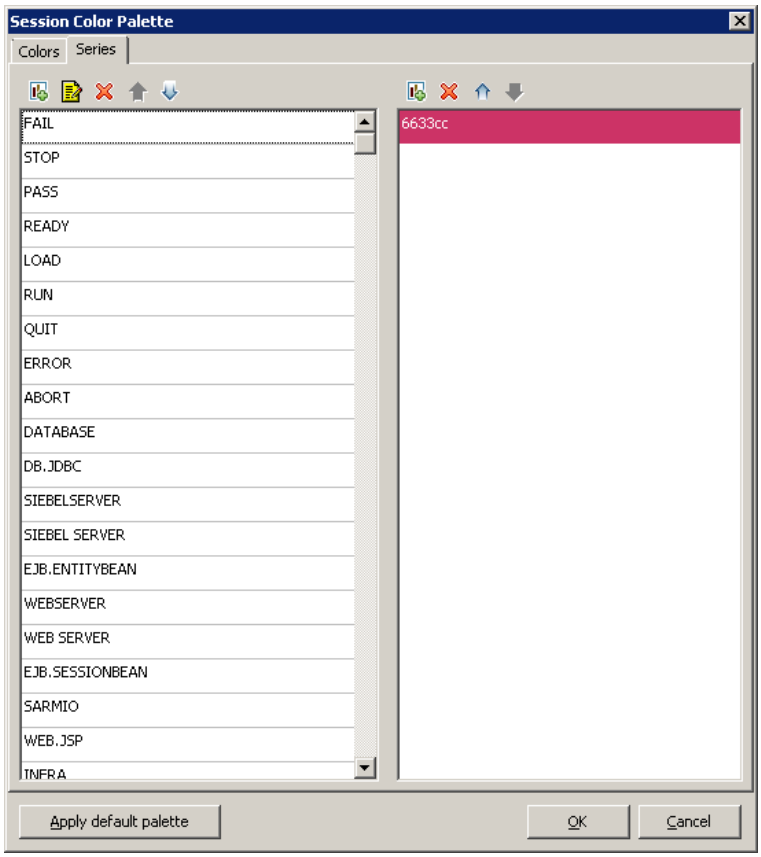
The colors are allocated to the graph in the order they appear in the palette. Colors allocated to a series, are used to represent graph elements for the series in the order the colors were allocated. To change the colors in the graph, update the palette, close and re-open the graph.

For more information, see "Color Palette Dialog Box" below.

## Color Palette Dialog Box

This dialog box enables you to configure the colors that will be used in graphs. You use the General Color Palette to define a default set of colors for all graphs and the Session Color Palette to define the set of colors for a specific session.


General Color Palette
Colors
Series
Series allocated to the selected color:
669933
6633cc
33ccff
990000
cccc99
336699
6699ff
66cccc
ccccff
ff3399
66ffff
99cc66
6633ff
999999
00cc00
009999
999933
ff3333
cccc33
EJB
NETWORK
FTP
EJB.ENTITYBEAN
SIEBEL SERVER
SIEBELSERVER
Return to default
OK
Cancel


Session Color Palette
Colors
Series
FAIL
STOP
PASS
READY
LOAD
RUN
QUIT
ERROR
ABORT
DATABASE
DB.JDBC
SIEBELSERVER
SIEBEL SERVER
EJB.ENTITYBEAN
WEBSERVER
WEB SERVER
EJB.SESSIONBEAN
SARMIO
WEB.JSP
INFRA
6633cc
Apply default palette
OK
Cancel

| | |
|---|---|
| **To access** | • **Tools > General Color Palette**<br>• **Tools > Session Color Palette** |
| **See also** | "Color Palettes" on page 1325 |

User interface elements are described below:

| UI Elements> | Description |
|---|---|
| Return to default | Restores the palette to the currently saved General Palette.<br>This button appears on the General Color Palette, not on the Session Color Palette. |
| Apply default palette | Applies the default palette as the session palette.<br>This button appears on the Session Color Palette, not on the General Color Palette. |
| **Colors tab** | Allows you to configure the colors on the palette. |
| | Add a new color to the palette. |
| | Replace an existing color with a new color. |
| | Delete a color from the palette. |
| | Move the color upwards. |
| | Move the color downwards. |
| **Series tab - left pane** | Allows you to configure the series on the palette. |
| | Add a new series to the palette. |
| | Edit a series. |
| | Delete a series from the palette. |
| | Move the series upwards. |
| | Move the series downwards. |
| **Series tab - right pane** | Allows you to define colors for the selected series. |

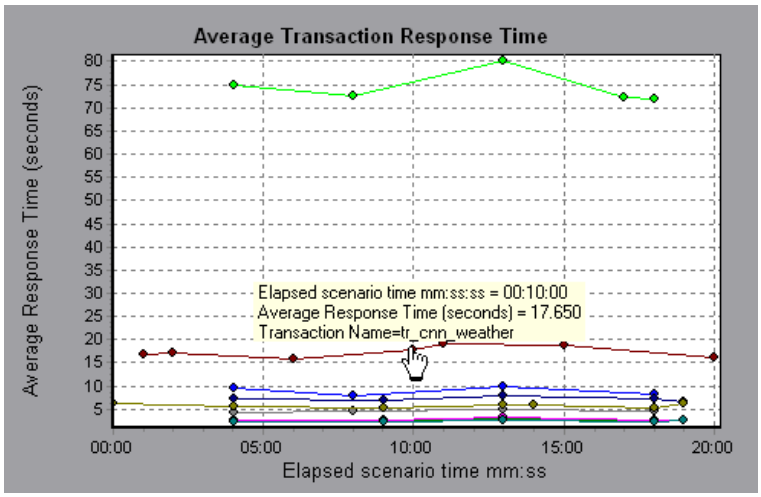
| UI Elements> | Description |
|---|---|
| | Add a color to the series. |
| | Delete a color from the series. |
| | Move the color upwards. |
| | Move the color downwards. |

## Working with Analysis Graph Data

Analysis contains several utilities that enable you to manage graph data to most effectively view the displayed data.

### Determining a Point's Coordinates

You can determine the coordinates and values at any point in a graph. Place the cursor over the point you want to evaluate and Analysis displays the axis values and other grouping information.

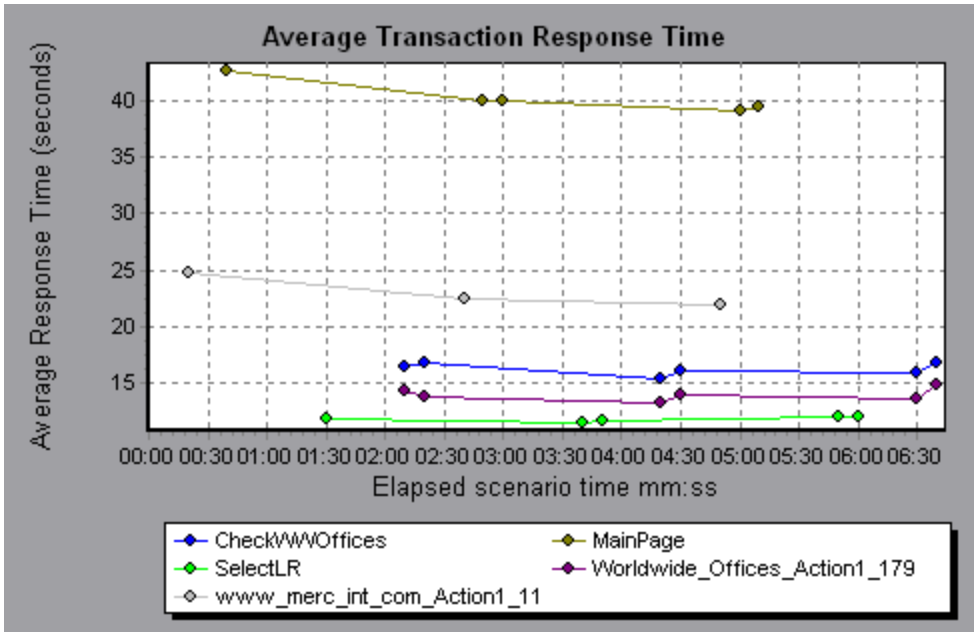



### Drilling Down in a Graph

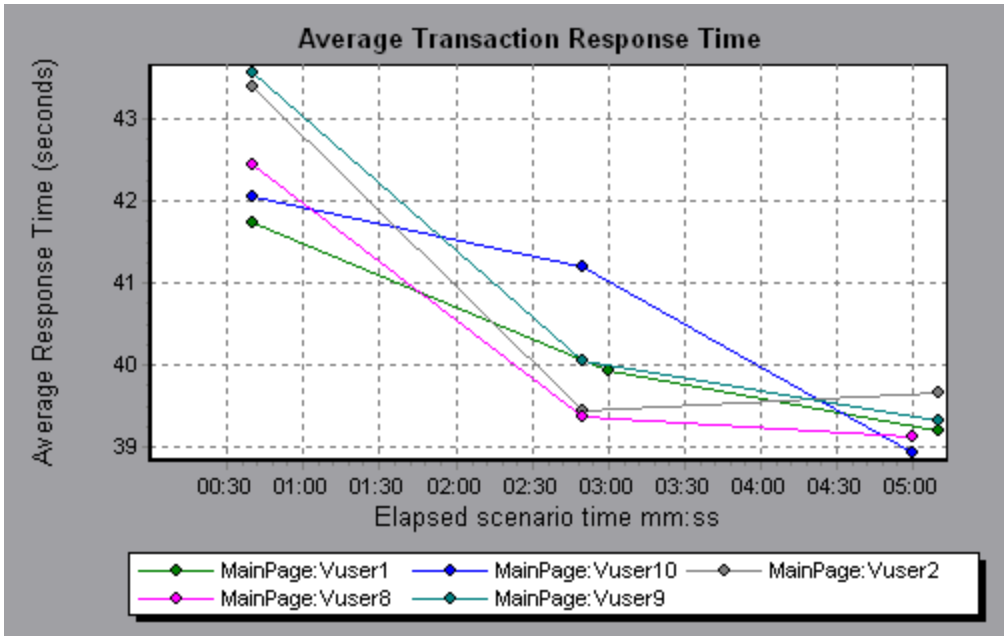
*Drill down* enables you to focus on a specific measurement within your graph and display it according to a desired grouping. The available groupings depend on the graph. For example, the Average Transaction Response Time graph shows one line per transaction. To determine the response time for each Vuser, you drill down on one transaction and sort it according to Vuser ID. The graph displays a separate line for each Vuser's transaction response time.

**Note:** The drill down feature is not available for the Web Page Diagnostics graph.

The following graph shows a line for each of five transactions.




When you drill down on the MainPage transaction, grouped by Vuser ID, the graph displays the response time only for the MainPage transaction, one line per Vuser.




You can see from the graph that the response time was longer for some Vusers than for others.

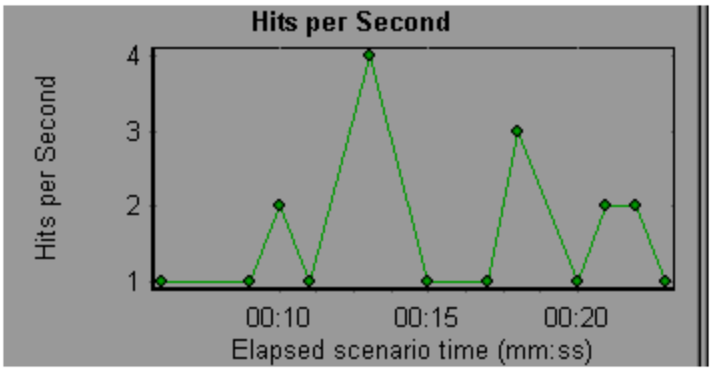
To determine the response time for each host, you drill down on one transaction and sort it according to host. The graph displays a separate line for the transaction response time on each host. For more information on drilling down in a graph, see "Getting more information from Graph Data" on page 1333.

## Changing the Granularity of the Data

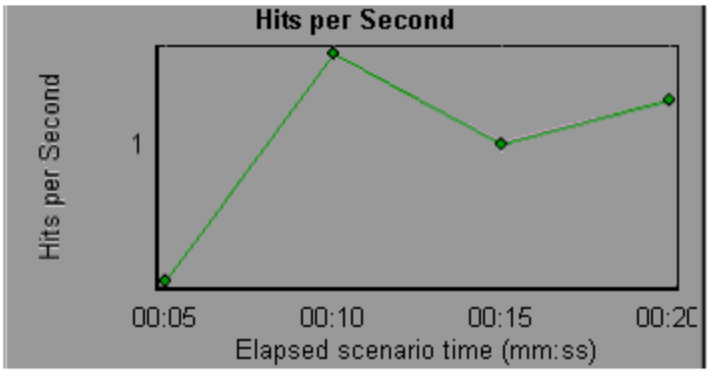
You can make the graphs easier to read and analyze by changing the granularity (scale) of the *x-axis*. The maximum granularity is half of the graph's time range. To ensure readability and clarity, Analysis automatically adjusts the minimum granularity of graphs with ranges of 500 seconds or more.

In the following example, the Hits per Second graph is displayed using different granularities. The y-axis represents the number of hits per second within the granularity interval. For a granularity of 1, the y-axis shows the number of hits per second for each one second period of the load test scenario.

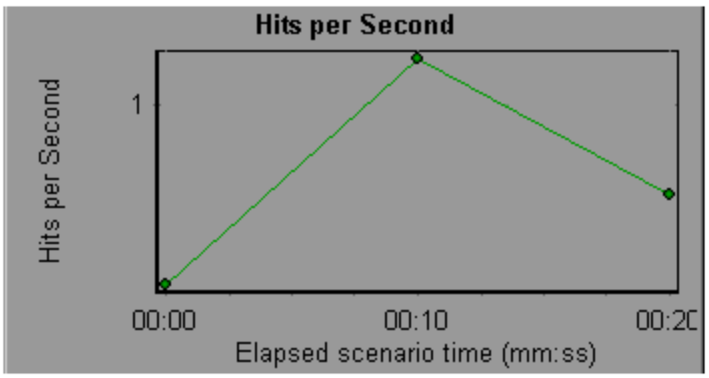
For a granularity of 5, the y-axis shows the number of hits per second for every five-second period of the scenario.




GRANULARITY=1




GRANULARITY=5




GRANULARITY=10

In the above graphs, the same load test scenario results are displayed in a granularity of 1, 5, and 10. The lower the granularity, the more detailed the results. For example, using a low granularity as in the upper graph, you see the intervals in which no hits occurred. It is useful to use a higher granularity to study the overall Vuser behavior throughout the scenario.

By viewing the same graph with a higher granularity, you can see that overall, there was an average of approximately 1 hit per second.

## Viewing Measurement Trends

You can view the pattern of a line graph more effectively by standardizing the graph's y-axis values. Standardizing a graph causes the graph's y-axis values to converge around zero. This cancels the

measurements' actual values and allows you to focus on the behavior pattern of the graph during the course of the load test scenario.

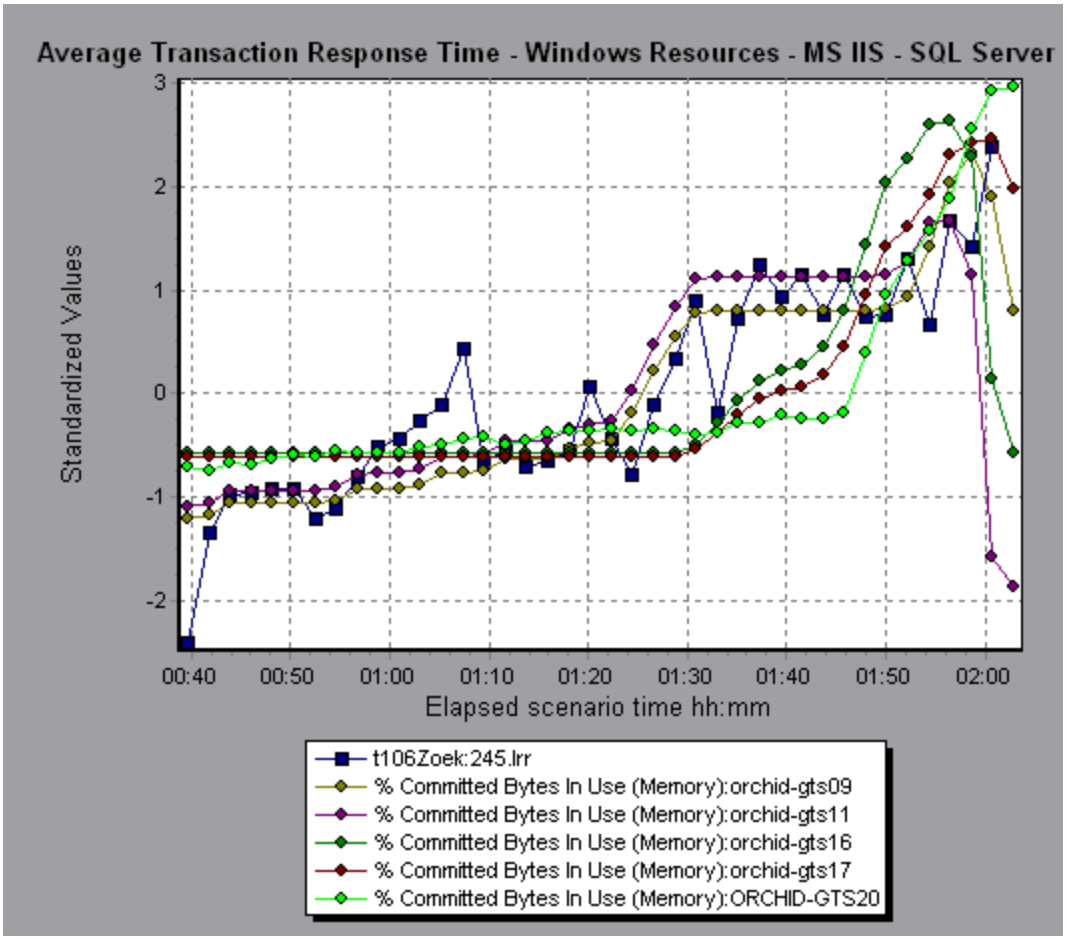
Analysis standardizes the y-axis values in a graph according to the following formula:

**New Y value = (Previous Y Value - Average of previous values) / STD of previous values**

## Auto Correlating Measurements

You can detect similar trends among measurements by correlating a measurement in one graph with measurements in other graphs. Correlation cancels the measurements' actual values and allows you to focus on the behavior pattern of the measurements during a specified time range of the load test scenario.

In the following example, the **t106Zoek:245.lrr** measurement in the Average Transaction Response Time graph is correlated with the measurements in the Windows Resources, Microsoft IIS, and SQL Server graphs. The five measurements most closely correlated with **t106Zoek:245.lrr** are displayed in the graph below.




**Note:** This feature can be applied to all line graphs except the Web Page Diagnostics graph.

## Viewing Raw Data

You can view the actual raw data collected during test execution for the active graph. The Raw Data view is not available for all graphs.

Viewing the raw data can be especially useful in the following cases:

- To determine specific details about a peak—for example, which Vuser was running the transaction that caused the peak value(s).
- To perform a complete export of unprocessed data for your own spreadsheet application.

For user interface details, click "Graph/Raw Data View Table" on page 1339.

## Getting more information from Graph Data

There are several techniques you can use in Analysis to enable you to effectively view the displayed data.

### Determine a point's coordinates

To determine the coordinates and values at any point in a graph, place the cursor over the point you want to evaluate. Analysis displays the axis values and other grouping information.

### Drill down in a graph

Drill down enables you to focus on a specific measurement within your graph and display it according to the desired grouping.

1. Right-click on a line, bar, or segment in the graph, and select **Drill Down**. The **Drill Down Options** dialog box opens, listing all of the measurements in the graph.
2. Select a measurement for drill down.
3. From the **Group By** box, select a group by which to sort.
4. Click **OK**. Analysis drills down and displays the new graph.
   - To undo the last drill down settings, right-click and select **Undo Set Filter/Group By** from the context menu.
   - To perform additional drill-downs, repeat steps 1 to 4.
   - To clear all filter and drill down settings, right-click and select **Clear Filter/Group By** from the context menu.

### Filter the data

This task describes how to filter the data and create custom filters.

1. In the Legend window, click the column header of the measurement you want to use as a base for the filter.
2. To show a single entry, expand the drop-down list and select that entry.
3. To create a custom filter, select **Custom** in the drop-down list. The Custom Filter dialog box opens.

4. Select an evaluation expression and provide a value. To use wildcards, use an underscore (_) to represent a single character and % for multiple characters. For details, see "Custom Filter Dialog Box" on page 1350.
5. To provide additional criteria, select a logical operator, **AND** or **OR** and set up the second expression.

### Change the granularity of the data

This task describes how to change the granularity of a graph.

1. Click inside a graph.
2. Select **View** > **Set Granularity**, or click the **Set Granularity** button . The Granularity dialog box opens.
3. Enter the granularity of the x-axis and select a time measurement. The maximum granularity is half of the graph's time range.
4. To ensure readability and clarity, LoadRunner automatically adjusts the minimum granularity of graphs with ranges of 500 seconds or more.
5. Click **OK**.

### View measurement trends

This task describes how to activate the View Measurements Trends option from a line graph.

1. Select **View > View Measurement Trends**, or right-click the graph and choose **View Measurement Trends**. Alternatively, you can select **View > Configure Measurements** and check the View measurement trends for all measurements box.

> **Note:** The standardization feature can be applied to all line graphs except the Web Page Diagnostics graph.

2. View the standardized values for the line graph you selected. The values in the Minimum, Average, Maximum, and Std. Deviation legend columns are real values.

To undo the standardization of a graph, repeat step 1.

> **Note:** If you standardize two line graphs, the two y-axes merge into one y-axis.

### Auto correlate measurements

You can detect similar trends among measurements by correlating a measurement in one graph with measurements in other graphs. Correlation cancels the measurements' actual values and allows you to focus on the behavior pattern of the measurements during a specified time range of the load test scenario.

1. From a graph or legend, right-click the measurement you want to correlate and choose **Auto Correlate**. The Auto Correlate dialog box opens with the selected measurement displayed in the graph.
2. Select a suggested time range method and time range.
3. If you applied a time filter to your graph, you can correlate values for the complete scenario time range by clicking the **Display** button in the upper right-hand corner of the dialog box.
4. To specify the graphs you want to correlate with a selected measurement and the type of graph output to be displayed, perform the following:
   - Select the **Correlation Options** tab.
   - Select the graphs to correlate, the data interval, and output options, as described in "Drill Down Options Dialog Box" below.
   - On the **Time Range** tab, click **OK**. Analysis generates the correlated graph you specified. Note the two new columns—**Correlation Match** and **Correlation**—that appear in the Legend window below the graph.

   To specify another measurement to correlate, select the measurement from the Measurement to Correlate box at the top of the Auto Correlate dialog box.

   The minimum time range should be more than 5% of the total time range of the measurement. Trends which are smaller than 5% of the whole measurement will be contained in other larger segments.

   Sometimes, very strong changes in a measurement can hide smaller changes. In cases like these, only the strong change is suggested, and the **Next** button is disabled.

> **Note:** This feature can be applied to all line graphs except the Web Page Diagnostics graph.

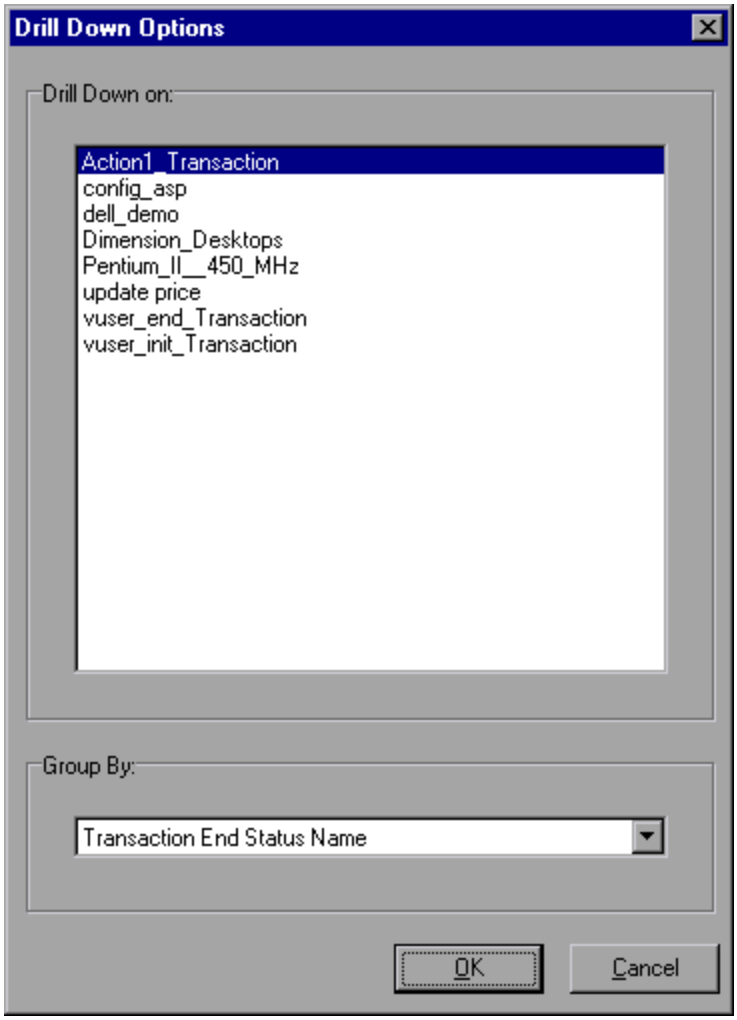
### Save graph as image

You can save a graph as an image file.

1. Right-click the graph and select **Save as image**.
2. In the Save as Image dilaog box, select the type of image (.jpeg/.png).
3. Select the location to save the file and click **Save**.

## Drill Down Options Dialog Box

This dialog box lists all the measurements in the graph.



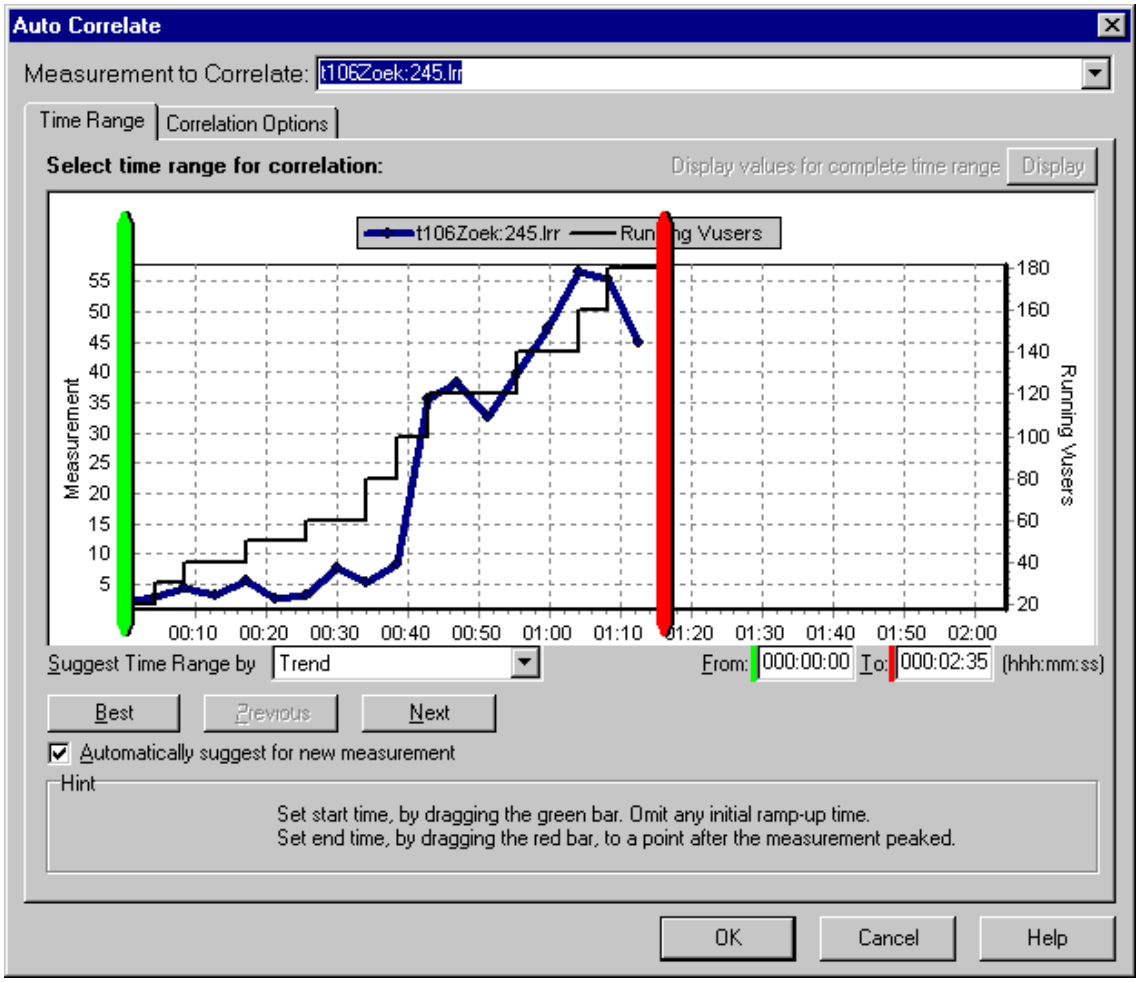

| | |
|---|---|
| **To access** | <Right-click> graph line/bar/segment **> Drill Down** |
| **See also** | "Drilling Down in a Graph" on page 1328 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Drill Down on** | Filter graph by selected transaction. |
| **Group By** | The selected transaction is sorted by selected criteria. |

## Auto Correlate Dialog Box

This dialog box enables you to configure settings used to correlate measurements from the selected graph with measurements in other graphs.




| | |
|---|---|
| **To access** | Right-click on a graph and select **> Auto Correlate** from the context menu. |
| **Important information** | • You can also use the green and red vertical drag bars to specify the start and end values for the scenario time range.<br>• The granularity of the correlated measurements graph may differ from that of the original graph, depending on the scenario time range defined. |

## Time Range Tab

The **Time Range** tab of the Auto Correlate dialog box enables you to specify a load test scenario time range for the correlated measurement graph.

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Measurement to Correlate** | Select the measurement you want to correlate. |

| UI Element | Description |
|---|---|
| **Display values for complete time range** | Click **Display** to correlate values for the complete scenario time range. This option is available only if you applied a time filter to your graph. |
| **Suggest Time Range By** | Analysis automatically demarcates the most significant time period for the measurement in the scenario.<br>• **Trend.** Demarcated an extended time segment which contains the most significant changes.<br>• **Feature.** Demarcates a smaller dimension segment which forms the trend. |
| **Best** | Choose the time segment most dissimilar to its adjacent segments. |
| **Next** | Suggest the next time segment for auto correlation. Each suggestion is successively less dissimilar. |
| **Previous** | Return to the previous suggestion of a time segment. |
| **Automatically suggest for new measurement** | Generates new suggestions each time that the Measurement to Correlate item changes. |
| **From** | Specify a start value (in hh:mm:ss format) for the desired scenario time range. |
| **To** | Specify an end value (in hh:mm:ss format) for the desired scenario time range. |

### Correlation Options tab

You use the **Correlation Options** tab to set the graphs to correlate, the data interval, and the output options.

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Select Graphs for Correlation** | Select the graphs whose measurements you want to correlate with your selected measurement. |
| **Data Interval** | Calculate the interval between correlation measurement polls.<br>• **Automatic**. Uses an automatic value, determined by the time range.<br>• **Correlate data based on X second intervals**. Enter a fixed value. |

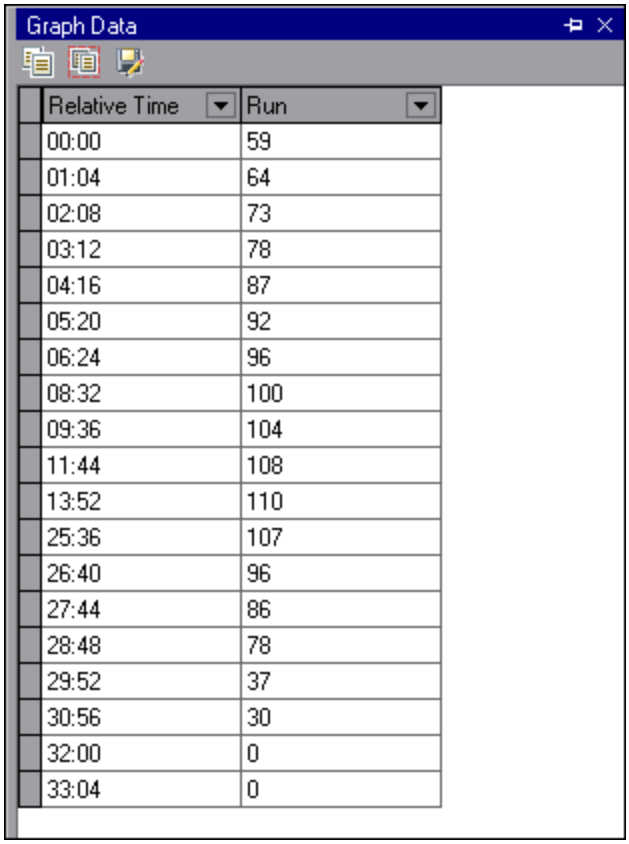
| UI Element | Description |
|---|---|
| **Output** | Choose the level of output displayed.<br>• **Show the X most closely correlated measurements.** Displays only the specified number of measurements most closely related to the selected measurement. The default value is 5.<br>• **Show measurements with an influence factor of at least X%.** Displays only those measurements that converge to the specified percent with the selected measurement. The default value is 50%. |

### See also:

- "Auto Correlating Measurements" on page 1332

## Graph/Raw Data View Table

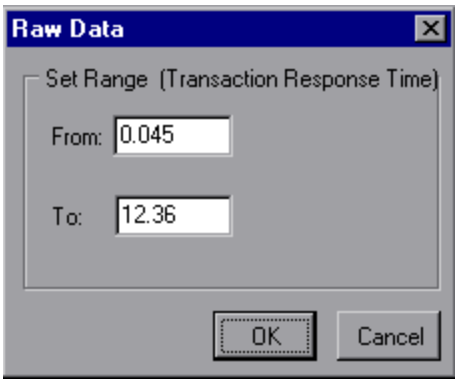
You can view graph data in spreadsheet view or raw data view. The data is instantly displayed on request.




| Relative Time | Run |
|---|---|
| 00:00 | 59 |
| 01:04 | 64 |
| 02:08 | 73 |
| 03:12 | 78 |
| 04:16 | 87 |
| 05:20 | 92 |
| 06:24 | 96 |
| 08:32 | 100 |
| 09:36 | 104 |
| 11:44 | 108 |
| 13:52 | 110 |
| 25:36 | 107 |
| 26:40 | 96 |
| 27:44 | 86 |
| 28:48 | 78 |
| 29:52 | 37 |
| 30:56 | 30 |
| 32:00 | 0 |
| 33:04 | 0 |

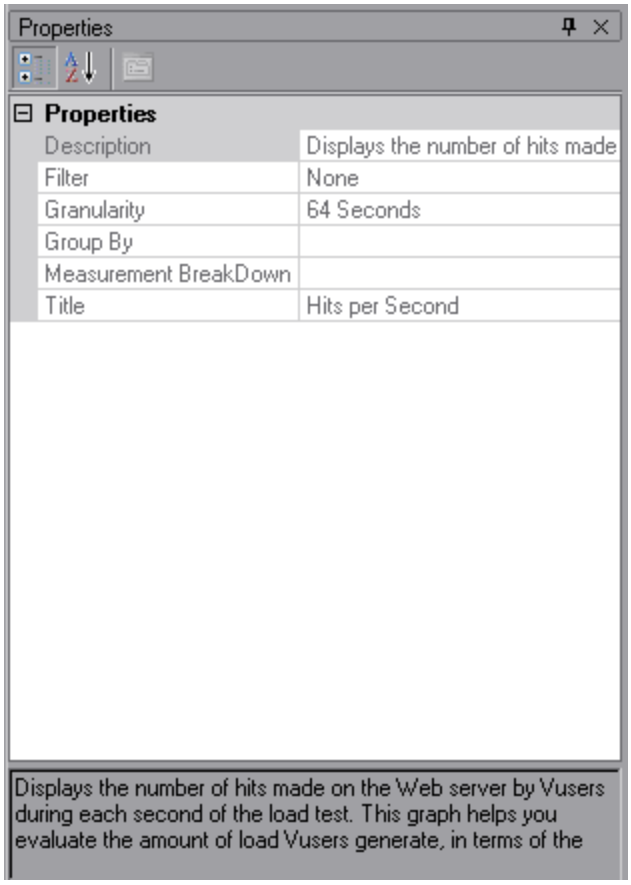
| | |
|---|---|
| **To access** | Click the appropriate tab on the right border of the Analysis window or do one of the following:<br>• **Windows > Graph Data**<br>• **Windows > Raw Data** |
| **Note** | Raw Data is not available for all graphs. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Copies the data that you have selected. |
| | Copies the spreadsheet to the clipboard. You can paste to a spreadsheet. |
| | Saves the spreadsheet data to an Excel or CSV file. In Excel, you can generate your own customized graphs. |
| | Use the buttons on the toolbar to navigate through the table, and mark any records for future reference. |
| **Relative Time** | The first column in the Graph Data window. displays the elapsed scenario time (the x-axis values). The following columns displays the relative y-axis values for each measurement represented on the graph. |
| **Raw Data dialog box** | In **Set Range**, set a time range.<br>  |

## Graph Properties Pane

This pane displays the details of the graph or report selected in the Session Explorer. Fields that appear in black are editable. When you select an editable field, an edit button is displayed next to the selected field value.




| | |
|---|---|
| **To access** | One of the following:<br>• **Windows > Properties**<br>• Select a graph in the Session Explorer, and select **Properties** from the right-click menu. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | Enables you to edit the value for the selected field. |
| **Graph fields** | • **Filter**. Shows configured filter.<br>• **Granularity.** Shows configured granularity.<br>• **Group By**. Shows the filter for selected group.<br>• **Measurement Breakdown**. Shows the measurements of the graph.<br>• **Title**. Shows the name of the graph in the graph display window. |

| UI Element | Description |
|---|---|
| **Summary Report fields** | • **Description**. A short summary of what is included in the summary report.<br>• **Filter**. Shows configured filter for the summary report.<br>• **Percentile.** The Summary Report contains a percentile column showing the response time of 90% of transactions (90% of transactions that fall within this amount of time). To change the value of the default 90 percentile, enter a new figure in the **Transaction Percentile** box.<br>• **Title**. The name of the summary report. |
| **Transaction Analysis Report fields** | When clicking the edit button for some of the fields, the Analyze Transaction Settings dialog box opens, enabling you to edit some of the Analyze Transaction settings. |

# Filtering and Sorting Graph Data

## Filtering Graph Data Overview

You can filter graph data to show fewer transactions for a specific segment of the load test scenario. More specifically, you can display four transactions beginning from five minutes into the scenario and ending three minutes before the end of the scenario.

You can filter for a single graph, in all graphs in a load test scenario, or in the summary graph.

The available filter conditions differ for each type of graph. The filter conditions also depend on your scenario. For example, if you only had one group or one load generator machine in your scenario, the Group Name and Load Generator Name filter conditions do not apply.

> **Note:** You can also filter merged graphs. The filter conditions for each graph are displayed on separate tabs.

## Sorting Graph Data Overview

You can sort graph data to show the data in more relevant ways. For example, Transaction graphs can be grouped by the Transaction End Status, and Vuser graphs can be grouped by Scenario Elapsed Time, Vuser End Status, Vuser Status, and Vuser ID.

You can sort by one or several groups—for example by Vuser ID and then Vuser status. The results are displayed in the order in which the groups are listed. You can change the grouping order by rearranging the list.

## Filter Conditions

You can filter different kinds of data with different condition.

### Common Filter Condition Options

The following filter conditions are common to many graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Host Name** | The name of the Host machine. Select one or more hosts from the drop-down list. |
| **Transaction End Status** | The end status of a transaction: *pass, fail, stop*. |
| **Scenario Elapsed Time** | The time that elapsed from the beginning to the end of the load test scenario. For more information about setting the time range, see "Scenario Elapsed Time Dialog Box" on page 1355. |
| **Vuser ID** | The Vuser ID. For more information, see "Vuser ID Dialog Box" on page 1356. |
| **Script Name** | The name of the script. |
| **Group Name** | The name of the group to filter by. |
| **Think Time** | The Think Time option in the graph filter for complete mode is turned off by default. The transaction time displayed shows pure time. |

### Vuser Graphs

You can apply the following filter conditions to Vuser graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Vuser Status** | The Vuser status: *load, pause, quit, ready, run* |
| **Vuser End Status** | The status of the Vuser at the end of the transaction: *error, failed, passed, stopped*. |
| **Number of Released Vusers** | The number of Vusers that were released. |
| **Rendezvous Name** | The name of the rendezvous point. |

### Error Graphs

You can apply the following filter conditions to Error graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Error Type** | The type of error (per error number). |
| **Parent Transaction** | The parent transaction. |
| **Line Number in Script** | The line number in the script. |

### Transaction Graphs

You can apply the following filter conditions to Transaction graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Transaction Name** | The name of the transaction. |
| **Transaction Response Time** | The response time of the transaction. |
| **Transaction Hierarchical Path** | The hierarchical path of the transaction. For more information on setting this condition, see "Hierarchical Path Dialog Box" on page 1354. |

### Web Resource Graphs

You can apply the following filter conditions to Web Resources graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Web Resource Name** | The name of the Web resource. |
| **Web Resource Value** | The value of the Web resource. |
| **Web Server Resource Name** | The name of the Web Server resource. |
| **Web Server Resource Value** | The value of the Web Server resource. |

### Web Page Diagnostics Graphs

You can apply the following filter conditions to Web Page Diagnostics graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Component Name** | The name of the component. |
| **Component Response Time** | The response time of the component. |

| Filter Condition | Filters the graph according to... |
|---|---|
| **Component DNS Resolution Time** | The amount of time the component needs to resolve the DNS name to an IP address, using the closest DNS server. |
| **Component Connection Time** | The time taken for the component to establish an initial connection with the Web server hosting the specified URL. |
| **Component First Buffer Time** | The time that passes from the component's initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server. |
| **Component Receive Time** | The time that passes until the component's last byte arrives from the server and the downloading is complete. |
| **Component SSL Handshaking Time** | The time take for the component to establish an SSL connection. (Applicable to HTTPS communication only.) |
| **Component FTP Authentication Time** | The time taken for the component to authenticate the client. (Applicable to FTP protocol communication only). |
| **Component Error Time** | The average amount of time that passes from the moment a component's HTTP request is sent until the moment an error message (HTTP errors only) is returned. |
| **Component Size (KB)** | The size of the component (in kilobytes). |
| **Component Type** | The type of component: *Application; Image; Page; Text* |
| **Component Hierarchical Path** | The hierarchical path of the component. For more information on setting this condition, see "Hierarchical Path Dialog Box" on page 1354. |
| **Component Network Time** | The amount of time from the component's first HTTP request, until receipt of ACK. |
| **Component Server Time** | The amount of time from when the component receives of ACK, until the first buffer is successfully received back from the Web server. |
| **Component Client Time** | The average amount of time that passes while a component request is delayed on the client machine due to browser think time or other client-related delays. |

### User Defined Data Point Graphs

You can apply the following filter conditions to User-Defined Data Point graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Datapoint Name** | The name of the data point. |
| **Datapoint Value** | The value of the data point. |

### System Resources Graphs

You can apply the following filter conditions to System Resource graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **System Resource Name** | The name of the system resource. |
| **System Resource Value** | The value of the system resource. See "Set Dimension Information Dialog Box" on page 1355. |

### Network Monitor Graphs

You can apply the following filter conditions to Network Monitor graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Network Path Name** | The name of the network path. |
| **Network Path Delay** | The delay of the network path. |
| **Network Path Father** | The father of the network path. |
| **Network SubPath Name** | The name of the network subpath. |
| **Network SubPath Delay** | The delay of the network subpath. |
| **Network Full Path** | The full network path. |
| **Network Segment Name** | The name of the network segment. |
| **Network Segment Delay** | The delay of the network segment. |
| **Network Segment Full Path** | The full network segment path. |

### Web Server Resource Graphs

You can apply the following filter conditions to Web Server Resource graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Measurement Name** | The name of the measurement. |
| **Measurement Value** | The measurement value. See "Set Dimension Information Dialog Box" on page 1355. |

### Web Application Server Resource Graphs

You can apply the following filter conditions to Web Application Server Resource graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Resource Name** | The name of the resource. |
| **Resource Value** | The value of the resource. See "Set Dimension Information Dialog Box" on page 1355. |

### Database Server Resource Graphs

You can apply the following filter conditions to Database Server Resource graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Database Resource Name** | The name of the database resource. |
| **Database Resource Value** | The value of the database resource. See "Set Dimension Information Dialog Box" on page 1355. |

### J2EE & .NET Diagnostics Graphs

You can apply the following filter conditions to J2EE & .NET Diagnostics graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Transaction Name** | The name of the Java transaction. |
| **Method Chain of Calls** | The chain of calls for the Java method. |
| **Layer Name** | The name of the layer. |
| **Class Name** | The name of the class. |

| Filter Condition | Filters the graph according to... |
|---|---|
| **Method Name** | The name of the method. |
| **SQL Logical Name** | The SQL logical name for the Java transaction. |
| **Response Time** | The response time of the Java transaction. |
| **Host Name - J2EE/.NET** | The name of the host for the J2EE & .NET transaction. |
| **Application Host Name - (VM)** | The name of the application host for the VM. |
| **Transaction Request** | The request for the transaction. |
| **Transaction Hierarchical Path** | The hierarchical path of the transaction. For more information on setting this condition, see "Hierarchical Path Dialog Box" on page 1354. |

### Application Component Graphs

You can apply the following filter conditions to Application Component graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Component Resource Name** | The resource name of the component. |
| **Component Resource Value** | The value of the component resource. See "Set Dimension Information Dialog Box" on page 1355. |
| **COM+ Interface** | The interface of the COM+ component. |
| **COM+ Response Time** | The response time of the COM+ component. |
| **COM+ Call Count** | The call count of the COM+ component. |
| **COM+ Method** | The method of the COM+ component. |
| **.NET Resource Name** | The resource name of the .NET component. |
| **.NET Value** | The .NET resource value. See "Set Dimension Information Dialog Box" on page 1355. |
| **.NET Class** | The class of the .NET component. |

| Filter Condition | Filters the graph according to... |
|---|---|
| **.NET Response Time** | The response time of the .NET component. |
| **.NET Call Count** | The call count of the .NET component. |
| **.NET Method** | The method of the .NET component. |

### Application Deployment Graphs

You can apply the following filter conditions to Application Deployment graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Citrix Resource Name** | The name of the Citrix resource. |
| **Citrix Resource Value** | The value of the Citrix resource. See "Set Dimension Information Dialog Box" on page 1355. |

### Middleware Performance Graphs

You can apply the following filter conditions to Middleware Performance graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Message Queue Resource Name** | The name of the message queue resource. |
| **Message Queue Resource Value** | The value of the Message Queue resource. See "Set Dimension Information Dialog Box" on page 1355. |

### Infrastructure Resource Graphs

You can apply the following filter conditions to Infrastructure Resource graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **Network Client** | The name of the network client. |
| **Network Client Value** | The value of the network client. See "Set Dimension Information Dialog Box" on page 1355. |

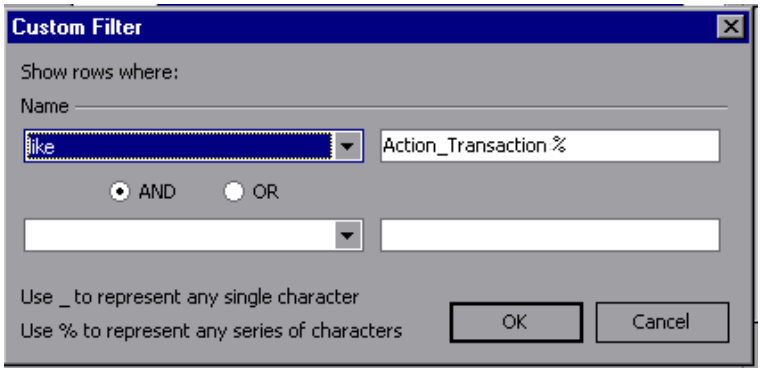
### External Monitor Graphs

You can apply the following filter conditions to External Monitor graphs:

| Filter Condition | Filters the graph according to... |
|---|---|
| **External Monitor Resource Name** | The name of the external monitor resource. |
| **External Monitor Resource Value** | The value of the external monitor resource. See "Set Dimension Information Dialog Box" on page 1355. |

## Custom Filter Dialog Box

This dialog box enables you to customize your filter criteria.




| | |
|---|---|
| **To access** | Do the following:<br>1. In a **Legend** window, click a column header.<br>2. Expand the down arrow and choose **(Custom)**. |
| **Tip** | You can use wildcards:<br>• Use **_** to represent a single character.<br>• Use **%** to represent a series of characters. |
| **See also** | "Legend Window" on page 1317 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **<First Evaluator Expression>** | A drop-down list of evaluation expressions such as **equals**, **is greater than**, **like**, and so forth, followed by a value. |
| **Operator** | The logical operator by which to add a second expression: **AND** or **OR**. |
| **<Second Evaluator Expression>** | A drop-down list of evaluation expressions such as **equals**, **is greater than**, **like**, and so forth, followed by a value. |

**Example:**

The image above shows how to filter the data for transactions that begin with the phrase "Action_Transaction", using **Like** and **Action_Transaction%**.

After you save a customization for one of the metrics, Analysis displays it in the lower section of the Legend window.

## Filter Dialog Boxes

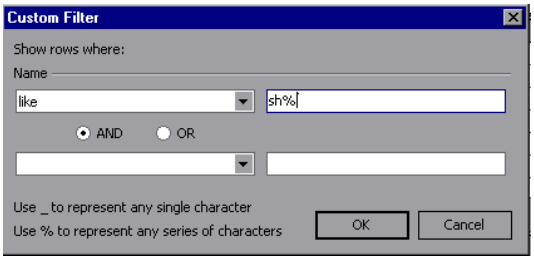
The filter dialog boxes (Graph Settings, Global Filter, and Analysis Summary Filter) enable you to filter the data that is displayed in the graph or report.

When adding a graph, the filter and sort button is displayed which enables you to filter and sort data before the graph is displayed.

| | |
|---|---|
| **To access** | **Use one of the following:**<br>• **View > Set Filter/Group By** or click <br>• **File > Set Global Filter** or click <br>• **View > Summary Filter** or click |
| **Note** | Some fields below only appear in some of the filter dialog boxes. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Filter Condition** | Select criteria and values for each filter condition that you want to employ. The applicable filter conditions are displayed for each graph. For details on each graphs filter conditions, see the chapter on the relevant graph. |
| **Criteria** | Select "=" (equals) or "<>" (does not equal). |

| UI Element | Description |
|---|---|
| **Values** | The filter conditions are grouped into three value types (discrete, continuous, and time-based).<br><br>A discrete value is a distinct integer (whole number) or string value such as Transaction Name or Vuser ID. Select the check box(es) of the value(s) that you want to include in your filter. You can also customize your filter by entering wild cards to depict any single character or any series of characters.<br><br>  <br><br>• A continuous value is a variable dimension that can take any value within the minimum and maximum range limits, such as Transaction Response Time. You set the dimension information for each measurement in the "Set Dimension Information Dialog Box" on page 1355.<br>• A time-based value is a value that is based on time relative to the start of the load test scenario. Scenario Elapsed Time is the only condition that uses time-based values. You specify time-based values in the "Scenario Elapsed Time Dialog Box" on page 1355.<br><br>For some filter conditions, one of the following dialog boxes opens to enable you to specify additional filtering details:<br><br>• "Set Dimension Information Dialog Box" on page 1355<br>• "Vuser ID Dialog Box" on page 1356<br>• "Scenario Elapsed Time Dialog Box" on page 1355<br>• "Hierarchical Path Dialog Box" on page 1354: Enables you to display the hierarchical path of a transaction or component, or a method chain of calls. |
| **Transaction Percentile** | The Summary Report contains a percentile column showing the response time of 90% of transactions (90% of transactions that fall within this amount of time). To change the value of the default 90 percentile, enter a new figure in the **Transaction Percentile** box. |
| **Set Default** | Displays the default criteria and values for each filter condition. |
| **Clear All** | Deletes all of the information you entered in the dialog box. |

| UI Element | Description |
|---|---|
| **Group By settings** | Use these settings to sort the graph display by grouping the data. You can group the data by:<br>• **Available groups**. Select the group by which you want to the sort the results, and click the right arrow.<br>• **Selected groups**. Displays a list of all the selected groups by which the results will be sorted. To remove a value, select it and click the left arrow. |
| **Clear all graph filters and the Summary Filter, before applying the Global Filter settings** | When this check box is selected for the Global Filter, Analysis clears all user-defined settings in the graph filters (including Group By settings) and in the Analysis Summary Filter, and then applies the current Global Filter conditions.<br>**Note:** The Transaction Percentile setting in the Analysis Summary Filter is not affected by the Global Filter. |

## Filter Builder Dialog Box

The Filter Builder dialog box enable you to design, add, and edit filters for your graph.

| | |
|---|---|
| **To access** | 1. In the Legend pane, expand the down arrow in a column header.<br>2. Select **Custom** to open the Custom Filter dialog box. Provide filter details and click OK.<br>3. Click **Customize** in the filter entry in the lower part of the **Legend** pane. |
| **See also** | "Custom Filter Dialog Box" on page 1350 |

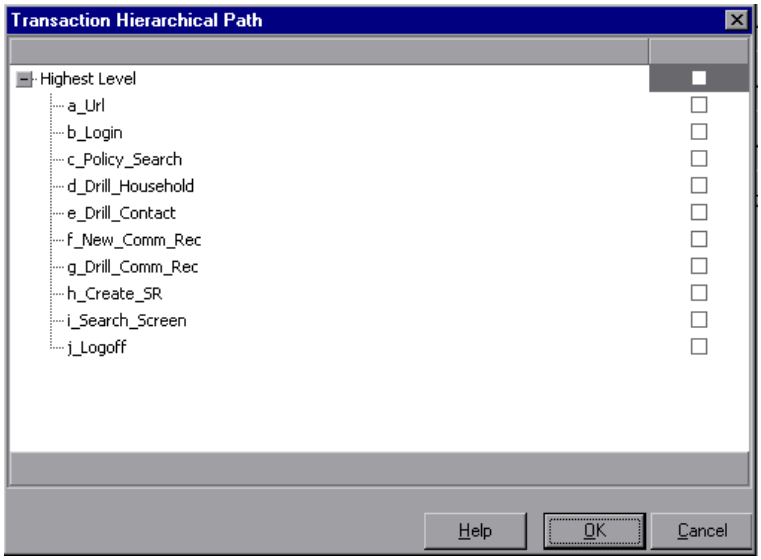
User interface elements are described below:

| UI Element | Description |
|---|---|
| **Filter button** | Opens a menu with the following options:<br>• **Add Condition.** Add another condition for the current filter.<br>• **Add Group.** Adds a second condition, joined by a logical operator AND or OR, to the last condition in the list.<br>• **Clear All.** Removes all of the conditions in the window. |

| UI Element | Description |
|---|---|
| ... | Opens a menu with the following options:<br>• **Add Condition.** Add another condition for the current filter.<br>• **Add Group.** Adds a second condition, joined by a logical operator AND or OR, to the selected condition in the list.<br>• **Remove Row.** Removes the selected condition. |
| **Open** | Opens an **.flt** file saved from a previous session. |
| **Save as** | Saves all of the conditions to an **.flt** file. |

## Hierarchical Path Dialog Box

This dialog box enables you to display the hierarchical path of a transaction or component, or a method chain of calls.



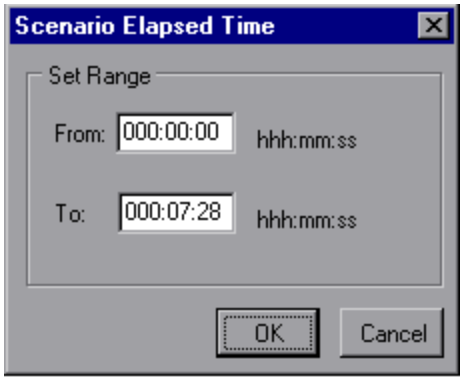

| | |
|---|---|
| **To access** | **View** menu **>** **Set Filter/Group by** > **Filter condition** pane > **Transaction, Component Hierarchical Path or a method chain of calls** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Transaction, Component Hierarchical Path or a method chain of calls** | Select the box for the path where you want to start to see results. Only the selected path and its immediate sub-nodes will be displayed. |

## Scenario Elapsed Time Dialog Box

This dialog box enables you to specify the start and end time range for the graph's x-axis.



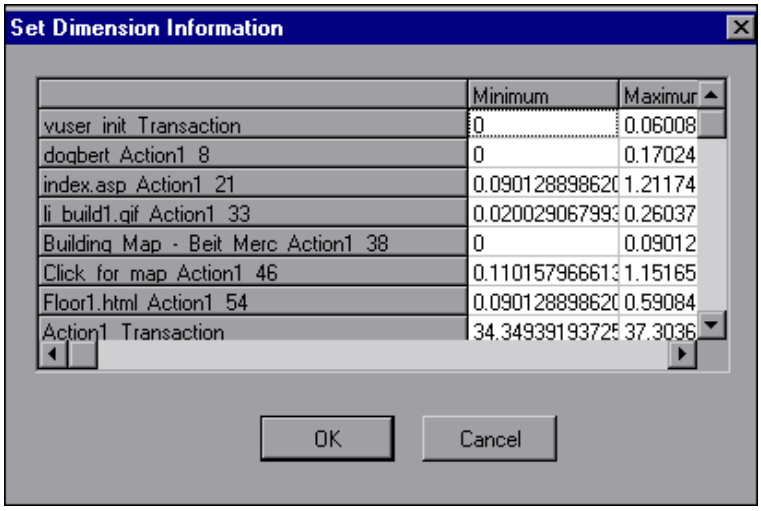

| | |
|---|---|
| **To access** | **View** menu **>** **Set Filter/Group by** > **Filter condition** pane > **Scenario Elapsed Time** |
| **Note** | The time is relative to the start of the scenario. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **From** | Specify a start value for the desired range. |
| **To** | Specify an end value for the desired range. |

## Set Dimension Information Dialog Box

This dialog box enables you to set the dimension information for each measurement (transaction, number of released Vusers, resource) in the result set. You specify the minimum and maximum values for each measurement you want in the analysis. By default, the full range of values for each measurement is displayed.



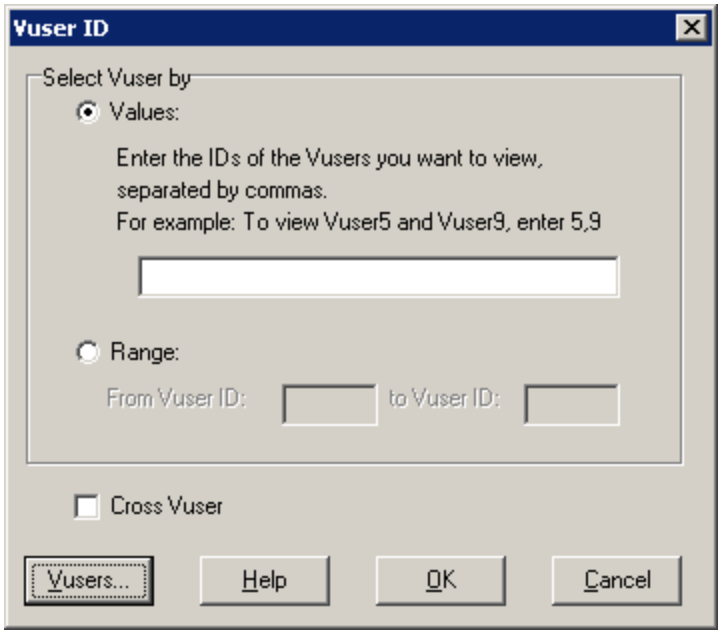

| | |
|---|---|
| **To access** | You can open this dialog box from the following locations:<br>• **Transaction** graphs **>** **View** menu **>** **Set Filter/Group by** > **Filter condition** pane > **Transaction Response Time**<br>• **Vusers** graph > **Rendezvous** graph > **View** menu **>** **Set Filter/Group by** > **Filter condition** pane > **Number of Released Vusers**<br>• All graphs that measure resources (**Web Server**, **Database Server**, and so on) > **View** menu **>** **Set Filter/Group by** > **Filter condition** pane > **Resource Value** |
| **Note** | If you are specifying the start and end time for a transaction (in minutes:seconds format), the time is relative to the beginning of the load test scenario. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Minimum** | Specify a minimum value for the measurement. |
| **Maximum** | Specify a maximum value for the measurement. |

## Vuser ID Dialog Box

This dialog box opens to enable the entering of additional filter information for the Vuser ID filter condition.




| | |
|---|---|
| **To access** | **View menu > Set Filter/Group by > Filter condition pane > Vuser ID** |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Value** | Enter the Vuser IDs of the Vusers you want the graph(s) to display, separated by commas. |
| **Range** | Specify the beginning and end of the desired range of Vusers you want the graph(s) to display. |
| **Cross Vuser** | Cross Vuser transactions are transactions that start with one Vuser and end with a different Vuser, such as sending an email.<br>Selecting this option places the value "CrossVuser" in the Vuser ID filter.<br>**Default:** not selected.<br>Note: Only transaction graphs have Cross Vuser data. |
| **Vusers** | Displays the existing Vuser IDs from which you can choose. |

## Cross Result and Merged Graphs

Comparing results is essential for determining bottlenecks and problems. You use Cross Result graphs to compare the results of multiple load test scenario runs. You create Merged graphs to compare results from different graphs within the same scenario run.

## Cross Result Graphs Overview

Cross Result graphs are useful for:

- Benchmarking hardware
- Testing software versions
- Determining system capacity

If you want to benchmark two hardware configurations, you run the same load test scenario with both configurations and compare the transaction response times using a single Cross Result graph.

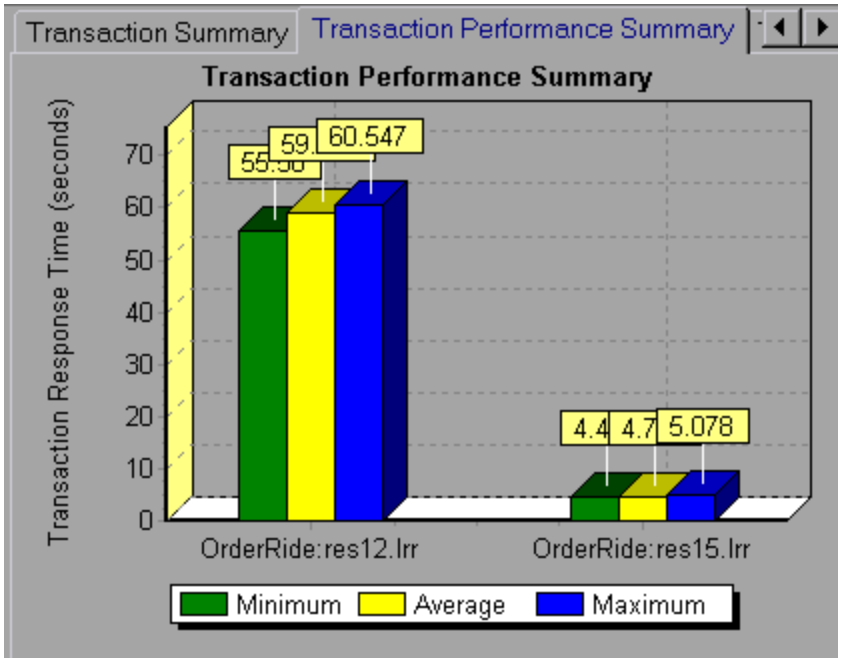
Suppose that your vendor claims that a new software version is optimized to run quicker than a previous version. You can verify this claim by running the same scenario on both versions of the software, and comparing the scenario results.

You can also use Cross Result graphs to determine your system's capacity. You run scenarios using different numbers of Vusers running the same script. By analyzing Cross Result graphs, you can determine the number of users that cause unacceptable response times.

In the following example, two scenario runs are compared by crossing their results, **res12**, and **res15**. The same script was executed twice—first with 100 Vusers and then with 50 Vusers.

In the first run, the average transaction time was approximately 59 seconds. In the second run, the average time was 4.7 seconds. It is apparent that the system works much slower with a greater load.

The Cross Result graphs have an additional filter and group by category: **Result Name**. The example below is filtered to the **OrderRide** transaction for results **res12**, and **res15**, grouped by **Result Name**.



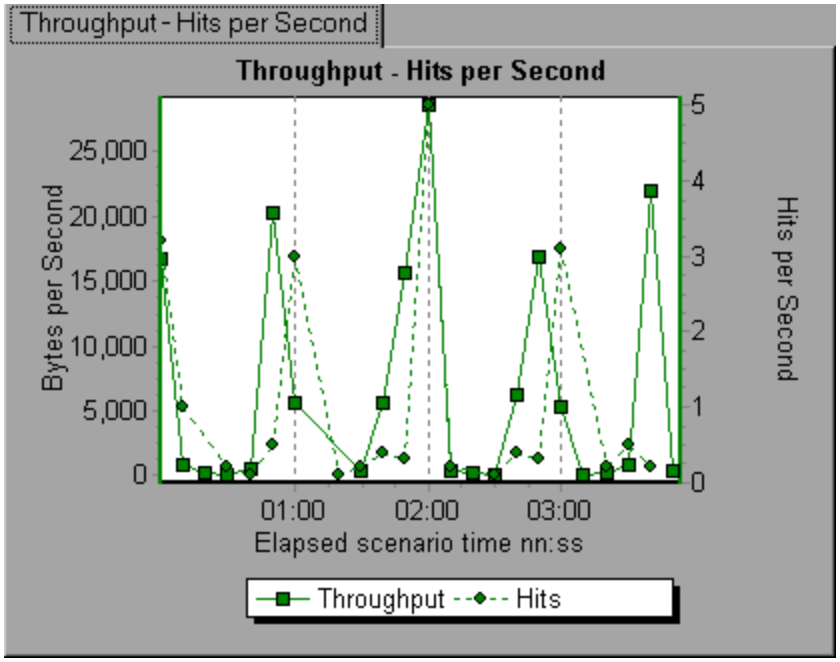
## Merging Types Overview

Analysis provides three types of merging.

### Overlay

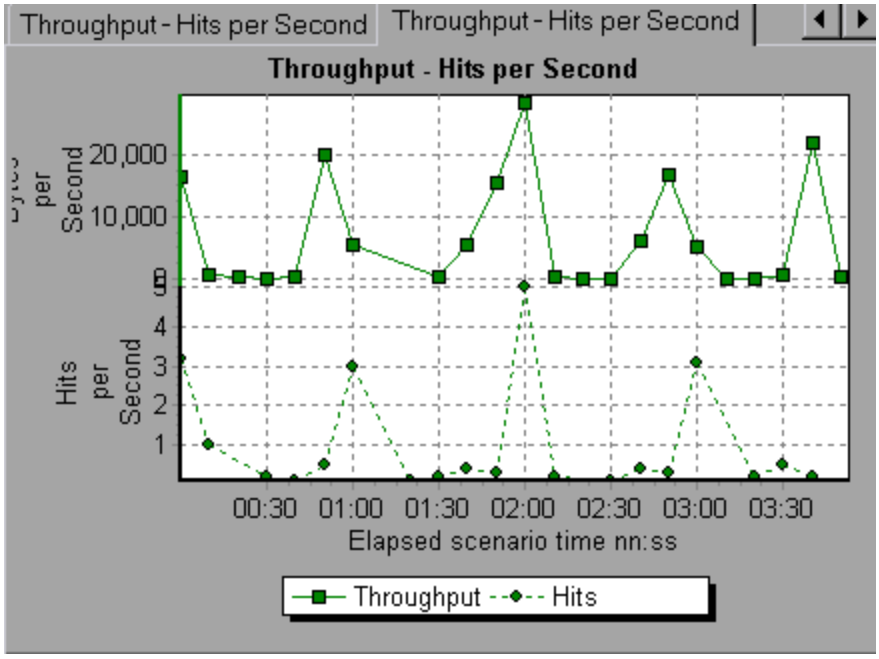
Superimpose the contents of two graphs that share a common x- axis. The left y-axis on the merged graph shows the current graph's values. The right y-axis shows the values of the graph that was merged. There is no limit to the number of graphs that you can overlay. When you overlay two graphs, the y-axis for each graph is displayed separately to the right and left of the graph. When you overlay more than two graphs, Analysis displays a single y-axis, scaling the different measurements accordingly.

In the following example, the Throughput and Hits per Second graph are overlaid with one another.




### Tile

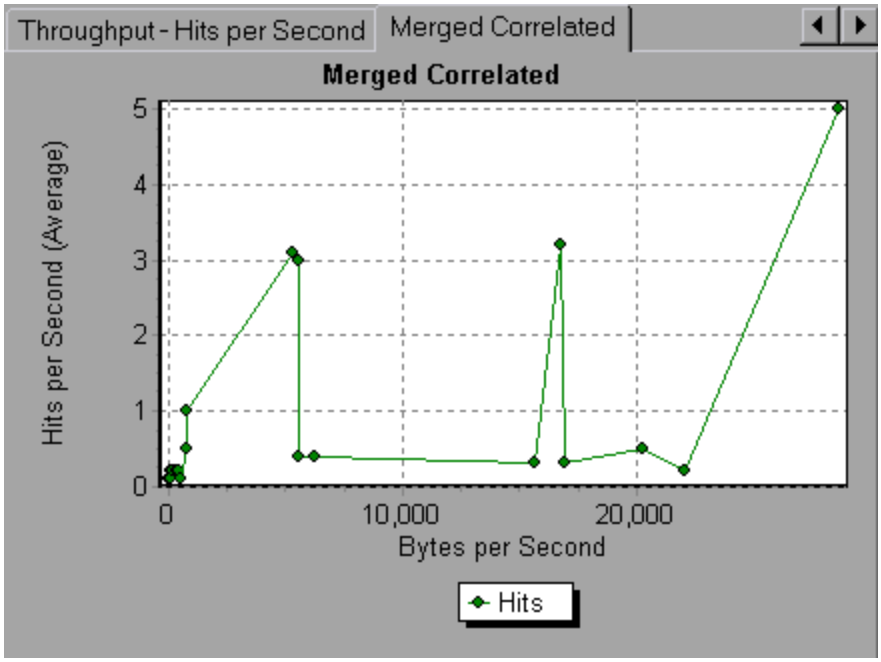
View contents of two graphs that share a common x-axis in a tiled layout, one above the other. In the following example the Throughput and Hits per Second graph are tiled one above the other.




## Correlate

Plot the y-axis of two graphs against each other. The active graph's y-axis becomes the x-axis of the merged graph. The y-axis of the graph that was merged, becomes the merged graph's y-axis.

In the following example, the Throughput and Hits per Second graph are correlated with one another. The x-axis displays the bytes per second (the Throughput measurement) and the y-axis shows the average hits per second.



## Generating Cross Results Graphs

This task describes how to create a Cross Result graph for two or more result sets. The Cross Result dialog box enables you to compare the results of multiple load test scenario runs.

1. Choose **File** > **Cross With Result**. The Cross Results dialog box opens.
2. Click **Add** to add an additional result set to the **Result List**. The Select Result Files for Cross Results dialog box opens.
3. Locate a results folder and select its result file (**.lrr**). Click **OK**. The scenario is added to the Result List.
4. Repeat steps 2 and 3 until all the results you want to compare are in the Result List.
5. When you generate a Cross Result graph, by default it is saved as a new Analysis session. To save it in an existing session, clear the **Create New Analysis Session for Cross Result** box.
6. Click **OK**. Analysis processes the result data and asks for a confirmation to open the default graphs.

> **Note:** When generating a Cross Results Session, verify that the transaction names do not contain a <_> or <@> symbol. This will cause errors to occur when attempting to open the Cross Results graphs.

After you generate a Cross Result graph, you can filter it to display specific scenarios and transactions. You can also manipulate the graph by changing the granularity, zoom, and scale.

You can view a summary report for the Cross Result graph.

## Generating Merged Graphs

This task describes how to merge the results of two graphs from the same load test scenario into a single graph. The merging allows you to compare several different measurements at once. For example, you can make a merged graph to display the network delay and number of running Vusers, as a function of the elapsed time.

You can merge all graphs with a common x-axis.

1. Select a graph in the Session Explorer or select its tab to make it active.
2. Choose **View > Merge Graphs** or click **Merge Graphs**. The Merge Graphs dialog box opens and displays the name of the active graph.
3. Select a graph with which you want to merge your active graph. Only the graphs with a common x-axis to the active graph are available.
4. Select the merge type and a title for the merged graph. By default, Analysis combines the titles of the two graphs being merged. For more information, see "Merge Graphs Dialog Box" on the next page.
5. Click **OK**.
6. Filter the graph just as you would filter any ordinary graph.

### Merge Graphs Dialog Box

This dialog box enables you to merge two graphs into a single graph.

| | |
|---|---|
| **To access** | **View > Merge Graphs** |
| **Important information** | In order to merge graphs, the graphs' x-axes must be the same measurement. For example, you can merge Web Throughput and Hits per Second graphs, because their x-axes are Scenario Elapsed Time. |
| **See also** | "Merging Types Overview" on page 1359 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Select Graph to merge with** | The drop-down list shows all of the open graphs that share a common x-axis measurement with the current graph. Select one of the graphs in the list. |
| **Select type of merge** | • **Overlay**. View contents of two graphs that share a common x-axis. The left y-axis on the merged graph shows the current graph's values. The right y-axis shows the values of the graph that was merged with the current graph.<br>• **Tile**. View contents of two graphs that share a common x-axis in a tiled layout, one above the other.<br>• **Correlate**. Plot the y-axes of two graphs against each other. The active graph's y-axis becomes the x-axis of the merged graph. The y-axis of the graph that was merged, becomes the merged graph's y-axis. |
| **Title of Merged Graph** | Enter a title for the merged graph. This title will appear in the Session Explorer (**Windows > Session Explorer**). |

# Working with Application Lifecycle Management

## Managing Results Using ALM - Overview

Analysis works together with Application Lifecycle Management (ALM). ALM provides an efficient method for storing and retrieving scenario and analysis results. You can store results in an ALM project and organize them into unique groups.

In order for the Analysis to access an ALM project, you must connect it to the Web server on which ALM is installed. You can connect to either a local or remote Web server.

When working against an ALM server with Performance Center, the ALM integration has several additional capabilities, such as the ability to save the Analysis session to a new location, and upload a report from the file system to ALM. For details, see "How to Work with Results in ALM - With Performance Center" on the next page.

For more information on working with ALM, see the *Application Lifecycle Management User Guide*.

## How to Connect to ALM from Analysis

To store and retrieve Analysis results from ALM, you need to connect to an ALM project. You can connect or disconnect from an ALM project at any time during the testing process.

You can connect to one version of ALM from Analysis and a different version from your browser. For more information, see the **Important Information** section in "ALM Connection Dialog Box" on page 1368.

To connect to ALM:

1. Select **Tools** > **ALM Connection**. The ALM Connection dialog box opens.
2. Enter the required information in the ALM Connection dialog box, as described in "ALM Connection Dialog Box" on page 1368.
3. To disconnect from ALM, click **Disconnect**.

**Note:**

- There is no explicit option in the Analysis user interface for enabling CAC mode (as in VuGen). Analysis automatically enables CAC mode if the ALM server machine supports it.
- If you have a slow connection to ALM, this may be because the Analysis machine does not have access to the WAN, and so cannot verify the digital signature of the relevant files.

  **Resolution:** Download and import the certificates from https://support.globalsign.com/customer/portal/articles/1426602-globalsign-root-certificates using certmgr MMC until the issue is resolved (you might not need to import all the certificates). This information is also available in a knowledge base article.

## How to Work with Results in ALM - Without Performance Center

The following steps describe the workflow for working with results saved in an ALM project, whose server does not have a Performance Center installation:

1. Connect to ALM

   Open a connection to the ALM server and project that contains the LoadRunner result or Analysis session files. For task details, see "How to Connect to ALM from Analysis" above.
2. Open an existing Analysis session file - optional

a. Select **File > Open**.

b. In the left pane select a script.

c. In the right pane, select the results for which the Analysis session file was created.

d. Click **OK**.

3. Create a new Analysis session file from the raw data - optional

   This procedure describes how to create a new Analysis session file on the ALM server, from the raw results file. If an Analysis session file already exists for the raw data, you can choose to overwrite the existing file.

   a. Select **File > New**.

   b. In the left pane select a script.

   c. In the right pane, select the results you want to analyze.

   d. Click **OK**.

4. Save the LoadRunner results file

   When you are finished analyzing your results and creating reports or graphs, save the changes. Select **File > Save**. The Analysis session file is in the ALM project.

> **Note:** When working with ALM without Performance Center, **Save As** is not supported—you cannot save the Analysis session file to another location.

### See also:

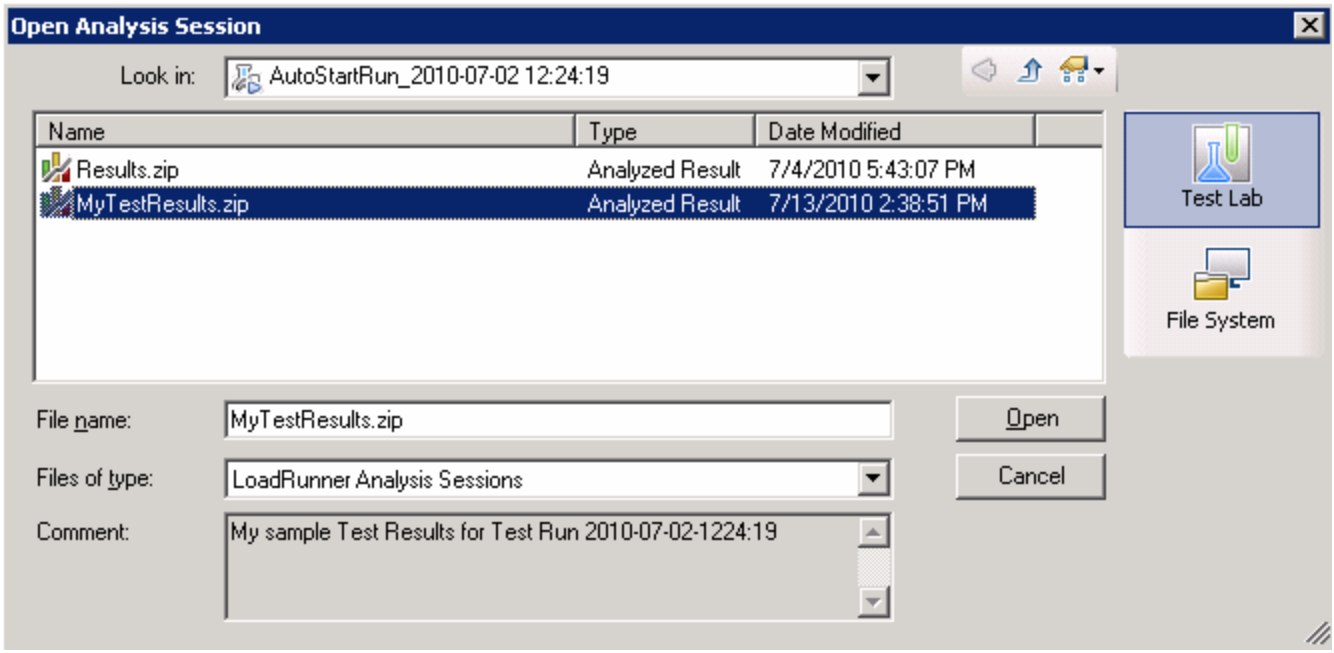
- "How to Work with Results in ALM - With Performance Center" below.

## How to Work with Results in ALM - With Performance Center

ALM servers with Performance Center, allow you to perform the following operations.

### Open an existing Analysis Session file

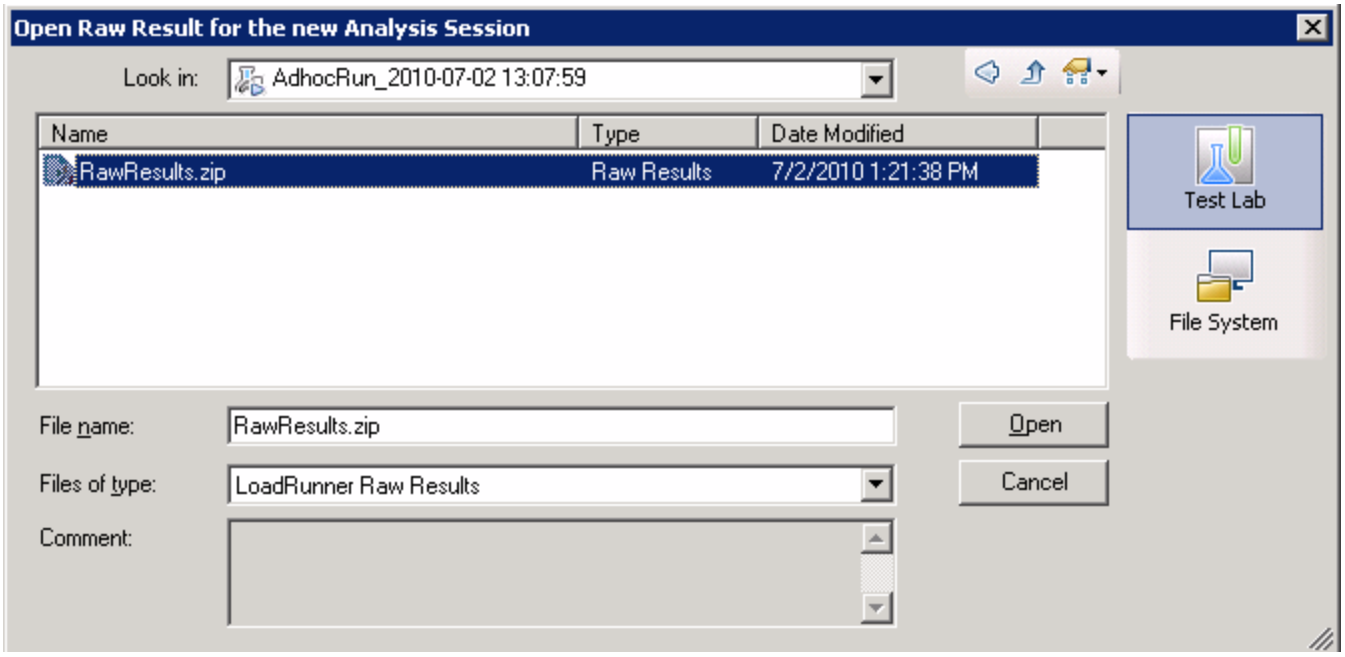
1. Select **Tools > ALM Connection** and make sure your connection to ALM is open.
2. Select **File > Open**.
3. Drill down to the Run level within the Test Plan module, and select an individual run.
4. Select a zip file containing the Analysis session file.




5. Click **Open**.

## Open raw data and create a new Analysis session

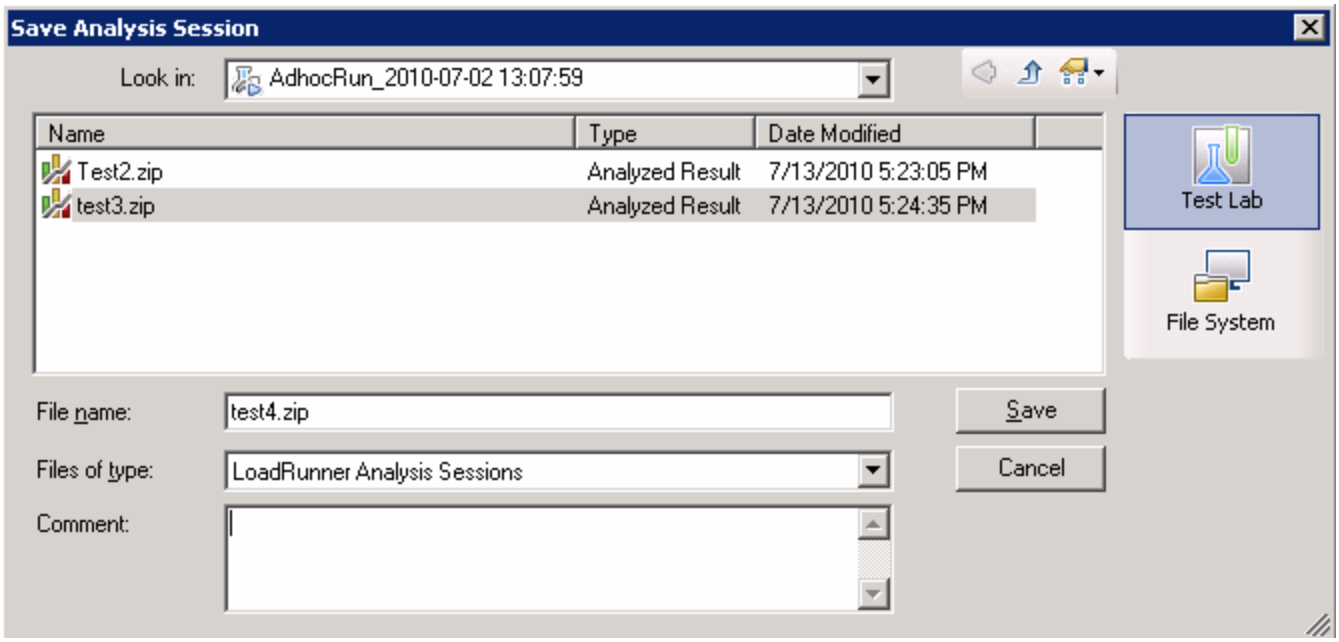
1. Select **Tools > ALM Connection** and make sure your connection to ALM is open.
2. To create a new Analysis session file from the raw data, select **File > New**.
3. Drill down to the Run level within the Test Plan module, and select an individual run.
4. Select a zip file containing the run's raw data.




5. Click **Open**.

## Save the changes to the Analysis session file

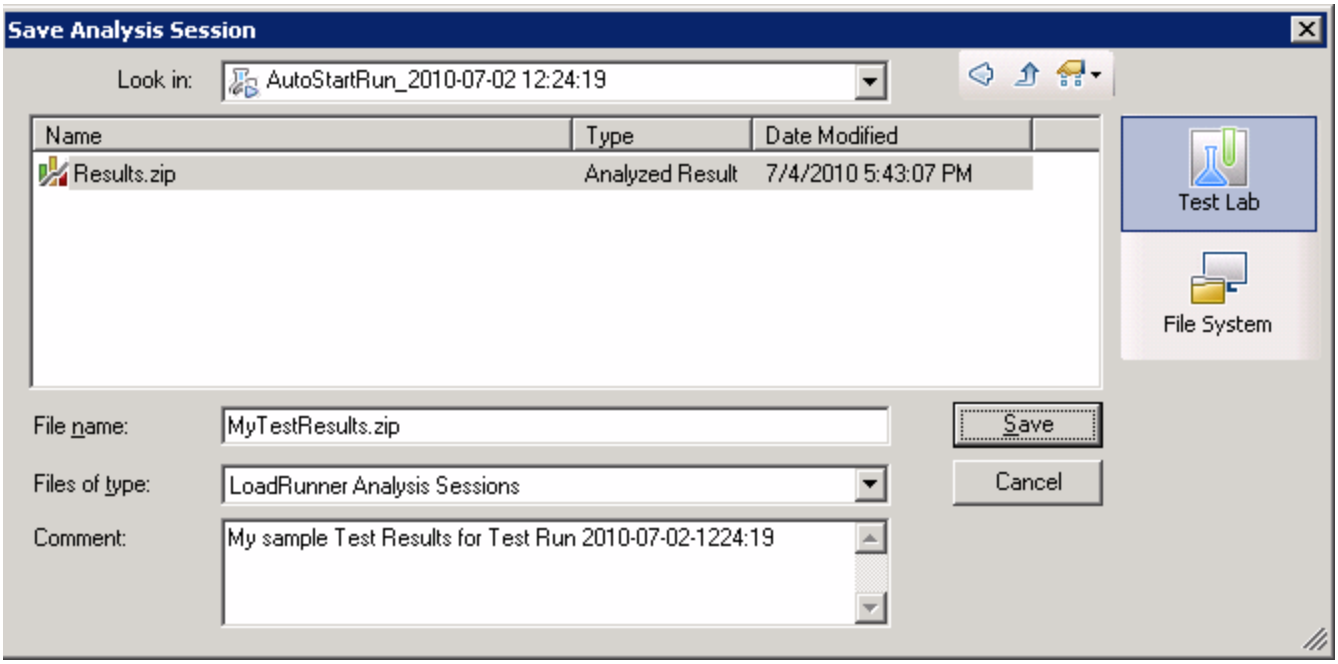
1. Complete your changes to the Analysis results.
2. Select **Tools > ALM Connection** and make sure your connection to ALM is open.
3. Select **File > Save**.
4. To save an Analysis session that was opened from the file system, click the **Test Lab** module button.
5. Drill down to the Run level within the Test Plan module, and specify a name for the zip file.




6. Provide a comment about the Analysis session (optional).
7. Click **Save**.

## Save the Analysis session file to a new ALM location

1. Select **Tools > ALM Connection** and make sure your connection to ALM is open.
2. Open an Analysis session file from the file system, or from ALM as described above.
3. Select **File > Save as**.
4. Drill down to the Run level within the Test Plan module, and select an individual run.
5. Specify a name for the Analysis session zip file. The name *Results* is reserved.




6. Provide a comment about the Analysis session (optional).
7. Click **Save**.

### Integration Methods - TestPlan or TestLab

Analysis uses different integration methods for ALM projects with Performance Center extensions, depending on how it was invoked:

- Through the Web-interface or from Controller—**TestPlan** integration is used.
- Through a manual launch, connected to a project through the ALM Connection dialog box—**TestLab** integration is used.

### See also:

- "How to Work with Results in ALM - Without Performance Center" on page 1363

## How to Upload a Report to ALM with Performance Center

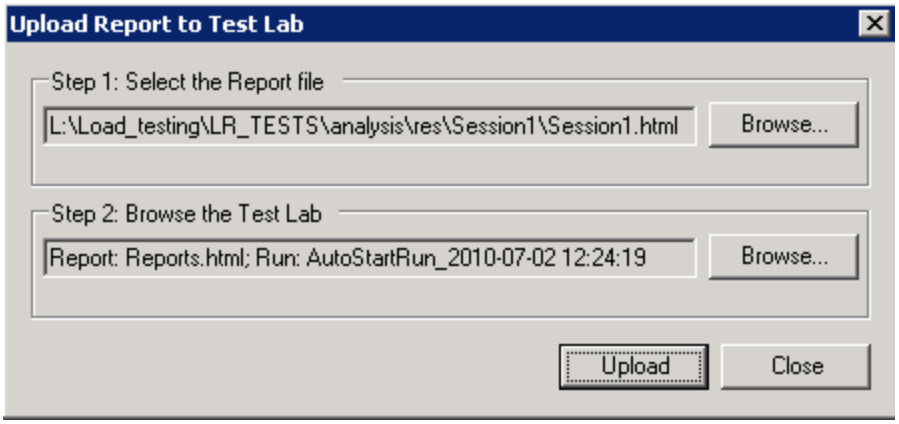
The following steps describe how to upload a report from the file system to an ALM's Test Lab module. This feature is only available for ALM installation with Performance Center.

1. Connect to ALM

   Open a connection to the ALM server and project that contains the LoadRunner result or Analysis session files. For task details, see "How to Connect to ALM from Analysis" on page 1363.

2. Open the Upload dialog box

   Select **Tools > Upload Report to Test Lab**.




3. Select a report

   Click **Browse** in the **Step 1** section**.** The **Select the Report file** dialog box opens. Select an HTML or XML file from the file system. Click **Open**.

4. Select a location on ALM

   Click **Browse** in the **Step 2** section**.** The **Select Location for the Report** dialog box opens. Navigate to a Run level in the Test Lab module. Specify a name for the report and include any relevant comments. Click **OK**.

5. Begin the upload

   Click **Upload**. The upload begins.

## ALM Connection Dialog Box

This dialog box enables you to connect to an ALM project.

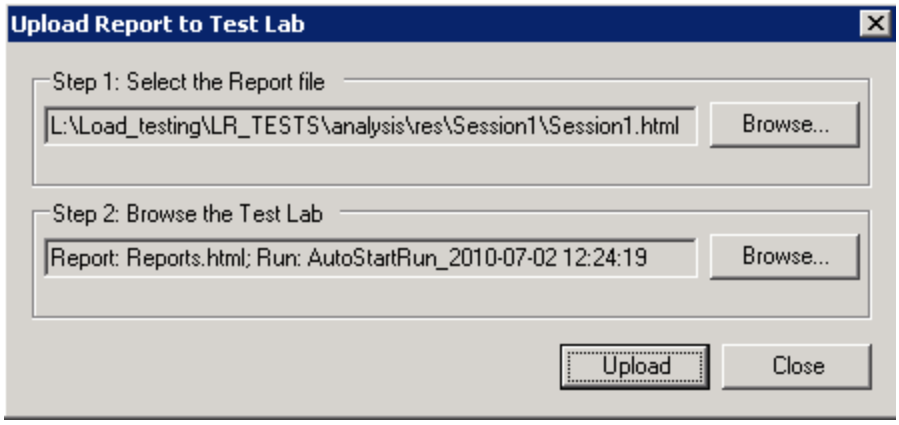
| **To access** | **Tools > ALM Connection** |
|---|---|
| **Important information** | You can connect to one version of ALM from LoadRunner and a different version of ALM from your browser.<br><br>You can only connect to different versions of ALM if one of the versions is ALM 11.00 or higher.<br><br>**Note:** Before you connect to ALM through the LoadRunner interface, it is recommended that you first connect to the ALM server through your browser. This automatically downloads the ALM client files to your computer. |
| **Relevant tasks** | "How to Connect to ALM from Analysis" on page 1363 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Step 1: Connect to Server** | • **Server URL.** The URL of the server on which ALM is installed. The URL must be in the following form **http://<server_name:port>/qcbin**.<br>• **Reconnect to server on startup.** Automatically reconnect to the server every time you start LoadRunner.<br>• Connect / Disconnect. Connects to the server specified in the **Server URL** box. Only one button is visible at a time, depending on your connection status. |
| **Step 2: Authenticate User Information** | • **User Name.** Your ALM project user name.<br>• **Password.** Your ALM project password.<br>• **Authenticate on startup.** Authenticates your user information automatically, the next time you open the application. This option is available only if you selected **Reconnect to server on startup** above.<br>• Authenticate. Authenticates your user information against the ALM server.<br>After your user information has been authenticated, the fields in the Authenticate user information area are displayed in read-only format. The Authenticate button changes to Change User.<br>You can log in to the same ALM server using a different user name by clicking **Change User**, entering a new user name and password, and then clicking **Authenticate** again. |
| **Step 3: Login to Project** | • **Domain.** The domain that contains the ALM project. Only those domains containing projects to which you have permission to connect to are displayed.<br>• **Project.** Enter the ALM project name or select a project from the list. Only those projects that you have permission to connect to are displayed.<br>• **Login to project on startup.** This option is only enabled when the **Authenticate on startup** check box is selected.<br>• Login / Logout. Logs into and out of the ALM project. |

## Upload Report to Test Lab Dialog Box

This dialog box enables you to upload an Analysis report to an ALM project's Test Lab module.

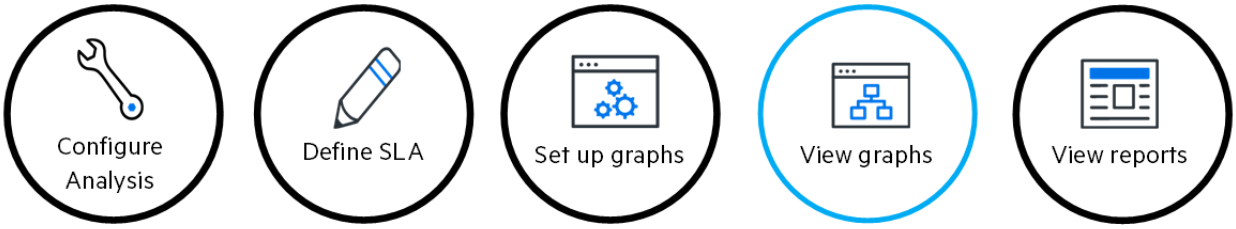



| | |
|---|---|
| **To access** | **Reports > Upload Report to Test Lab** |

User interface elements are described below:

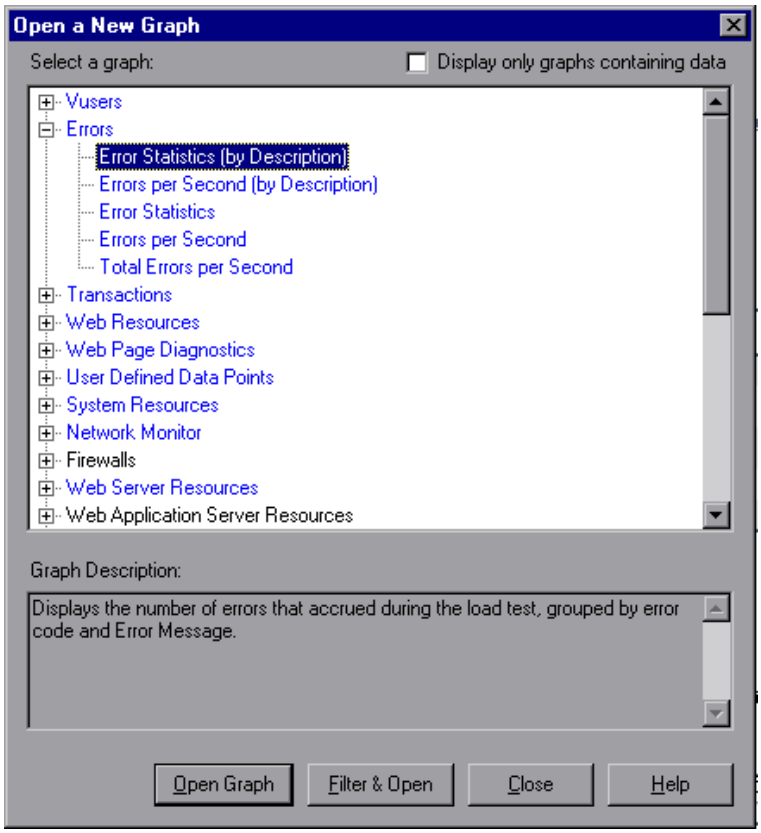
| UI Element | Description |
|---|---|
| **Step 1: Select the report file** | Allows you to select an Analysis report from the file system. You can select an HTML report, or Rich report in XML format. |
| **Step 2: Browse the test lab** | Allows you to select an location within the Test Lab module, for the report.<br>**Note:** You must drill down to the level of a Run within the Test Lab module. |
| **Upload** | Begins the uploading of the report. If the uploading succeeds, the Analysis issues a message. |

# Analysis Graphs



# Open a New Graph Dialog Box

The Open a New Graph dialog box enables you to select the graph type to activate in the main Analysis window.




| | |
|---|---|
| **To access** | **Session Explorer > Graphs >** |

User interface elements are described below:

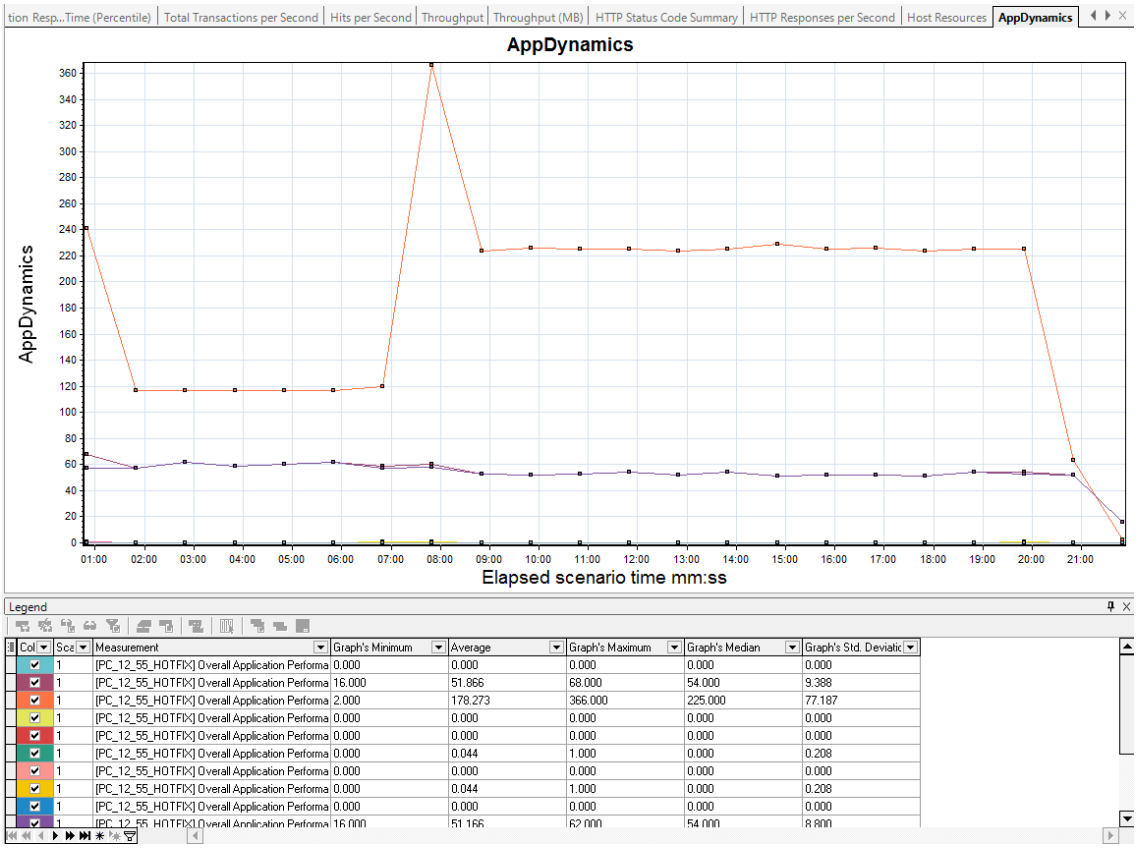
| UI Element | Description |
|---|---|
| **Select a graph** | Displays list of graph types. |
| **Display only graphs containing data** | If checked, only graphs that contain data are listed (in blue) in the Select a graph area. |
| **Graph Description** | Displays detailed information about the selected graph. |

| UI Element | Description |
|---|---|
| Open Graph | Analysis generates the selected graph and adds it to the Session Explorer. |
| Filter & Open | Opens the graphs Graph Settings dialog box. For details, see "Filter Dialog Boxes" on page 1351. This option enables you to apply filter conditions on the selected graph before the graph is displayed. |

# AppDynamics Graph

> **Note:** Available for Performance Center only.

The AppDynamics graph displays metrics about the availability and performance of software applications and the impact on user experience measured by the AppDynamics Platform, an application performance monitoring tool.




For details about the AppDynamics monitor, see the Performance Center help.

For more information about AppDynamics, see the AppDynamics documentation.

# Application Component Graphs

**Microsoft COM+ performance graphs** provide you with performance information for COM+ interfaces and methods.

To obtain data for these graphs, you need to activate the various Microsoft COM+ performance monitors before running the load test scenario.

When you set up the Microsoft COM+ performance online monitors, you indicate which statistics and measurements to monitor.

The **.NET CLR performance graphs** provide you with performance information for .NET classes and methods. To obtain data for these graphs, you must activate the .NET CLR performance monitor before running the load test scenario run.

Displayed measurements are specified using the .NET monitor.

For more information, see the section on online monitors in the LoadRunner Controller documentation.

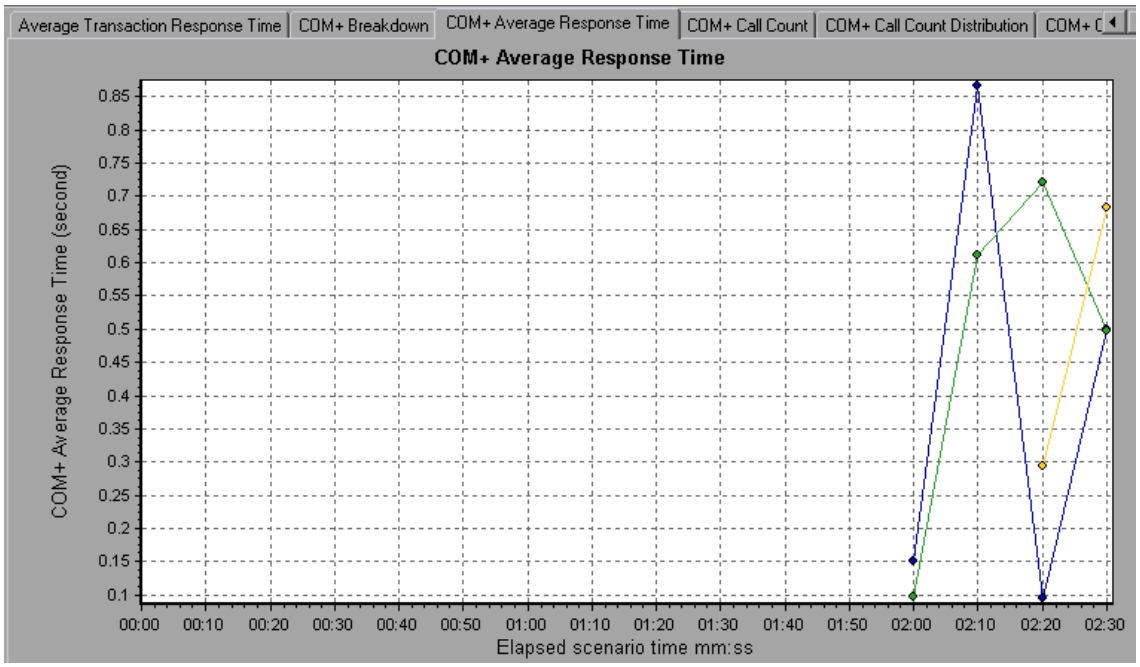
## COM+ Average Response Time Graph

This graph specifies the average time COM+ interfaces or methods take to perform during the load test scenario.

| | |
|---|---|
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | Average response time of a COM+ interface or method. |

| | |
|---|---|
| **Breakdown options** | Each interface or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the interfaces by color:<br><br>This legend shows that the blue colored line belongs to the COM+ interface **_ConstTime.** Looking at the graph above, we see that this interface has higher response times than all other COM+ interfaces. At 2:10 minutes into the scenario, it records an average response time of 0.87 seconds.<br><br>**Note:** The 0.87 second data point is an average, taken from all data points recorded within a 10 second interval (the default granularity). You can change the length of this sample interval.<br><br>Viewing CON+ Methods<br><br>The table initially displays COM+ interfaces, but you can also view the list of COM+ methods by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific interface line in the graph, select the interface row in the legend. |
| **See also** | "Application Component Graphs" on the previous page |

| Color | Scale | Measurement | Minimum | Average | Maximum |
|---|---|---|---|---|---|
| ☑ | 1 | ContrRnd\COMPlusServer.ConstTime\IDispatch | 0.096 | 0.499 | 1.501 |
| ☑ | 1 | ContrRnd\COMPlusServer.ConstTime_ConstTime | 0 | 0.5 | 1.502 |
| ☑ | 1 | ContrRnd\COMPlusServer.RandomTime_RandomTime | 0.058 | 0.391 | 0.747 |

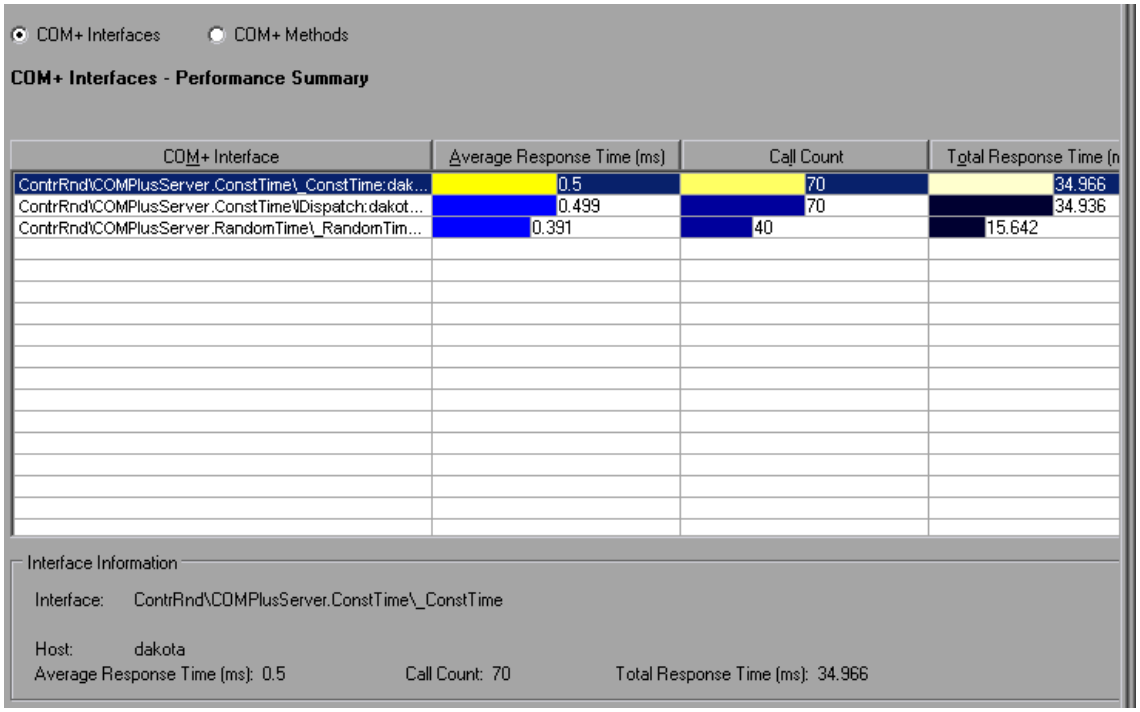


# COM+ Breakdown Graph

This graph summarizes fundamental result data about COM+ interfaces or methods and presents it in table format.

| | |
|---|---|
| **Purpose** | Using the COM+ Breakdown table, you can identify the COM+ interfaces or methods which consume the most time during the test. The table can be sorted by column, and the data can be viewed either by COM+ interface or COM+ method. |
| **Breakdown options** | **Average Response Time**<br>The **Average Response Time** column shows how long, on average, an interface or method takes to perform. The graphical representation of this column is the "COM+ Average Response Time Graph" on page 1373.<br>**Call Count**<br>The next column, **Call Count**, specifies the number of times the interface or method was invoked. The graphical representation of this column is the "COM+ Average Response Time Graph" on page 1373.<br>**Total Response Time**<br>The final column, **Total Response Time**, specifies how much time was spent overall on the interface or method. It is calculated by multiplying the first two data columns together. The graphical representation of this column is the "COM+ Average Response Time Graph" on page 1373.<br>The graphical representations of each of these columns are the "COM+ Average Response Time Graph" on page 1373, the "COM+ Call Count Distribution Graph" on the next page and the "COM+ Total Operation Time Distribution Graph" on page 1380<br>Interfaces are listed in the **COM+ Interface** column in the form **Interface:Host**. In the table above, the **_ConstTime** interface took an average of .5 seconds to execute and was called 70 times. Overall, this interface took 34.966 seconds to execute. |

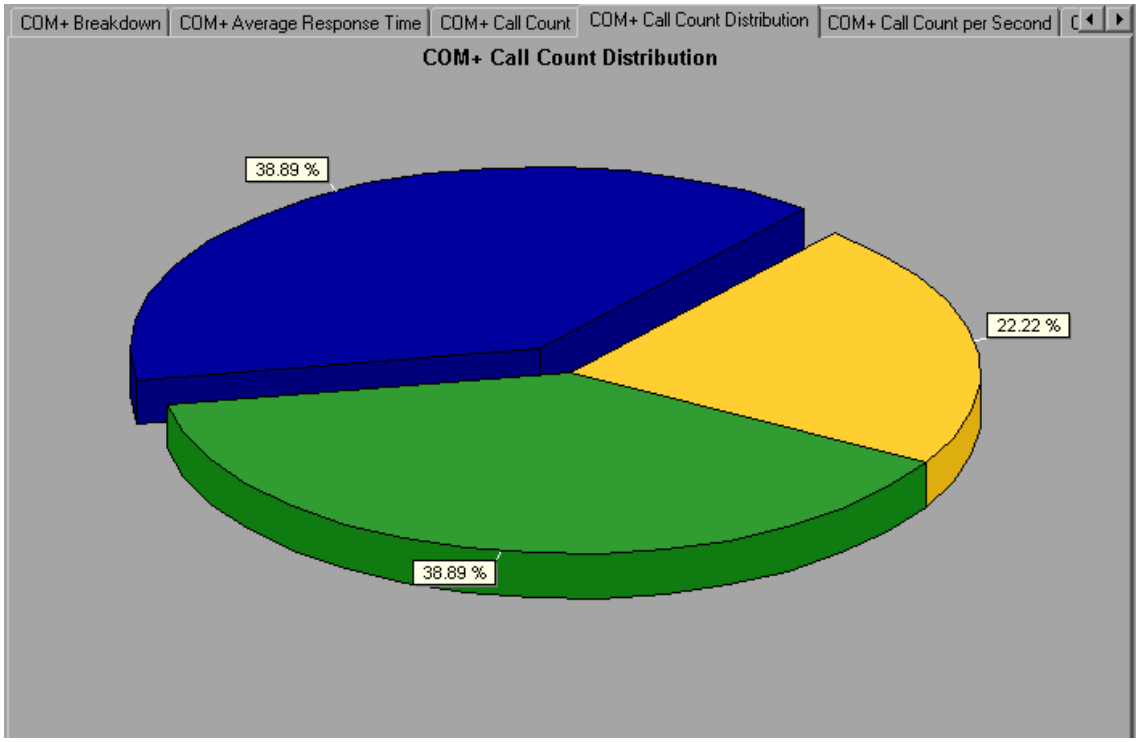
| | |
|---|---|
| **Tips** | **Sorting List**<br>To sort the list by a column, click on the column heading. The list above is sorted by **Average Response Time** which contains the triangle icon specifying a sort in descending order.<br>**Viewing COM+ Methods**<br>The table initially displays COM+ interfaces, but you can also view the list of COM+ methods.<br>To view the methods of a selected interface, select the **COM+ Methods** option. You can also double-click on the interface row to view the methods. The methods of the specified interface are listed in the **COM+ Method** column. |
| **See also** | "Application Component Graphs" on page 1373 |




## COM+ Call Count Distribution Graph

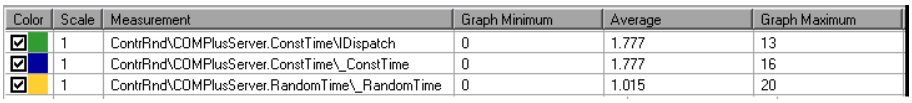
This graph shows the percentage of calls made to each COM+ interface compared to all COM+ interfaces. It can also show the percentage of calls made to a specific COM+ method compared to other methods within the interface

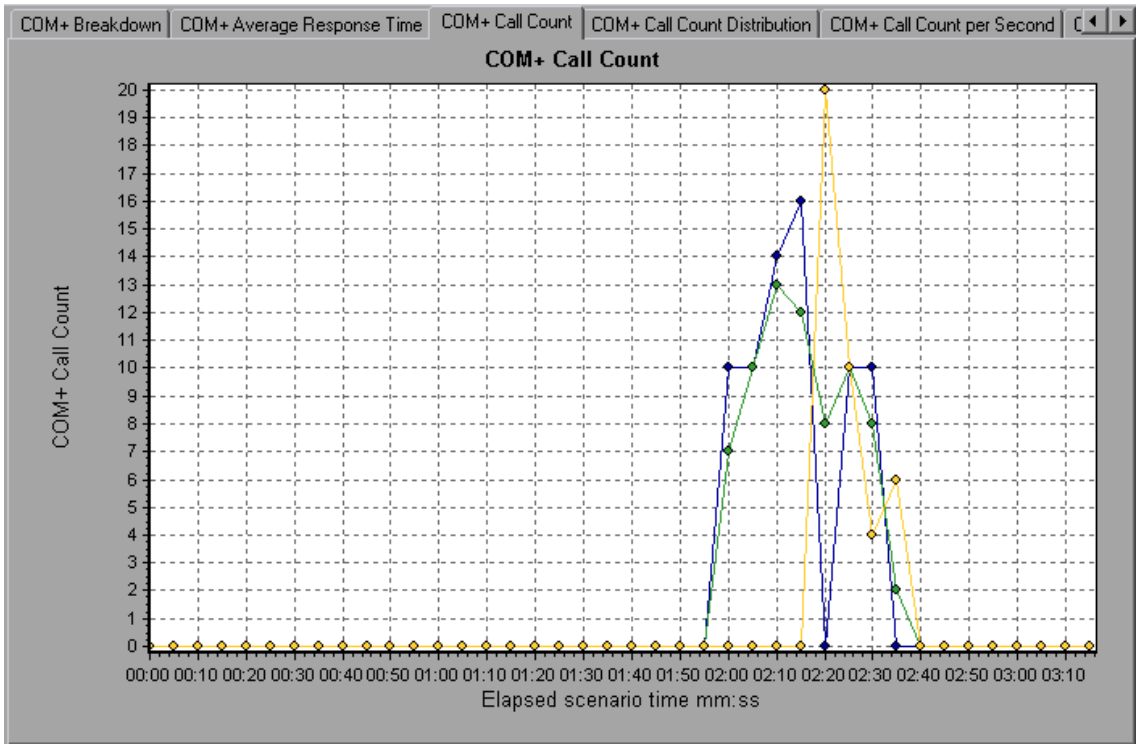
| | |
|---|---|
| **Breakdown options** | The number of calls made to the interface or method is listed in the **Call Count** column of the "COM+ Breakdown Graph" on page 1375 table.<br><br>Each interface or method is represented by a different colored area on the pie graph. The legend frame (which is found below the graph) identifies the interfaces by color:<br><br>Color \| Scale \| Measurement \| Graph Average<br>1 \| ContrRnd\COMPlusServer.ConstTime\IDispatch \| 70<br>1 \| ContrRnd\COMPlusServer.ConstTime_ConstTime \| 70<br>1 \| ContrRnd\COMPlusServer.RandomTime_RandomTime \| 40<br><br>This legend shows that the green colored area belongs to the COM+ interface **IDispatch.** Looking at the example graph below, we see that 38.89% of calls are made to this interface. The actual figures can be seen in the **Call Count** column of the "COM+ Breakdown Graph" on page 1375 table.<br><br>Viewing COM+ Methods<br><br>The table initially displays COM+ interfaces, but you can also view the list of COM+ methods by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific interface line in the graph, select the interface row in the legend. |
| **See also** | "Application Component Graphs" on page 1373 |



# COM+ Call Count Graph

This graph displays the number of times COM+ interfaces and methods are invoked during the test.

| | |
|---|---|
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | How many calls were made to a COM+ interface or method. |
| **Breakdown options** | Each interface or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the interfaces by color:<br>  <br>This legend shows that the yellow colored line belongs to the COM+ interface **_RandomTime.** Looking at the graph above, we see that calls to this interface begin at the beginning of the scenario run. There are 20 calls at the 2:20 minute point.<br>Viewing COM+ Methods<br>The table initially displays COM+ interfaces, but you can also view the list of COM+ methods by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Note** | The call count is computed by multiplying the call frequency by a time interval. As a result, the reported measurement may be rounded. |
| **Tips** | To highlight a specific interface line in the graph, select the interface row in the legend. |
| **See also** | "Application Component Graphs" on page 1373 |

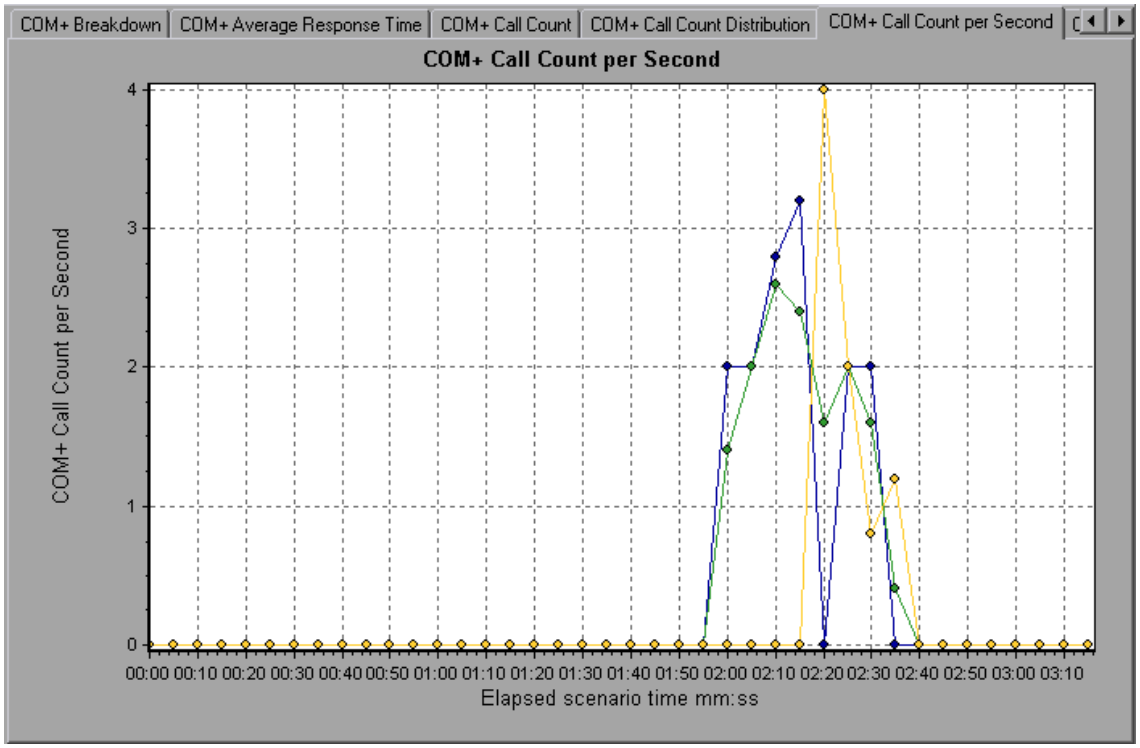



## COM+ Call Count Per Second Graph

This graph shows the number of times per second a COM+ interface or method is invoked.

| | |
|---|---|
| **Breakdown options** | This graph is similar to the "COM+ Call Count Graph" on the previous page except that the y-axis indicates how many invocations were made to a COM+ interface or method per second.<br><br>Each interface or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the interfaces by color:<br><br>Legend \| Graph Details \| User Notes \| Graph Data \| Raw Data<br><br>Color \| Scale \| Measurement \| Graph Minimum \| Average \| Graph Maximum<br>1 \| ContrRnd\COMPlusServer.ConstTime\IDispatch \| 0 \| 0.355 \| 2.6<br>1 \| ContrRnd\COMPlusServer.ConstTime_ConstTime \| 0 \| 0.355 \| 3.2<br>1 \| ContrRnd\COMPlusServer.RandomTime_RandomTime \| 0 \| 0.203 \| 4<br><br>This legend shows that the green colored line belongs to the COM+ interface **IDispatch.** Looking at the graph above, we see that calls to this interface begins 1:55 minutes into the scenario run. There is an average of 2.5 calls per second at the 2:10 minute mark.<br><br>Viewing COM+ Methods<br><br>To view the average response time of the individual methods within a COM+ interface, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |

| | |
|---|---|
| **Tips** | To highlight a specific interface line in the graph, select the interface row in the legend. |
| **See also** | "Application Component Graphs" on page 1373 |



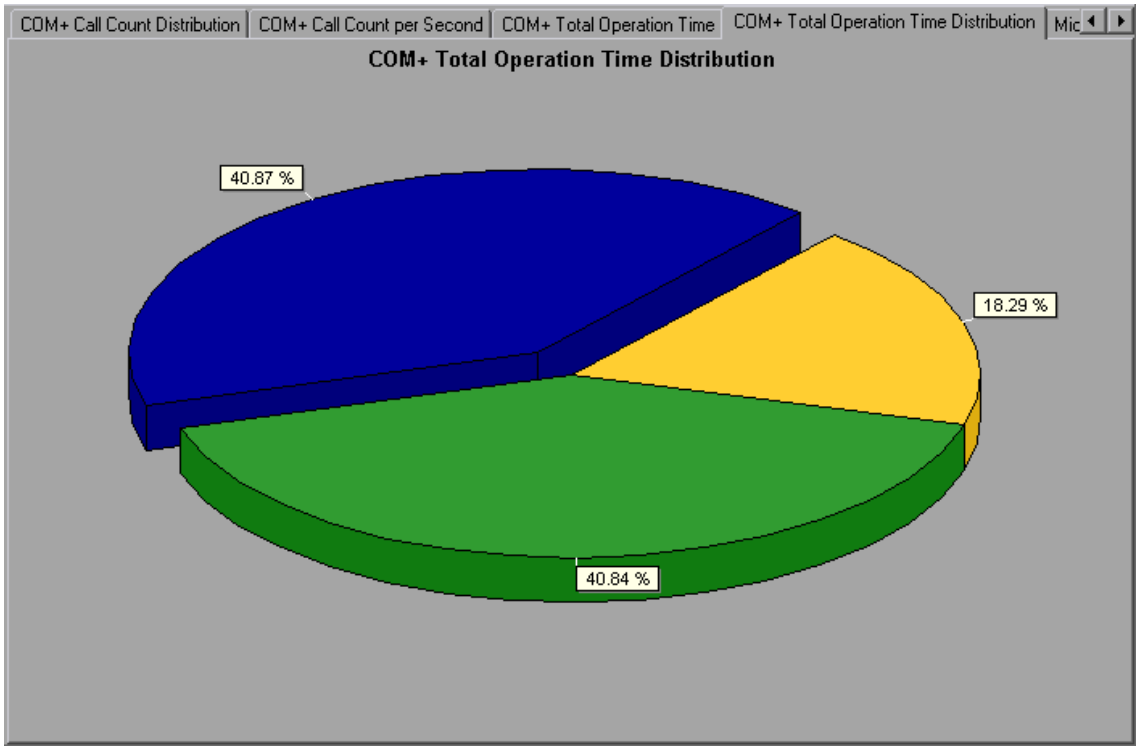

## COM+ Total Operation Time Distribution Graph

This graph shows the percentage of time a specific COM+ interface takes to execute in relation to all COM+ interfaces. It can also show the percentage of time a COM+ method takes to execute in relation to all COM+ methods within the interface.

| | |
|---|---|
| **Purpose** | Use it to identify those interfaces or methods which take up an excessive amount of time. |

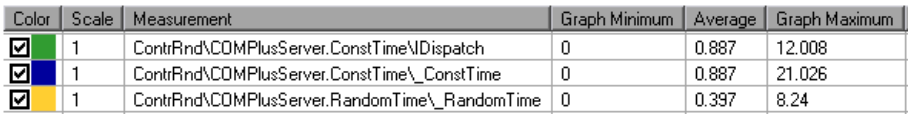
| | |
|---|---|
| **Breakdown options** | Each interface or method is represented by a different colored area on the pie graph. The legend frame (which is found below the graph) identifies the interfaces by color:<br><br>This legend shows that the green colored line belongs to the COM+ interface **IDispatch.** Looking at the graph above, we see that this interface takes up 40.84% of the COM+ operational time.<br><br>Viewing COM+ Methods<br><br>To view the average response time of the individual methods within a COM+ interface, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific interface line in the graph, select the interface row in the legend. |
| **See also** | "Application Component Graphs" on page 1373 |

| Color | Scale | Measurement | Graph Average |
|---|---|---|---|
| | 1 | ContrRnd\COMPlusServer.ConstTime\IDispatch | 34.936 |
| | 1 | ContrRnd\COMPlusServer.ConstTime_ConstTime | 34.966 |
| | 1 | ContrRnd\COMPlusServer.RandomTime_RandomTime | 15.642 |

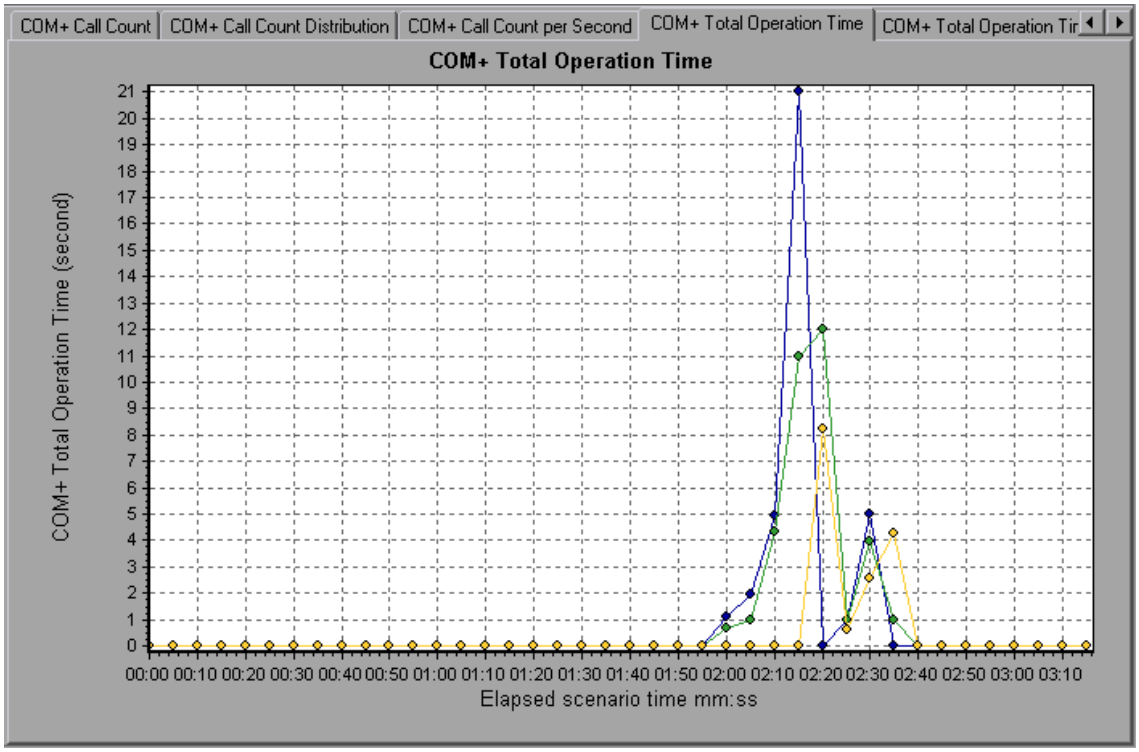



## COM+ Total Operation Time Graph

This graph displays the amount of time each COM+ interface or method takes to execute during the test.

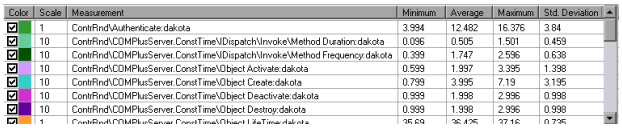
| | |
|---|---|
| **Purpose** | Use it to identify those interfaces or methods which take up an excessive amount of time. |
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | Total time a COM+ interface or method is in operation. |
| **Breakdown options** | Each interface or method is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the interfaces by color:<br>  <br>This legend shows that the blue colored line belongs to the COM+ interface **_ConstTime.** Looking at the graph above, we see that throughout the scenario, this interface consumes more time than any other, especially at 2 minutes and 15 seconds into the scenario run, where the calls to this interface take an average of 21 seconds.<br>Viewing COM+ Methods<br>The table initially displays COM+ interfaces, but you can also view the list of COM+ methods by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific interface line in the graph, select the interface row in the legend. |
| **See also** | "Application Component Graphs" on page 1373 |

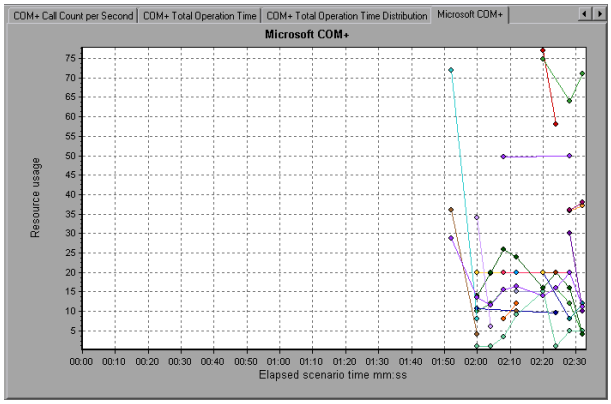
| Color | Scale | Measurement | Graph Minimum | Average | Graph Maximum |
|---|---|---|---|---|---|
| ☑ | 1 | ContrRnd\COMPlusServer.ConstTime\IDispatch | 0 | 0.887 | 12.008 |
| ☑ | 1 | ContrRnd\COMPlusServer.ConstTime_ConstTime | 0 | 0.887 | 21.026 |
| ☑ | 1 | ContrRnd\COMPlusServer.RandomTime_RandomTime | 0 | 0.397 | 8.24 |




## Microsoft COM+ Graph

This graph shows the resource usage of COM+ objects as a function of the elapsed load test scenario time.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource usage of COM+ objects. |
| **Breakdown Options** | Each COM+ object is represented by a different colored line on the graph. The legend frame (which is found below the graph) identifies the objects by color:<br>  |
| **See also** | "Application Component Graphs" on page 1373 |




## Authentication Metrics

| Measurement | Description |
|---|---|
| **Authenticate** | Frequency of successful method call level authentication. When you set an authentication level for an application, you determine what degree of authentication is performed when clients call into the application. |
| **Authenticate Failed** | Frequency of failed method call level authentication. |

## Application Event

| Measurement | Description |
|---|---|
| **Activation** | Frequency of application activation or startup. |
| **Shutdown** | Frequency of application shutdown or termination. |

## Thread Event

| Measurement | Description |
|---|---|
| **Thread Start** | Rate at which single-threaded apartment (STA) thread for application have been started. |
| **Thread Terminate** | Rate at which single-threaded apartment (STA) thread for application have been terminated. |
| **Work Enque** | Event sent if a work is queued in single thread apartment object (STA). Note: These events are not signaled/sent in Windows Server 2003 and later. |
| **Work Reject** | Event sent if a work is rejected from single thread apartment object (STA). Note: These events are not signaled/sent in Windows Server 2003 and later. |

## Transaction Events

| Measurement | Description |
|---|---|
| **Transaction Duration** | Duration of COM+ transactions for selected application. |
| **Transaction Start** | Rate at which transactions have started. |
| **Transaction Prepared** | Rate at which transactions have completed the prepare phase of the two-phase protocol. |
| **Transaction Aborted** | Rate at which transactions have been aborted. |
| **Transaction Commit** | Rate at which transactions have completed the commit protocol. |

## Object Events

| Measurement | Description |
|---|---|
| **Object Life Time** | Duration of object existence (from instantiation to destruction). |
| **Object Create** | Rate at which new instances of this object are created. |
| **Object Destroy** | Rate at which instances of the object are destroyed. |
| **Object Activate** | Rate of retrieving instances of a new JIT-activated object. |
| **Object Deactivation** | Rate of freeing JIT-activated object via SetComplete or SetAbort. |
| **Disable Commit** | Rate of client calls to DisableCommit on a context. DisableCommit declares that the object's transactional updates are inconsistent and cannot be committed in their present state. |
| **Enable Commit** | Rate of client calls to EnableCommit on a context. EnableCommit declares that the current object's work is not necessarily finished, but that its transactional updates are consistent and could be committed in their present form. |

| Measurement | Description |
|---|---|
| **Set Complete** | Rate of client calls to SetComplete on a context. SetComplete declares that the transaction in which the object is executing can be committed, and that the object should be deactivated on returning from the currently executing method call. |
| **Set Abort** | Rate of client calls to SetAbort on a context. SetAbort declares that the transaction in which the object is executing must be aborted, and that the object should be deactivated on returning from the currently executing method call. |

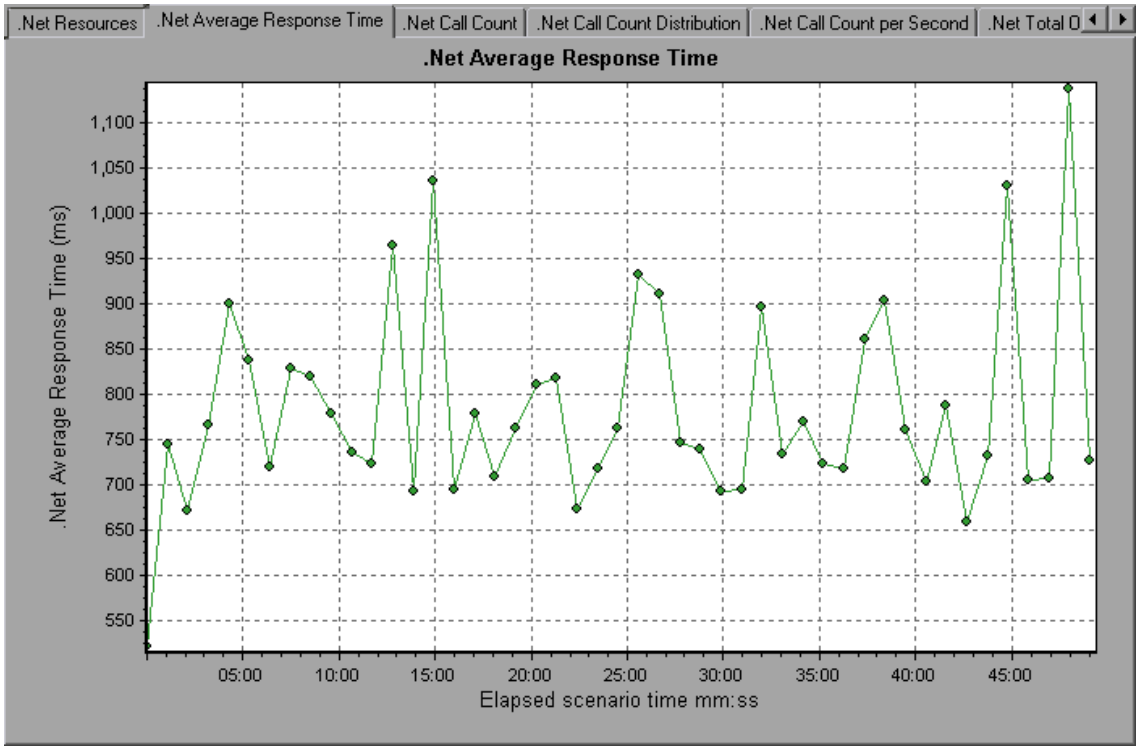
### Method Events

| Measurement | Description |
|---|---|
| **Method Duration** | Average duration of method. |
| **Method Frequency** | Frequency of method invocation. |
| **Method Failed** | Frequency of failed methods (i.e. methods that return error HRESULT codes). |
| **Method Exceptions** | Frequency of exceptions thrown by selected method. |

## .NET Average Response Time Graph

This graph specifies the average time that .NET classes or methods took to perform during the load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | Average response time of a .NET class or method. |
| **Breakdown options** | The graph initially displays .NET classes, but you can also view the individual methods within a .NET class by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | You can change the length of the sample interval.<br>**Hint**: To highlight a specific class line in the graph, select the class row in the legend (displayed below the graph). |
| **See also** | "Application Component Graphs" on page 1373 |



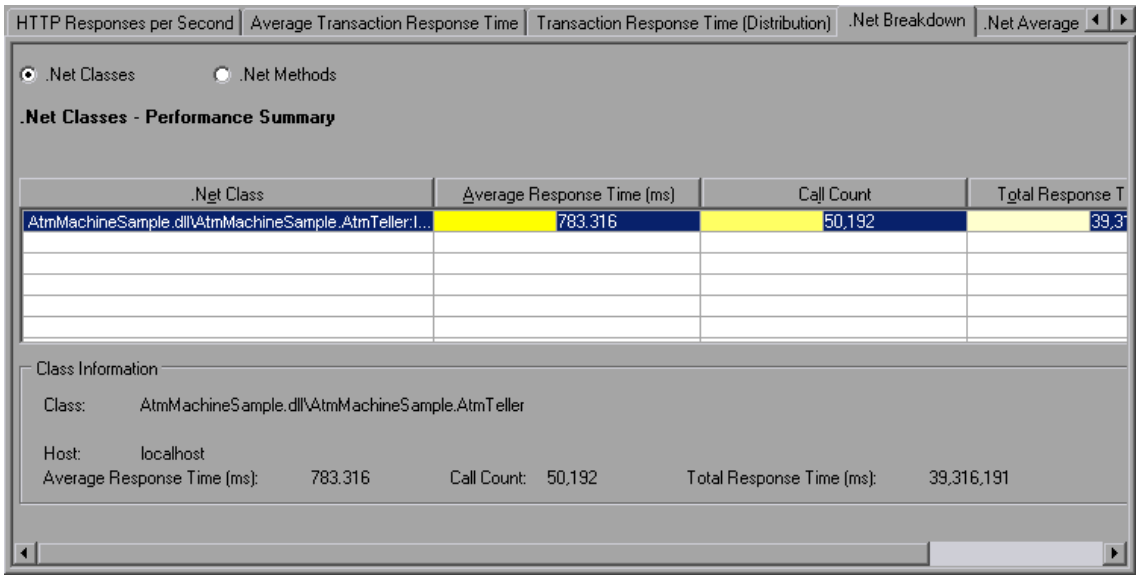

## .NET Breakdown Graph

This graph summarizes fundamental result data about .NET classes or methods and presents it in table format.

| | |
|---|---|
| **Purpose** | Using the .NET Breakdown table, you can identify the .NET classes or methods which consume the most time during the test. The table can be sorted by column, and the data can be viewed either by .NET class or .NET method. |

| | |
|---|---|
| **Breakdown options** | The **Average Response Time** column shows how long, on average, a class or method took to perform. The next column, **Call Count**, specifies the number of times the class or method was invoked. The final column, **Total Response Time**, specifies how much time was spent overall on the class or method. It is calculated by multiplying the results from the first two columns together.<br><br>Classes are listed in the **.NET Class** column in the form **Class:Host**. In the table above, the **AtmMachineSample.AtmTeller** class took an average of 783 seconds to execute and was called 50,912 times. Overall, this class took 39,316 seconds to execute.<br><br>To sort the list by a column, click the column heading.<br><br>Each column in the .NET Breakdown graph is graphically represented by another graph.<br><br>The table initially displays .NET classes, but you can also view the list of .NET methods. To view .NET methods, select the **.NET Methods** option, or double-click the class row. The methods of the specified class are listed in the **.NET Method** column. |
| **See also** | "Application Component Graphs" on page 1373 |

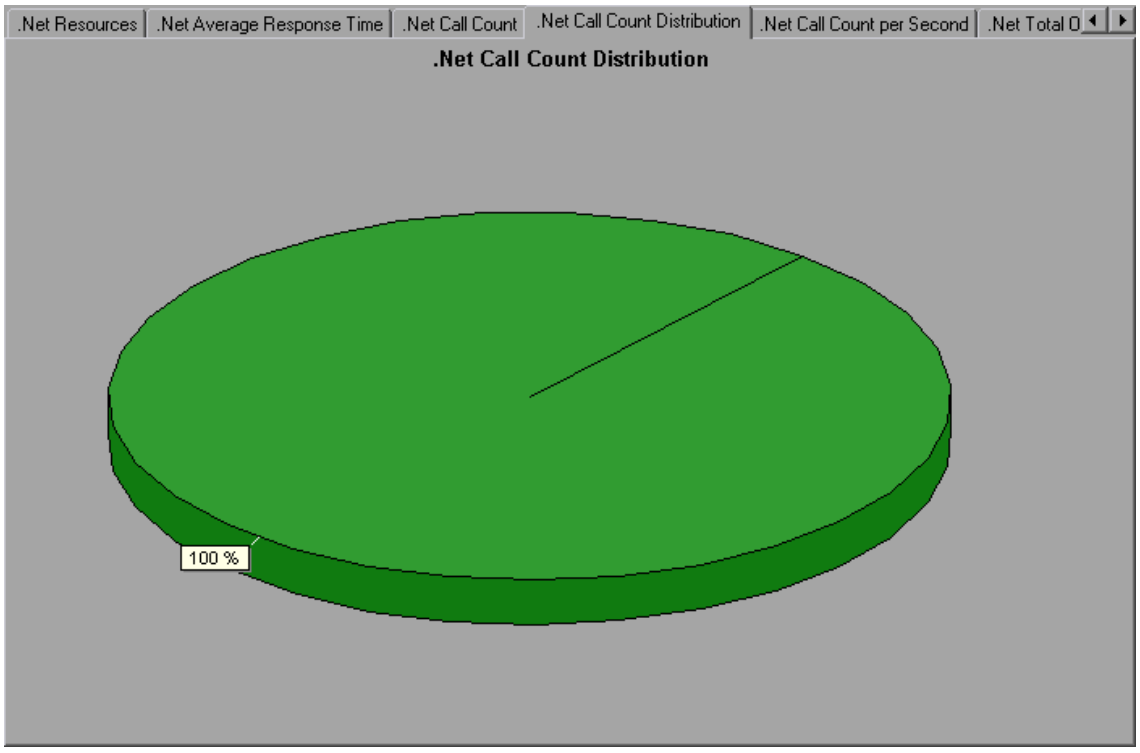
| .NET Breakdown Column | Graphical Representation |
|---|---|
| Average Response Time | .NET Average Response Time Graph. |
| Call Count | .NET Call Count Graph. |
| Total Response Time | .NET Total Operation Time Distribution Graph. |



## .NET Call Count Distribution Graph

This graph shows the percentage of calls made to each .NET class compared to all .NET classes. It can also show the percentage of calls made to a specific .NET method compared to other methods within the class

| | |
|---|---|
| **Breakdown options** | The number of calls made to the class or method is listed in the **Call Count** column of the .NET Breakdown graph table.<br><br>The graph initially displays .NET classes, but you can also view the individual methods within a .NET class by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific class line in the graph, select the class row in the legend (displayed below the graph). |
| **See also** | "Application Component Graphs" on page 1373 |

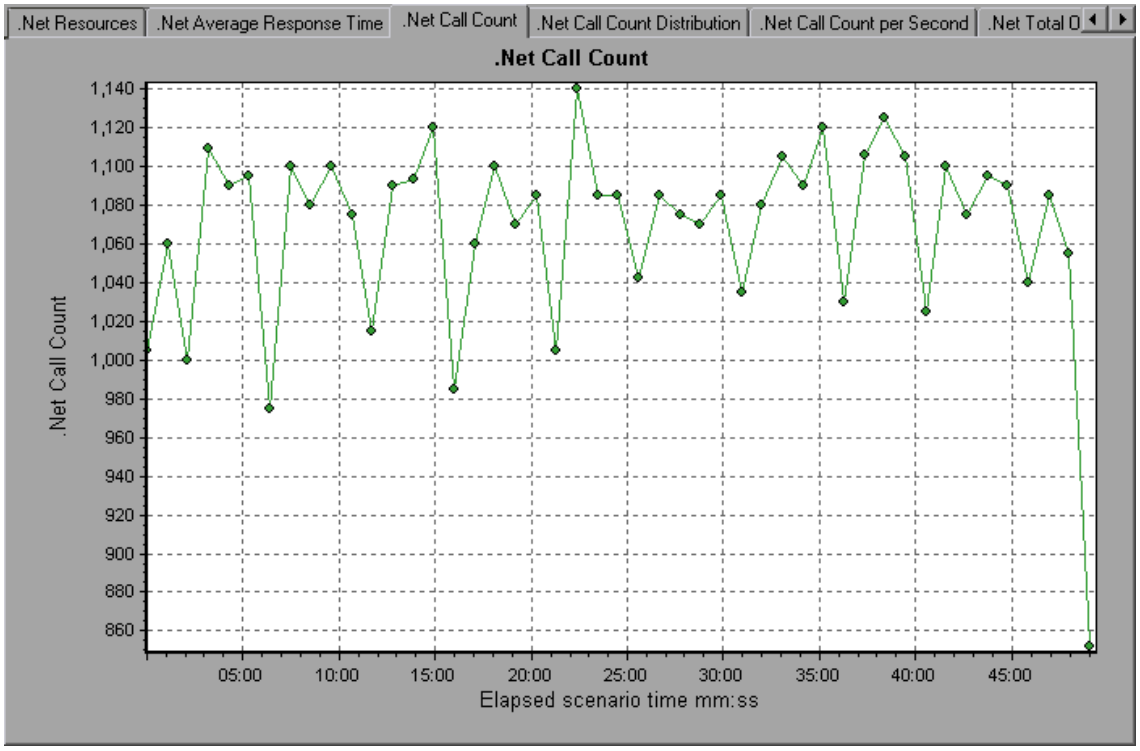



## .NET Call Count Graph

This graph displays the number of times that .NET classes and methods are invoked during the test.

| | |
|---|---|
| **X-axis** | Elapsed time from the beginning of the scenario run. |

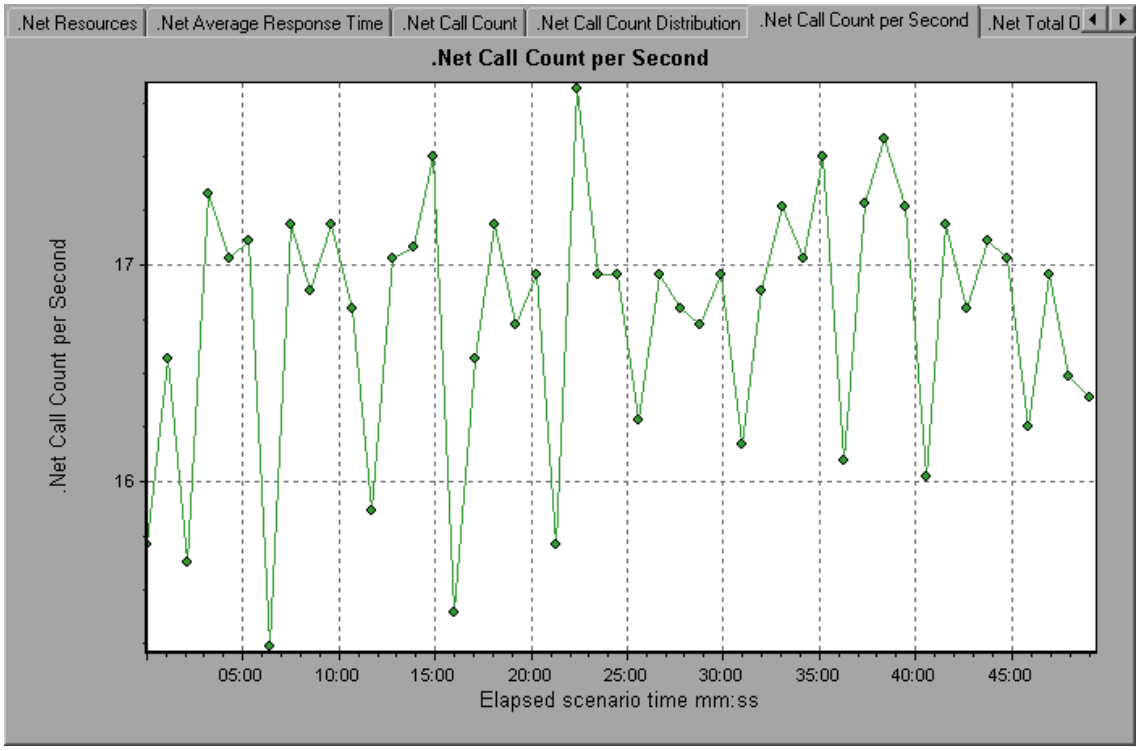
| | |
|---|---|
| **Y-axis** | Indicates how many calls were made to a .NET class or method. |
| **Breakdown options** | The graph initially displays .NET classes, but you can also view the individual methods within a .NET class by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific class line in the graph, select the class row in the legend (displayed below the graph). |
| **Note** | The call count is computed by multiplying the call frequency by a time interval. As a result, the reported measurement may be rounded. |
| **See also** | "Application Component Graphs" on page 1373 |




## .NET Call Count per Second Graph

This graph shows the number of times per second that a .NET class or method is invoked.

| | |
|---|---|
| **Breakdown options** | This graph is similar to the .NET Call Count graph except that the y-axis indicates how many invocations were made to a .NET class or method per second.<br><br>The graph initially displays .NET classes, but you can also view the individual methods within a .NET class by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific class line in the graph, select the class row in the legend (displayed below the graph). |
| **See also** | "Application Component Graphs" on page 1373 |




## .NET Resources Graph

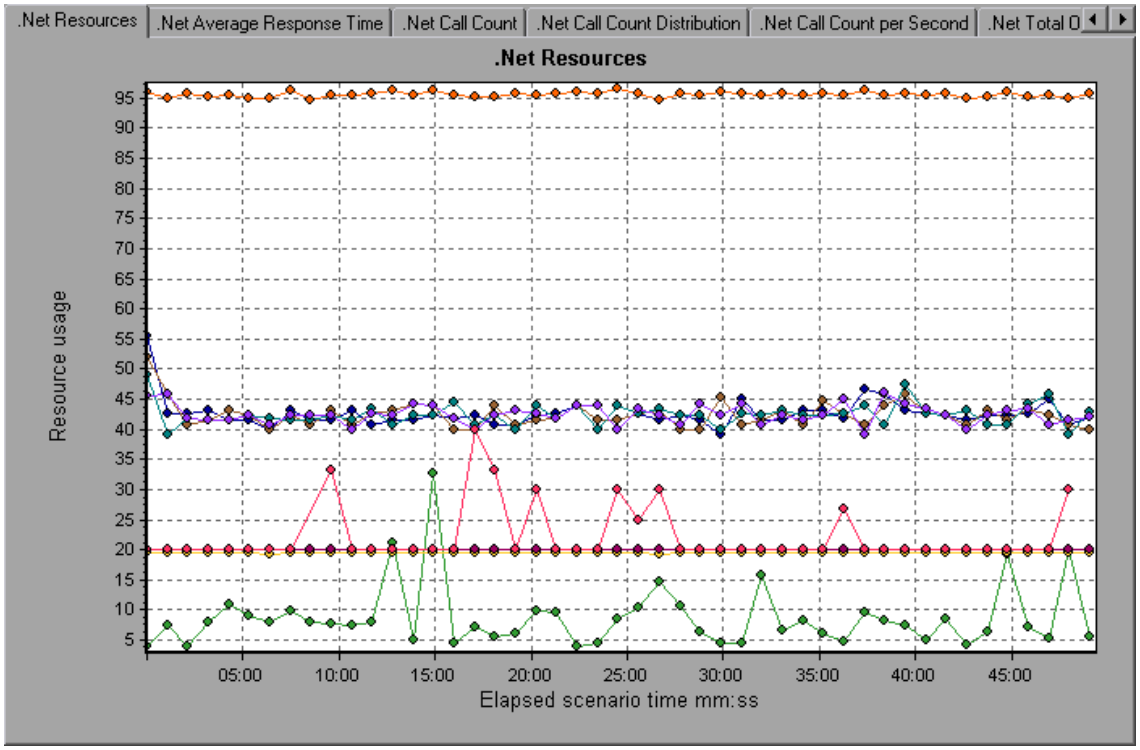
This graph shows the resource usage of .NET methods as a function of the elapsed load test scenario time.

Each .NET method is represented by a different colored line on the graph. The legend frame (located below the graph) identifies the methods by color:

| Color | Scale | Measurement | Minimum | Average | Maximum |
|---|---|---|---|---|---|
| ☑ | 0.01 | AtmMachineSample.dll\AtmMachineSample.At... | 390.749 | 888.061 | 37848.727 |
| ☑ | 10 | AtmMachineSample.dll\AtmMachineSample.At... | 1 | 4.244 | 10 |
| ☑ | 0.1 | AtmMachineSample.dll\AtmMachineSample.At... | 190.944 | 194.783 | 207.318 |
| ☑ | 10 | AtmMachineSample.dll\AtmMachineSample.At... | 1 | 4.235 | 10 |

You can monitor .NET counters at the application, assembly, class, and method levels. Measurements that take place before the application is fully loaded (such as Assembly Load Time, that measures the time it takes to load an assembly) will not be measured.

The following tables describe the counters that can be measured at each level. All durations are reported in seconds, and all frequencies are reported per five-second polling periods. For example, if 20 events occur in a 5 second polling period, the reported frequency is 4.




## Application Level

| Measurement | Description |
|---|---|
| **Application Lifetime** | Monitors the duration of the application in seconds. |
| **Exception Frequency** | Monitors the number of exceptions per second, in the five second polling period. |
| **JIT (Just In Time) Duration** | Monitors the time (in seconds) it takes for the JIT to compile code. |
| **Thread Creation Frequency** | Monitors the number of threads that are created in a polling period. |

| Measurement | Description |
|---|---|
| **Thread Lifetime** | Monitors the duration of threads. |
| **Domain Creation Frequency** | Monitors the number of domain creations in a polling period. (Domains protect areas of code. All applications run in a domain which keeps them encapsulated, so that they cannot interfere with other applications outside the domain.) |
| **Domain Load Time** | Monitors the time it takes to load a domain. (Domains protect areas of code. All applications run in a domain which keeps them encapsulated, so that they cannot interfere with other applications outside the domain). |
| **Domain Unload Time** | Monitors the time it takes to unload a domain. (Domains protect areas of code. All applications run in a domain which keeps them encapsulated, so that they cannot interfere with other applications outside the domain). |
| **Domain Lifetime** | Monitors the duration of a domain. (Domains protect areas of code. All applications run in a domain which keeps them encapsulated, so that they cannot interfere with other applications outside the domain). |
| **Module Creation Frequency** | Monitors the number of modules that get created in a polling period. (Modules are groups of assemblies that make up a DLL or EXE). |
| **Module Load Time** | Monitors the time it takes to load a module. (Modules are groups of assemblies that make up a dll or exe). |
| **Module Unload Time** | Monitors the time it takes to unload a module. (Modules are groups of assemblies that make up a dll or exe). |
| **Module Lifetime** | Monitors the duration of a module. (Modules are groups of assemblies that make up a dll or exe). |
| **Garbage Collection Duration** | Monitors the duration between the start and stop of Garbage Collection. |
| **Garbage Collection Frequency** | Monitors the number of breaks for Garbage Collections in a polling period. |
| **Unmanaged Code Duration** | Monitors the duration of the calls to unmanaged code. |
| **Unmanaged Code Frequency** | Monitors the number of calls to unengaged code in a polling period. |

### Assembly Level

| Measurement | Description |
|---|---|
| **Assembly Creation Frequency** | Monitors the number of assembly creations in a polling period. (Assemblies hold the .NET byte code and metadata). |
| **Assembly Load Time** | Monitors the time it takes to load an assembly. (Assemblies hold the .NET byte code and metadata). |
| **Assembly Unload Time** | Monitors the time it takes to unload an assembly. (Assemblies hold the .NET byte code and metadata). |
| **Assembly Lifetime** | Monitors the duration of an assembly. (Assemblies hold the .NET byte code and metadata). |

### Class Level

| Measurement | Description |
|---|---|
| **Class Lifetime** | Monitors the duration of a class. |
| **Class Load Time** | Monitors the time it takes to load a class. |
| **Class Unload Time** | Monitors the time it takes to unload a class. |

### Method Level

At the method level, the measured time is per method, exclusive of other methods, calls to unmanaged code, and garbage collection time.

| Measurement | Description |
|---|---|
| **Method Duration** | Monitors the duration of a method. |
| **Method Frequency** | Monitors the number of methods called in a polling period. |

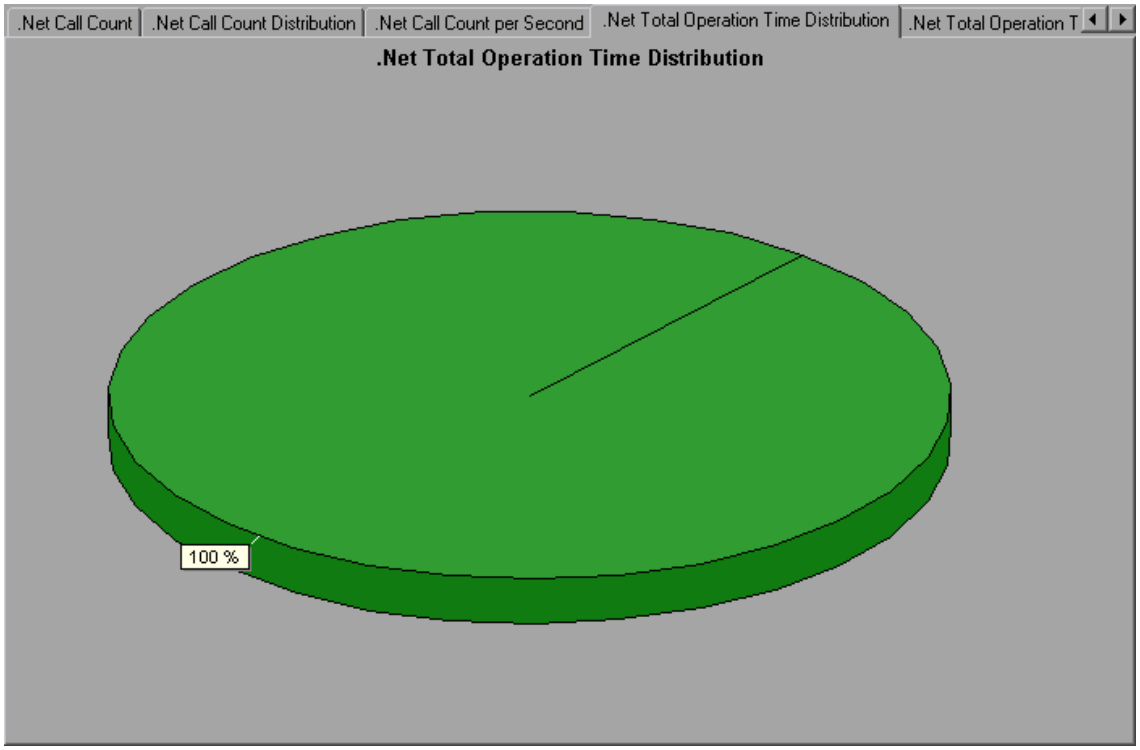
### See also:

- "Application Component Graphs" on page 1373

## .NET Total Operation Time Distribution Graph

This graph shows the percentage of time that a specific .NET class took to execute in relation to all the .NET classes. It can also show the percentage of time that a .NET method took to execute in relation to all the .NET methods within the class.

| | |
|---|---|
| **Purpose** | Use this graph to identify those classes or methods that take an excessive amount of time. |
| **Breakdown options** | The graph initially displays .NET classes, but you can also view the individual methods within a .NET class by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific class line in the graph, select the class row in the legend (displayed below the graph). |
| **See also** | "Application Component Graphs" on page 1373 |

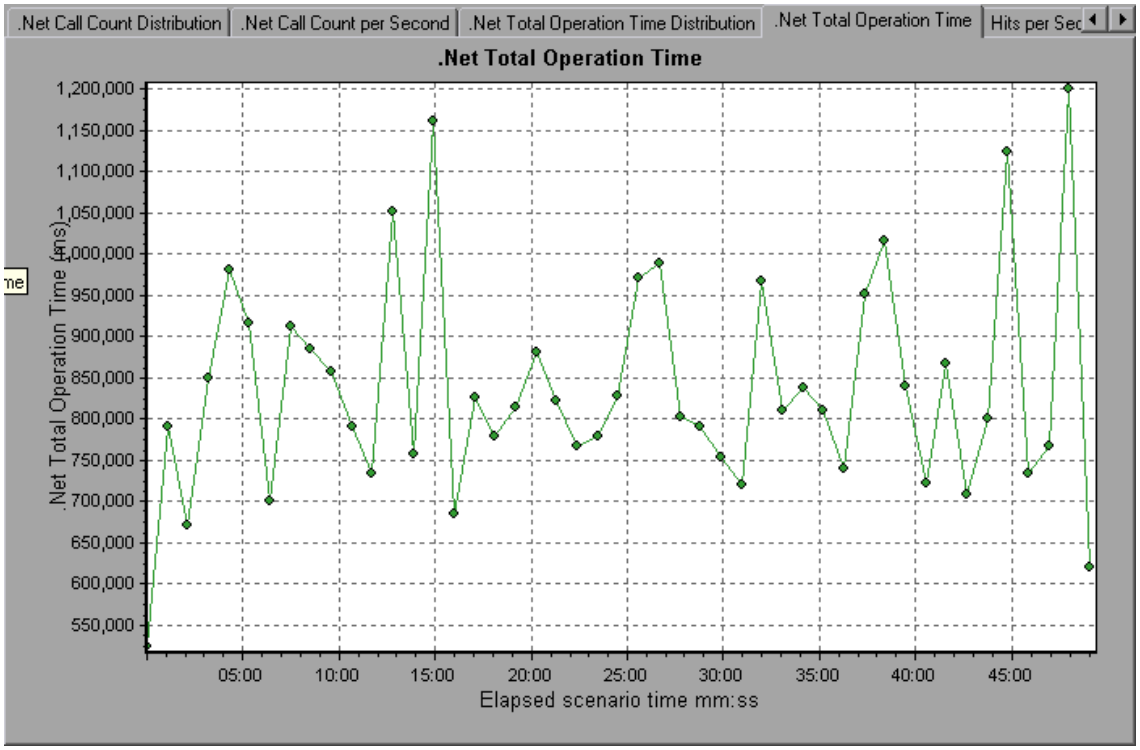



## .NET Total Operation Time Graph

This graph displays the amount of time that each .NET class or method took to execute during the test.

| | |
|---|---|
| **Purpose** | Use this graph to identify those classes or methods that take an excessive amount of time. |
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | Total time a .NET class or method is in operation. |

| | |
|---|---|
| **Breakdown options** | The graph initially displays .NET classes, but you can also view the individual methods within a .NET class by using drill-down or filtering techniques. For more information, see "Filtering and Sorting Graph Data" on page 1342 and "Drilling Down in a Graph" on page 1328. |
| **Tips** | To highlight a specific class line in the graph, select the class row in the legend (displayed below the graph). |
| **See also** | "Application Component Graphs" on page 1373 |




# Application Deployment Solutions Graphs

LoadRunner's Citrix Server monitor provides you with information about the application deployment usage of the Citrix server during a load test scenario execution. In order to obtain performance data, before you execute the scenario you need to activate the online monitor for the server and specify which resources you want to measure.

For more information on activating and configuring the Citrix monitors, see the section on online monitors in the LoadRunner Controller documentation.

## Citrix Measurements

LoadRunner's Citrix Server monitor provides you with information about the application deployment usage of the Citrix server during a load test scenario execution.

## Non-Virtual Counters

| Measurement | Description |
|---|---|
| **% Disk Time** | The percentage of elapsed time that the selected disk drive services read or write requests. |
| **% Processor Time** | The percentage of time that the processor executes a non-Idle thread. This counter is a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the Idle process in each sample interval, and subtracting that value from 100%. (Each processor has an Idle thread which consumes cycles when no other threads are ready to run.) It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%. |
| **File data Operations/sec** | The rate that the computer issues Read and Write operations to file system devices. It does not include File Control Operations. |
| **Interrupts/sec** | The average number of hardware interrupts the processor receives and services per second. It does not include DPCs, which are counted separately. This value is an indirect indicator of the activity of devices that generate interrupts, such as the system clock, the mouse, disk drivers, data communication lines, network interface cards and other peripheral devices. These devices normally interrupt the processor when they have completed a task or require attention. Normal thread execution is suspended during interrupts. Most system clocks interrupt the processor every 10 milliseconds, creating a background of interrupt activity. This counter displays the difference between the values observed in the last two samples, divided by the duration of the sample interval. |
| **Output Session Line Speed** | This value represents the line speed from server to client for a session in bps. |
| **Input Session Line Speed** | This value represents the line speed from client to server for a session in bps. |
| **Page Faults/sec** | A count of the Page Faults in the processor. A page fault occurs when a process refers to a virtual memory page that is not in its Working Set in main memory. A Page Fault will not cause the page to be fetched from disk if that page is on the standby list, and hence already in main memory, or if it is in use by another process with whom the page is shared. |

| Measurement | Description |
|---|---|
| **Pages/sec** | The number of pages read from the disk or written to the disk to resolve memory references to pages that were not in memory at the time of the reference. This is the sum of Pages Input/sec and Pages Output/sec. This counter includes paging traffic on behalf of the system Cache to access file data for applications. This value also includes the pages to/from non-cached mapped memory files. This is the primary counter to observe if you are concerned about excessive memory pressure (that is, thrashing), and the excessive paging that may result. |
| **Pool Nonpaged Bytes** | The number of bytes in the Nonpaged Pool, a system memory area where space is acquired by operating system components as they accomplish their appointed tasks. Nonpaged Pool pages cannot be paged out to the paging file, but instead remain in main memory as long as they are allocated. |
| **Private Bytes** | The current number of bytes this process has allocated that cannot be shared with other processes. |
| **Processor Queue Length** | The instantaneous length of the processor queue in units of threads. This counter is always 0 unless you are also monitoring a thread counter. All processors use a single queue in which threads wait for processor cycles. This length does not include the threads that are currently executing. A sustained processor queue length greater than two generally indicates processor congestion. This is an instantaneous count, not an average over the time interval. |
| **Threads** | The number of threads in the computer at the time of data collection. Notice that this is an instantaneous count, not an average over the time interval. A thread is the basic executable entity that can execute instructions in a processor. |
| **Latency – Session Average** | The average client latency over the life of a session. |
| **Latency – Last Recorded** | The last recorded latency measurement for this session. |
| **Latency – Session Deviation** | The difference between the minimum and maximum measured values for a session. |
| **Input Session Bandwidth** | The bandwidth (in bps) from client to server traffic for a session in bps. |
| **Input Session Compression** | The compression ratio for client to server traffic for a session. |

| Measurement | Description |
|---|---|
| **Output Session Bandwidth** | The bandwidth (in bps) from server to client traffic for a session. |
| **Output Session Compression** | The compression ratio for server to client traffic for a session. |
| **Output Session Linespeed** | The line speed (in bps) from server to client for a session. |

## Virtual Channel Counters

All the counters in the following table are measured in bytes per second (bps):

| Measurement | Description |
|---|---|
| **Input Audio Bandwidth** | The bandwidth from client to server traffic on the audio mapping channel. |
| **Input Clipboard Bandwidth** | The bandwidth from client to server traffic on the clipboard mapping channel. |
| **Input COM1 Bandwidth** | The bandwidth from client to server traffic on the COM1 channel. |
| **Input COM2 Bandwidth** | The bandwidth from client to server traffic on the COM2 channel. |
| **Input COM Bandwidth** | The bandwidth from client to server traffic on the COM channel. |
| **Input Control Channel Bandwidth** | The bandwidth from client to server traffic on the ICA control channel. |
| **Input Drive Bandwidth** | The bandwidth from client to server traffic on the client drive mapping channel. |
| **Input Font Data Bandwidth** | The bandwidth from client to server traffic on the local text echo font and keyboard layout channel. |
| **Input Licensing Bandwidth** | The bandwidth from server to client traffic on the licensing channel. |
| **Input LPT1 Bandwidth** | The bandwidth from client to server traffic on the LPT1 channel. |
| **Input LPT2 Bandwidth** | The bandwidth from client to server traffic on the LPT2 channel. |

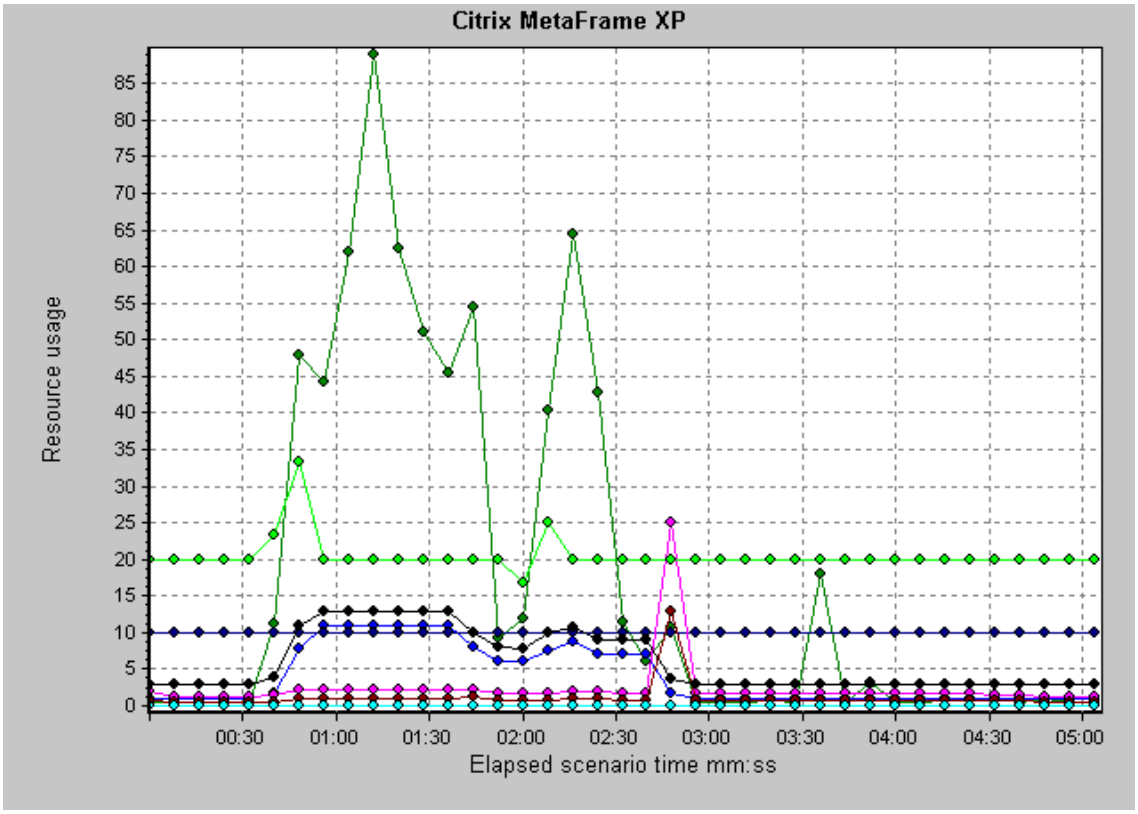
| Measurement | Description |
|---|---|
| **Input Management Bandwidth** | The bandwidth from client to server traffic on the client management channel. |
| **Input PN Bandwidth** | The bandwidth from client to server traffic on the Program Neighborhood channel. |
| **Input Printer Bandwidth** | The bandwidth from client to server traffic on the printer spooler channel. |
| **Input Seamless Bandwidth** | The bandwidth from client to server traffic on the Seamless channel. |
| **Input Text Echo Bandwidth** | The bandwidth from client to server traffic on the local text echo data channel. |
| **Input Thinwire Bandwidth** | The bandwidth from client to server traffic on the Thinwire (graphics) channel. |
| **Input VideoFrame Bandwidth** | The bandwidth from client to server traffic on the VideoFrame channel. |
| **Output Audio Bandwidth** | The bandwidth from server to client traffic on the audio mapping channel. |
| **Output Clipboard Bandwidth** | The bandwidth from server to client traffic on the clipboard mapping channel. |
| **Output COM1 Bandwidth** | The bandwidth from server to client traffic on the COM1 channel. |
| **Output COM2 Bandwidth** | The bandwidth from server to client traffic on the COM2 channel. |
| **Output COM Bandwidth** | The bandwidth from server to client traffic on the COM channel. |
| **Output Control Channel Bandwidth** | The bandwidth from server to client traffic on the ICA control channel. |
| **Output Drive Bandwidth** | The bandwidth from server to client traffic on the client drive channel. |
| **Output Font Data Bandwidth** | The bandwidth from server to client traffic on the local text echo font and keyboard layout channel. |
| **Output Licensing Bandwidth** | The bandwidth from server to client traffic on the licensing channel. |
| **Output LPT1 Bandwidth** | The bandwidth from server to client traffic on the LPT1 channel. |
| **Output LPT2 Bandwidth** | The bandwidth from server to client traffic on the LPT2 channel. |

| Measurement | Description |
|---|---|
| **Output Management Bandwidth** | The bandwidth from server to client traffic on the client management channel. |
| **Output PN Bandwidth** | The bandwidth from server to client traffic on the Program Neighborhood channel. |
| **Output Printer Bandwidth** | The bandwidth from server to client traffic on the printer spooler channel. |
| **Output Seamless Bandwidth** | The bandwidth from server to client traffic on the Seamless channel. |
| **Output Text Echo Bandwidth** | The bandwidth from server to client traffic on the local text echo data channel. |
| **Output Thinwire Bandwidth** | The bandwidth from server to client traffic on the Thinwire (graphics) channel. |
| **Output VideoFrame Bandwidth** | The bandwidth from server to client traffic on the VideoFrame channel. |

## Citrix Server Graph

This graph is an Application Deployment solution which delivers applications across networks. The Citrix Server monitor is an Application Deployment Solution monitor, which provides performance information for the Citrix server.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource usage on the Citrix server. |
| **Note** | To obtain data for this graph, you need to enable the Citrix Server monitor (from Controller) and select the default measurements you want to display, before running the scenario. |
| **See also** | "Application Deployment Solutions Graphs" on page 1396<br>"Citrix Measurements" on page 1396 |




# Database Server Resource Graphs

The Database Server Resource graphs show statistics for several database servers. Currently Oracle and SQL Server databases are supported. These graphs require that you specify the resources you want to measure *before* running the load test scenario. For more information, see the section on online monitors in the LoadRunner Controller documentation.

## Oracle Server Monitoring Measurements

The following measurements are most commonly used when monitoring the Oracle server (from the V$SYSSTAT table):

| Measurement | Description |
|---|---|
| **CPU used by this session** | The amount of CPU time (in tens of milliseconds) used by a session between the time a user call started and ended. Some user calls can be completed within 10 milliseconds and, as a result, the start- and end-user call time can be the same. In this case, 0 milliseconds are added to the statistic. A similar problem can exist in the operating system reporting, especially on systems that suffer from many context switches. |

| Measurement | Description |
|---|---|
| **Bytes received via SQL*Net from client** | The total number of bytes received from the client over Net8. |
| **Logons current** | The total number of current logons. |
| **Opens of replaced files** | The total number of files that needed to be reopened because they were no longer in the process file cache. |
| **User calls** | Oracle allocates resources (Call State Objects) to keep track of relevant user call data structures every time you log in, parse, or execute. When determining activity, the ratio of user calls to RPI calls gives you an indication of how much internal work is generated as a result of the type of requests the user is sending to Oracle. |
| **SQL*Net roundtrips to/from client** | The total number of Net8 messages sent to, and received from, the client. |
| **Bytes sent via SQL*Net to client** | The total number of bytes sent to the client from the foreground process(es). |
| **Opened cursors current** | The total number of current open cursors. |
| **DB block changes** | Closely related to consistent changes, this statistic counts the total number of changes that were made to all blocks in the SGA that were part of an update or delete operation. These are changes that generate redo log entries and hence cause permanent changes to the database if the transaction is committed. This statistic is a rough indication of total database work and indicates (possibly on a per-transaction level) the rate at which buffers are being dirtied. |
| **Total file opens** | The total number of file opens being performed by the instance. Each process needs a number of files (control file, log file, database file) in order to work against the database. |

## SQL Server Default Counters

| Measurement | Description |
|---|---|
| **% Total Processor Time** | The average percentage of time that all the processors on the system are busy executing non-idle threads. On a multi-processor system, if all processors are always busy, this is 100%, if all processors are 50% busy this is 50% and if 1/4 of the processors are 100% busy this is 25%. It can be viewed as the fraction of the time spent doing useful work. Each processor is assigned an Idle thread in the Idle process which consumes those unproductive processor cycles not used by any other threads. |
| **Cache Hit Ratio** | The percentage of time that a requested data page was found in the data cache (instead of being read from disk). |
| **I/O - Batch Writes/sec** | The number of pages written to disk per second, using Batch I/O. The checkpoint thread is the primary user of Batch I/O. |
| **I/O - Lazy Writes/sec** | The number of pages flushed to disk per second by the Lazy Writer. |
| **I/O - Outstanding Reads** | The number of physical reads pending. |
| **I/O - Outstanding Writes** | The number of physical writes pending. |
| **I/O - Page Reads/sec** | The number of physical page reads per second. |
| **I/O - Transactions/sec** | The number of Transact-SQL command batches executed per second. |
| **User Connections** | The number of open user connections. |
| **% Processor Time** | The percentage of time that the processor is executing a non-idle thread. This counter was designed as a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the idle process in each sample interval, and subtracting that value from 100%. (Each processor has an idle thread which consumes cycles when no other threads are ready to run). It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%. |

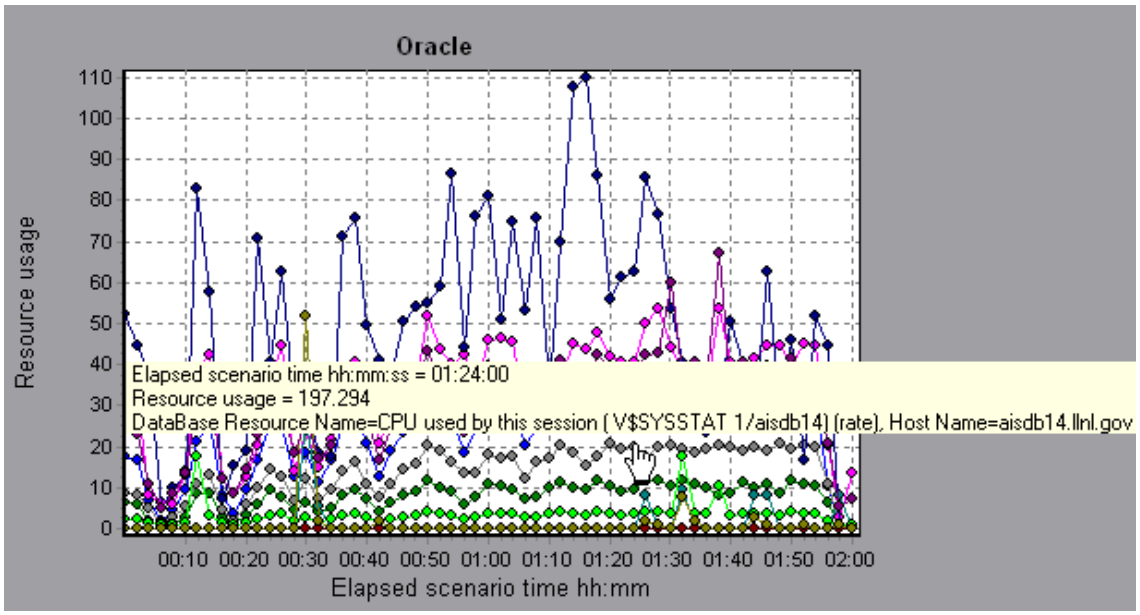
# Oracle Graph

This graph displays information from Oracle V$ tables: Session statistics, V$SESSTAT, system statistics, V$SYSSTAT, and other table counters defined by the user in the custom query.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource usage on the Oracle server. |
| **Note** | To obtain data for this graph, you need to enable the Oracle online monitor (from Controller) and select the default measurements you want to display, before running the scenario. |
| **See also** | "Database Server Resource Graphs" on page 1402<br>"Oracle Server Monitoring Measurements" on page 1402 |

**Example**

In the following example, the V$SYSSTAT resource values are shown as a function of the elapsed load test scenario time:

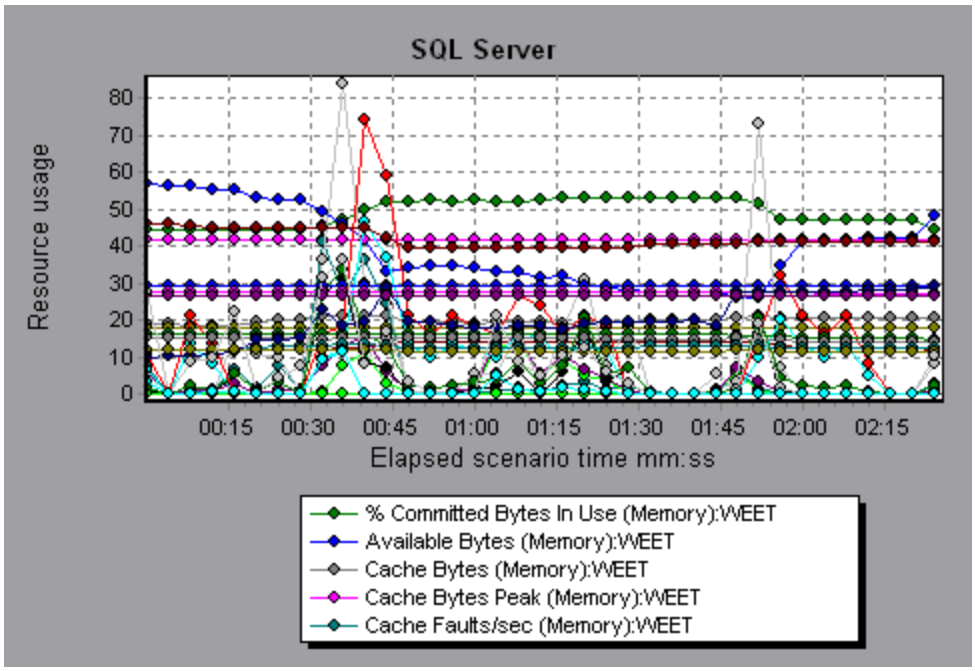



# SQL Server Graph

This graph shows the standard Windows resources on the SQL server machine.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the load test scenario run. |
| **Y-axis** | Resource usage |
| **Note** | To obtain data for this graph, you need to enable the SQL Server online monitor (from Controller) and select the default measurements you want to display, before running the scenario. |
| **See also** | "Database Server Resource Graphs" on page 1402<br>"SQL Server Default Counters" on page 1404 |

**Example**




# Dynatrace Graph

The Dynatrace graph displays metrics monitored by Controller's Dynatrace monitor from the Dynatrace server about monitored applications during a scenario run. For more information, see "Dynatrace Monitor" on page 1189.

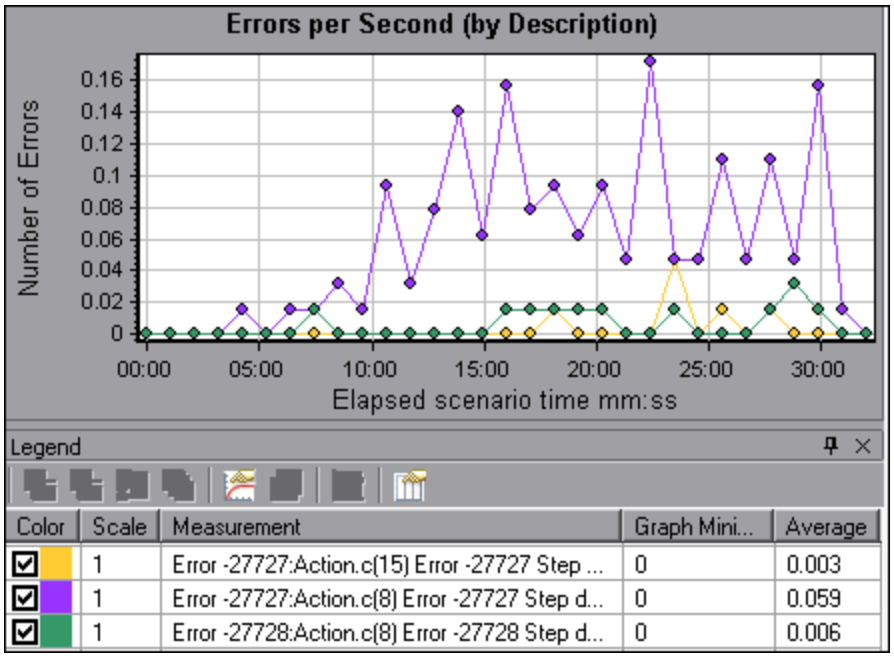
# Error Graphs

## Errors per Second (by Description) Graph

This graph displays the average number of errors that occurred during each second of the load test scenario run, grouped by error description. The error description is displayed in the legend.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of errors. |
| **See also** | "Error Graphs" on the previous page |

**Example**

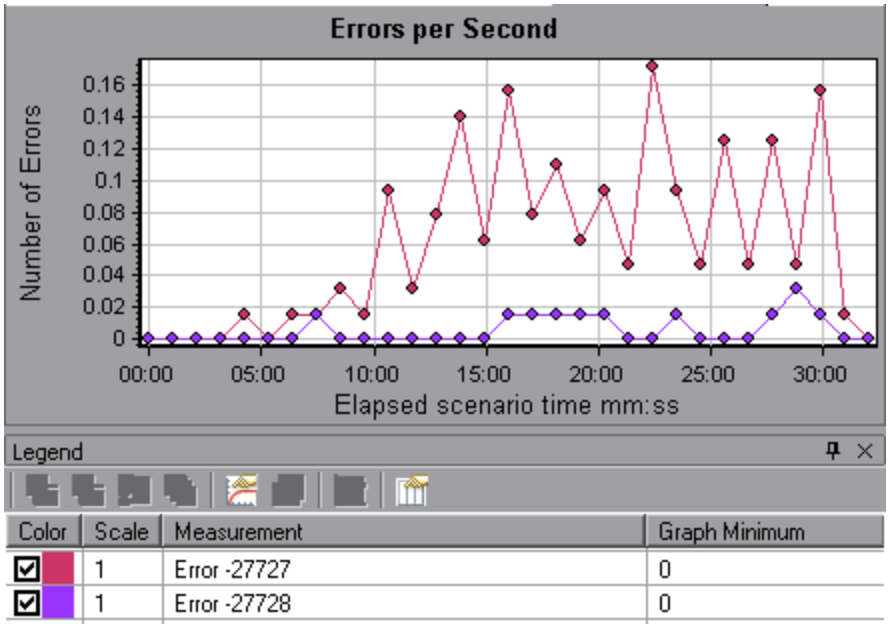



## Errors per Second Graph

This graph displays the average number of errors that occurred during each second of the load test scenario run, grouped by error code.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of errors. |
| **See also** | "Error Graphs" on the previous page |

**Example**

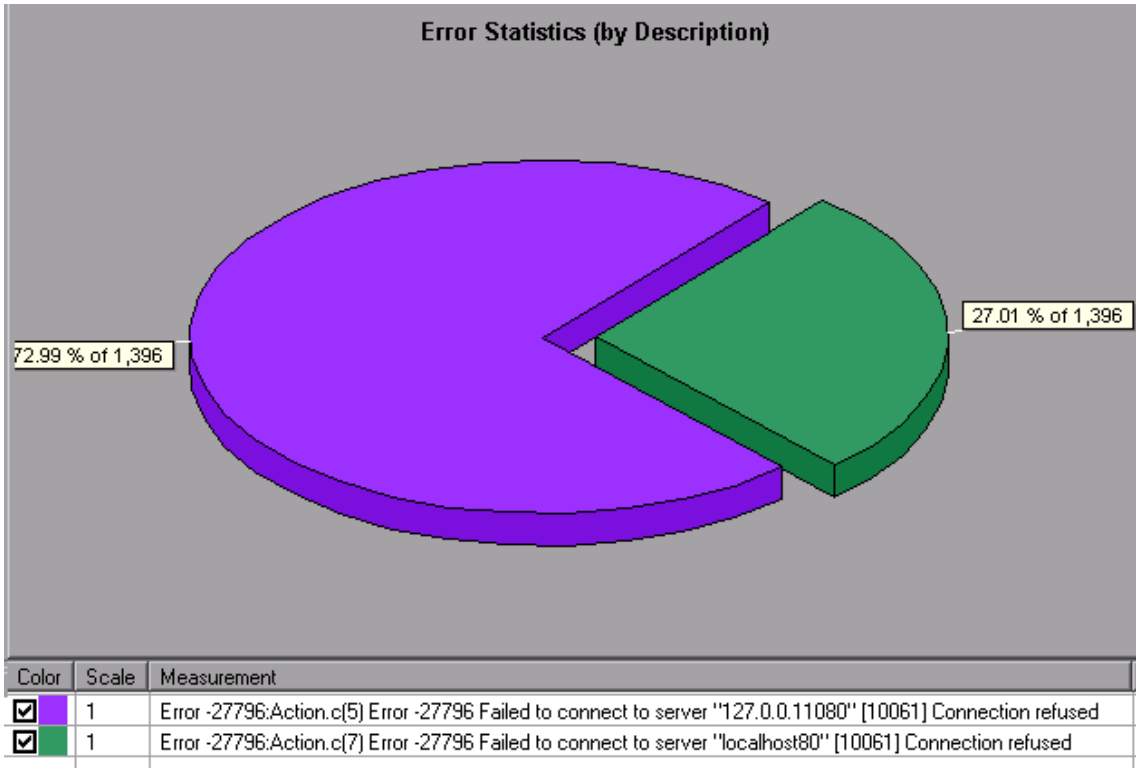



## Error Statistics (by Description) Graph

This graph displays the number of errors that occurred during load test scenario execution, grouped by error description. The error description is displayed in the legend.

| | |
|---|---|
| **Note** | This graph can only be viewed as a pie. |
| **See also** | "Error Graphs" on page 1406 |

**Example**



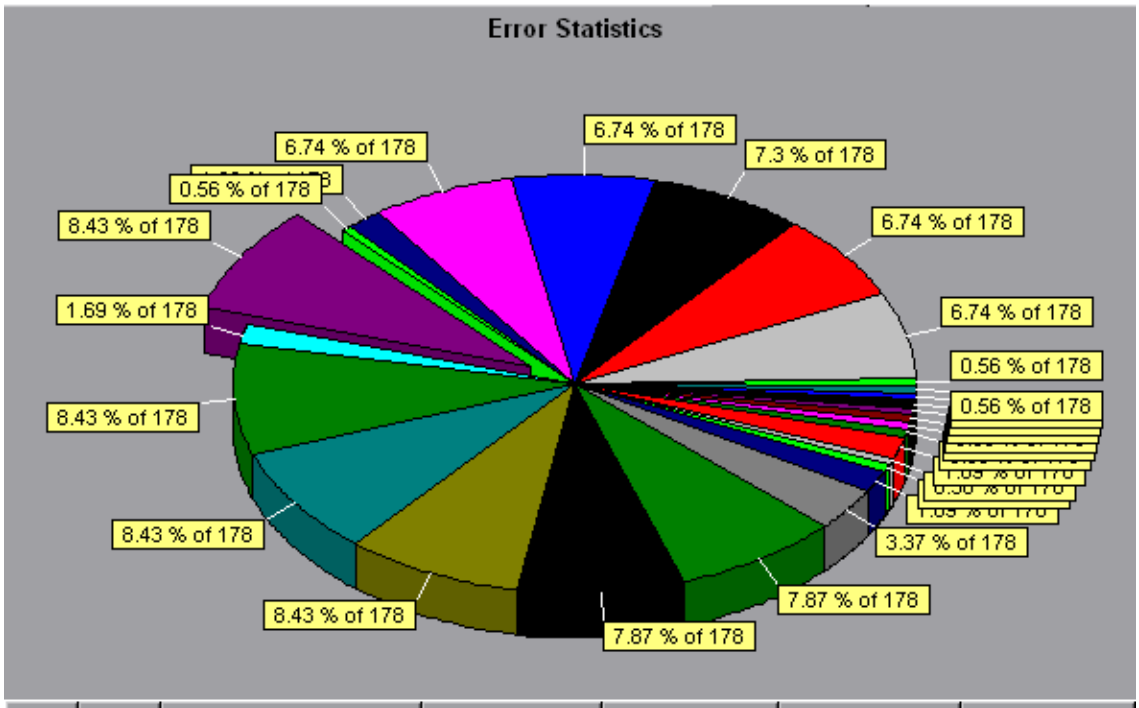

## Error Statistics Graph

This graph displays the number of errors that occurred during load test scenario execution, grouped by error code.

| | |
|---|---|
| **Note** | This graph can only be viewed as a pie. |
| **See also** | "Error Graphs" on page 1406 |

**Example**

In the following example, out of a total of 178 errors that occurred during the scenario run, the second error code displayed in the legend occurred twelve times, comprising 6.74% of the errors.



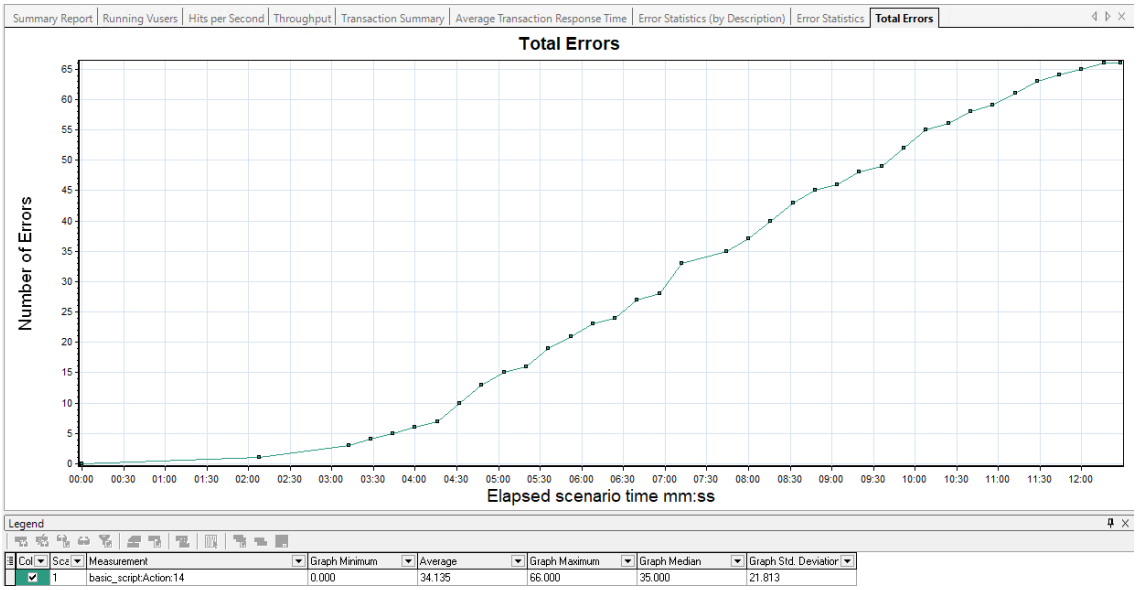

| Color | Scale | Measurement | Graph's Minimum | Graph's Average | Graph's Maximum | Graph's Median |
|---|---|---|---|---|---|---|
| | 1 | Error -26627:-26627 Ac... | 13 | 13 | 13 | 13 |
| | 1 | Error -26627:-26627 Ac... | 12 | 12 | 12 | 12 |
| | 1 | Error -26627:-26627 Ac... | 12 | 12 | 12 | 12 |
| | 1 | Error -26627:-26627 Ac... | 3 | 3 | 3 | 3 |
| | 1 | Error -26627:-26627 Ac... | 1 | 1 | 1 | 1 |

## Total Errors Graph

This graph displays the total number of errors that occurred during the test run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of errors. |
| **See also** | "Error Graphs" on page 1406 |

**Example**

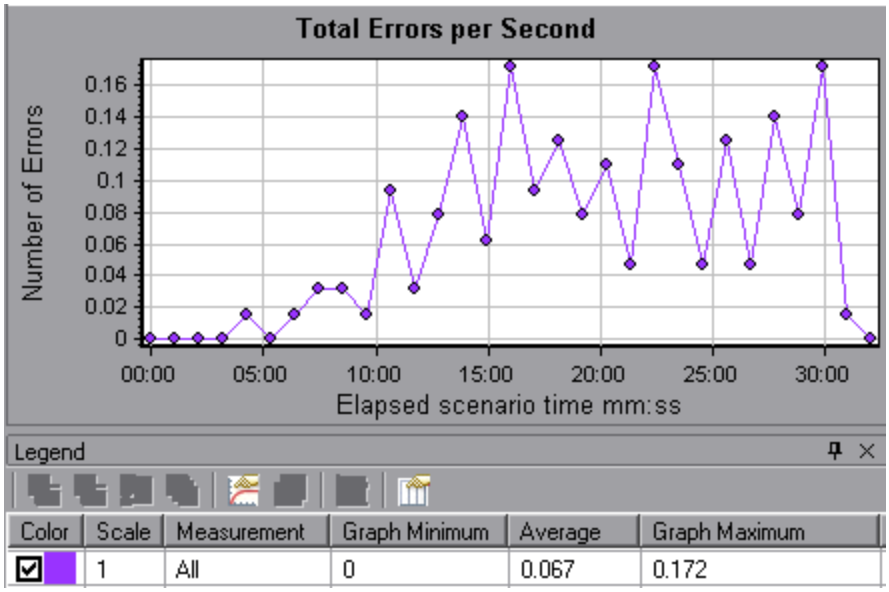



# Total Errors per Second Graph

This graph displays the average number of errors that occurred during each second of the load test scenario run. (complete: add sentence about being sum of all errors)

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of errors. |
| **See also** | "Error Graphs" on page 1406 |

**Example**



# Flex Graphs

Flex graphs provide you with information about the performance of your Flex server.

The Flex graphs are available while monitoring on line with Controller and in Analysis.

You use the Flex graphs to analyze the following data:

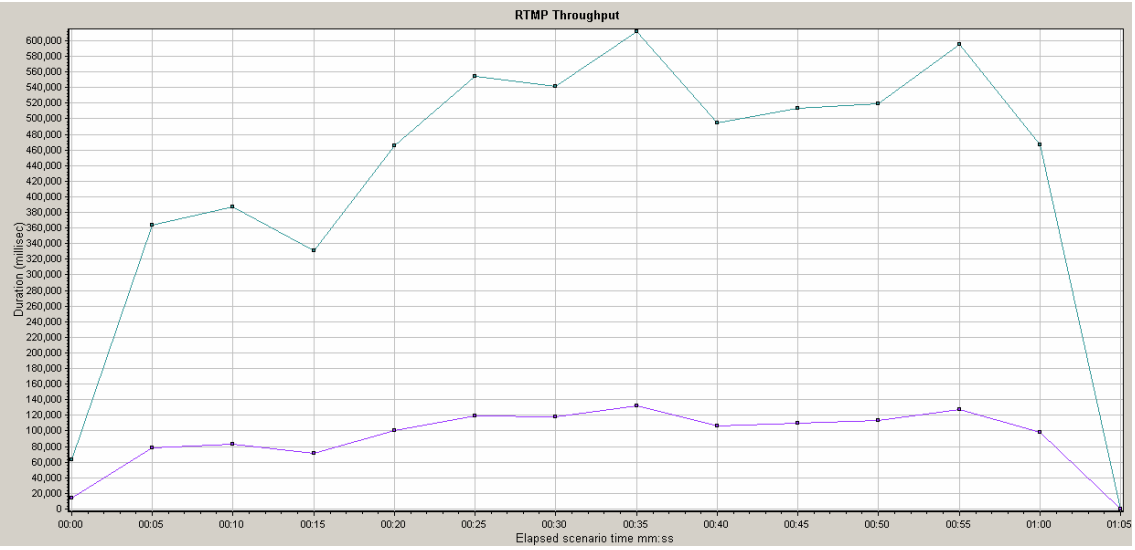
## Flex RTMP Throughput Graph

This graph shows the amount of throughput (in bytes) on the RTMP/T server during each second of the load test scenario run. The throughput represents the amount of data that the Vusers received from the server or sent to the server at any given second.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of load that Vusers generate, in terms of server throughput. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | Throughput of the server in bytes. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |

### Example

In the following example, the highest throughput is over 600,000 bytes during the 35th second of the scenario.



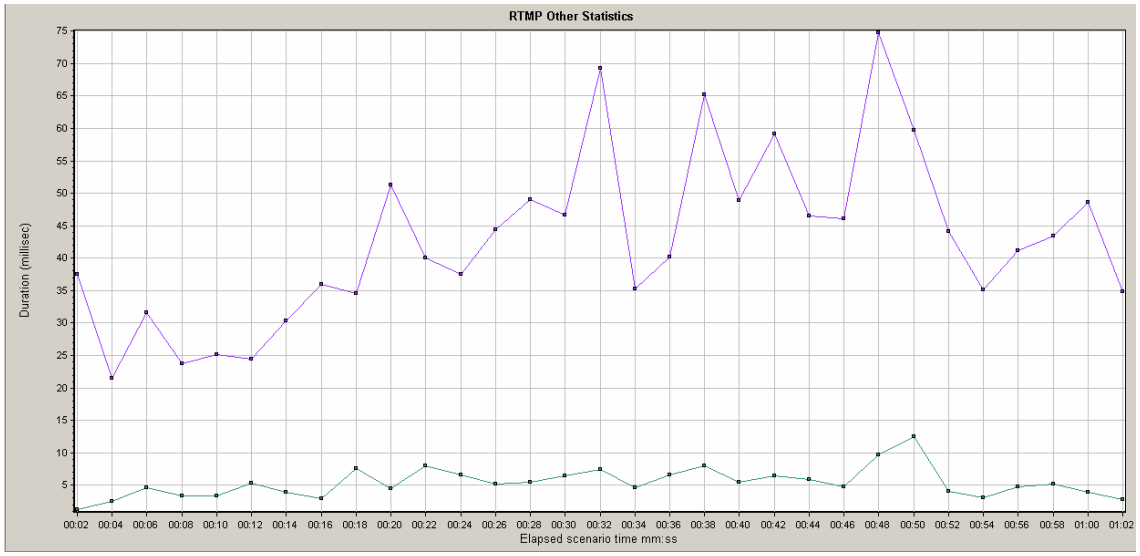
# Flex RTMP Other Statistics Graph

This graph shows various statistics about Flex RTMP Vusers.

| | |
|---|---|
| **Purpose** | The graph shows the duration taken to perform various RTMP tasks. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | Task duration (in milliseconds). |

## Example

In the following example, the RTMP Handshake has a duration of 75 milliseconds at the 48th second of the scenario.



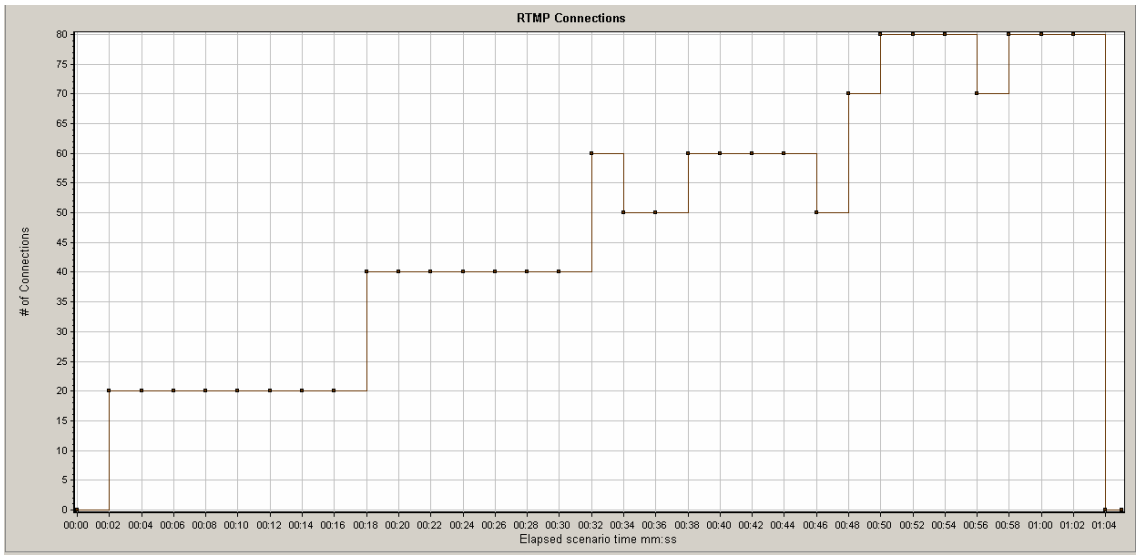

# Flex RTMP Connections Graph

This graph shows the number of open RTMP connections at any time during the load test scenario run. The throughput represents the amount of data that the Vusers received from the server or sent to the server at any given second.

| | |
|---|---|
| **Purpose** | This graph is useful in indicating when additional connections are needed. For example, if the number of connections reaches a plateau, and the transaction response time increases sharply, adding connections would probably cause a dramatic improvement in performance (reduction in the transaction response time). |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | Number of connections. |

### Example

In the following example, between the 50th and the 56th second of the scenario there are 80 open connections.



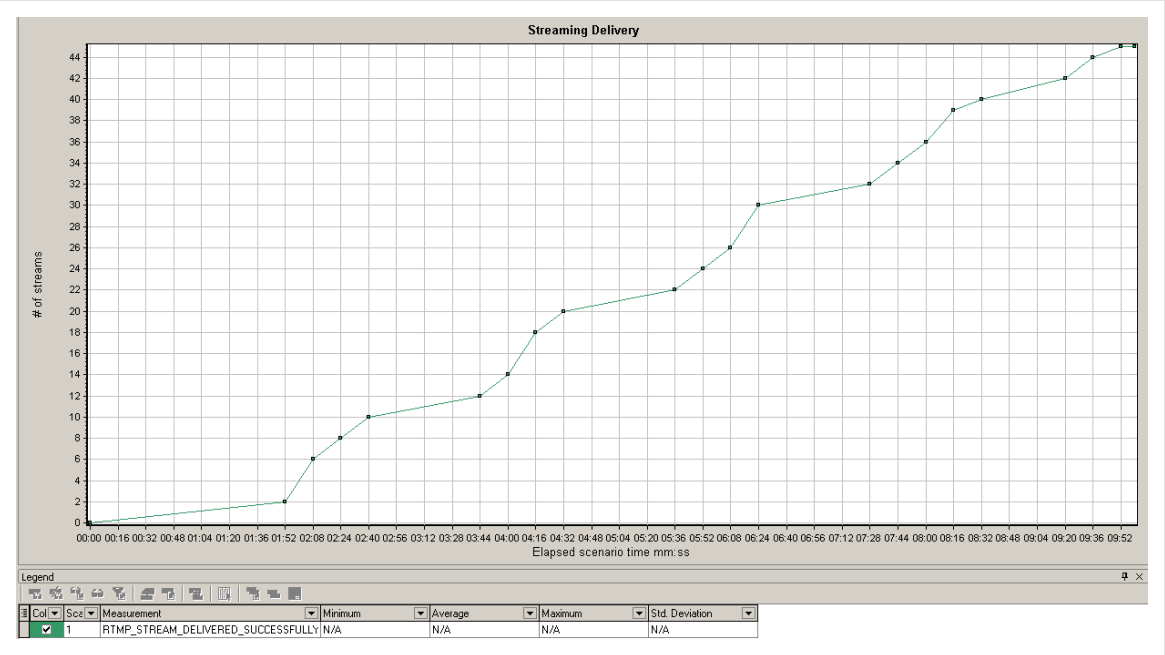

## Flex Streaming Delivery Graph

This graph displays the total number of streams that were successfully delivered by the server. A successful delivery is indicated when the server initiates a **NetStream.Stop** message at the end of the requested stream.

| | |
|---|---|
| **Purpose** | Helps you evaluate the performance of your server, by indicating the number of successfully delivered streams. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | Number of streams delivered over time. |

### Example

In the following example, the graph rises at a 45 degree angle, indicating a constant number of streams being delivered over time.



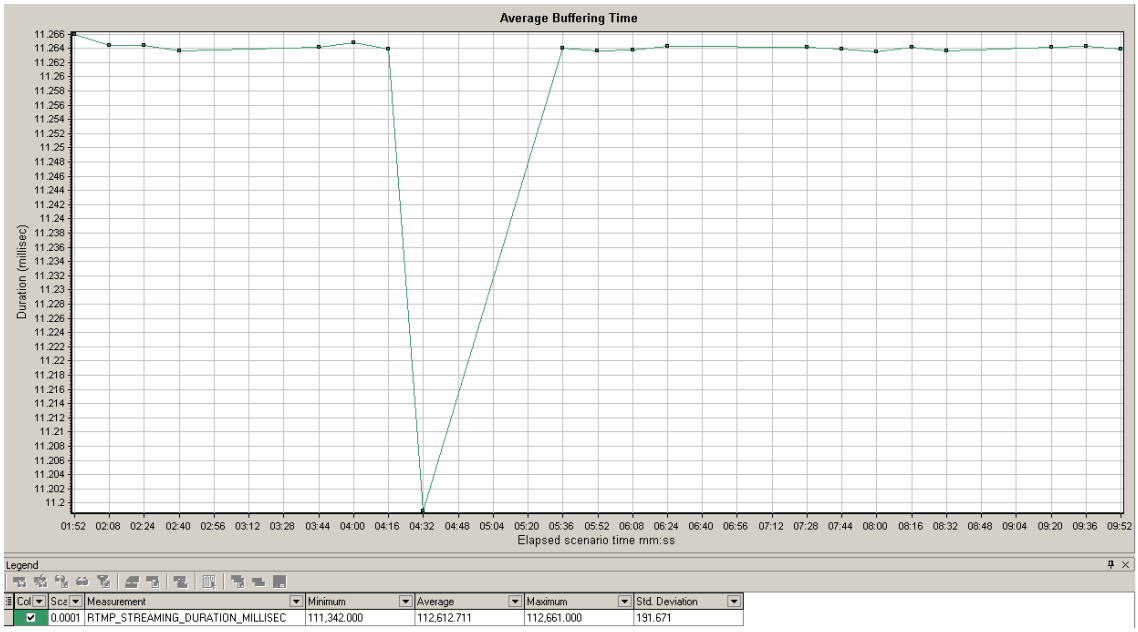

# Flex Average Buffering Time Graph

This graph displays the average buffering time for RTMP streams.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of load that Vusers generate, in terms of time spent for streams in the buffer. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | The buffering time for the RTMP streams in milliseconds. |

## Example

In the following example, the buffering time reaches its lowest after 4 minutes and 32 seconds of the scenario before climbing up to a peak again. Compare this to other graphs to see what caused the reduction at that time.




# Infrastructure Resources Graphs

LoadRunner's Infrastructure Resources monitor provides you with information about the performance of FTP, POP3, SMTP, IMAP, and DNS Vusers on the network client during load test scenario execution.

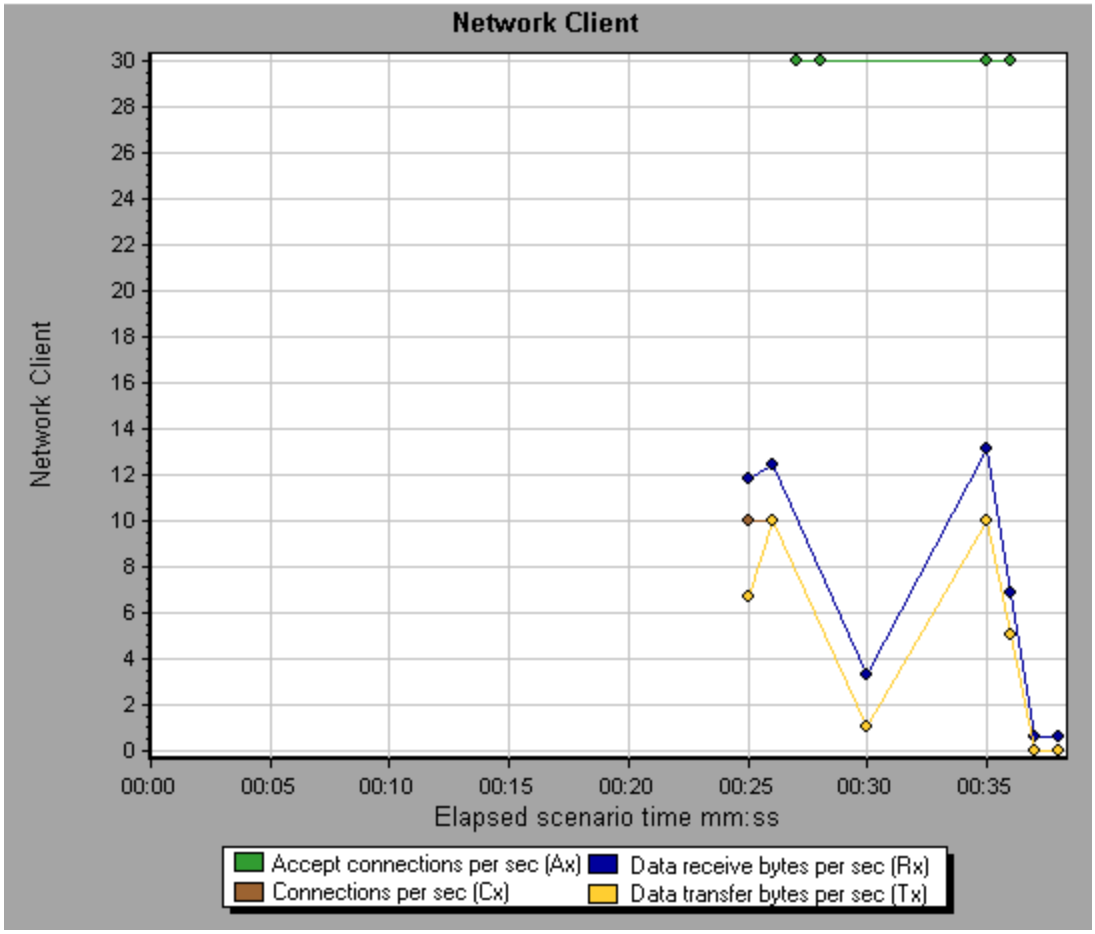
## Network Client Measurements

| Measurement | Description |
|---|---|
| **Pings per sec** | Number of pings per second. |
| **Data transfer bytes per sec** | Number of data bytes transferred per second. |
| **Data receive bytes per sec** | Number of data bytes received per second. |
| **Connections per sec** | Number of connections per second. |
| **Accept connections per sec** | Number of connections accepted per seconds. |
| **SSL Connections per sec** | Number of SSL connections per second. |
| **SSL Data transfer bytes per sec** | Number of SSL data bytes transferred per second. |
| **SSL Data receive bytes per sec** | Number of SSL data bytes received per second. |
| **SSL Accept connections per sec** | Number of SSL connections accepted per seconds. |

## Network Client Graph

This graph displays network client data points for FTP, POP3, SMTP, IMAP, and DNS Vusers during a load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource value of the network client data points.. |
| **See also** | "Infrastructure Resources Graphs" on the previous page |




# J2EE and .NET Diagnostics Graphs

## J2EE and .NET Diagnostics Graphs Overview

The J2EE/.NET Diagnostics graphs in LoadRunner Analysis enable you to trace, time, and troubleshoot individual transactions and server requests through J2EE & .NET Web, application, and database

servers. You can also pinpoint problem servlets and JDBC calls to maximize business process performance, scalability, and efficiency.

The J2EE/.NET Diagnostics graphs are comprised of two groups:

- **J2EE & .NET Diagnostics Graphs.** These graphs show you the performance of requests and methods generated by virtual user transactions. They show you the transaction that generated each request.
- **J2EE & .NET Server Diagnostics Graphs.** These graphs show you the performance of all the requests and methods in the application you are monitoring. These include requests generated by virtual user transactions and by real users.

## How to Enable Diagnostics for J2EE and .NET

To generate diagnostics data for J2EE and .NET, you must first install J2EE/.NET Diagnostics.

Before you can use J2EE/.NET Diagnostics with LoadRunner, you need to ensure that you have specified the Diagnostics Server details in LoadRunner. Before you can view J2EE/.NET Diagnostics data in a particular load test scenario, you need to configure the Diagnostics parameters for that scenario. For more information, see the section on online monitors in the LoadRunner Controller documentation.

> **Note:** To ensure that valid J2EE/.NET Diagnostics data is generated during the scenario run, you must manually mark the beginning and end of each transaction in the Vuser script, rather than using automatic transactions.

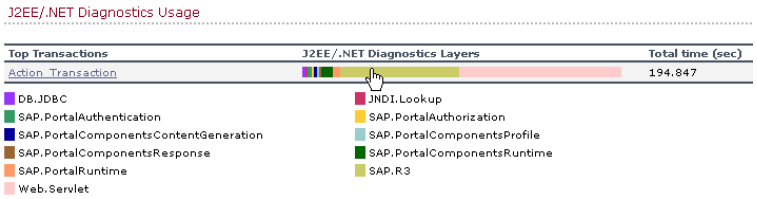
## Viewing J2EE to SAP R3 Remote Calls

The *Remote Function Call* (RFC) protocol in SAP allows communication to take place between SAP J2EE and SAP R3 environments. When remote calls take place between SAP J2EE and SAP R3 environments, Analysis displays information about the RFC functions, including the name of each function.

You view information about RFC functions by breaking down the SAP R3 layer. You can view the RFC function information in a graph display or in the Chain Of Calls window.

### View RFC function information in a graph display

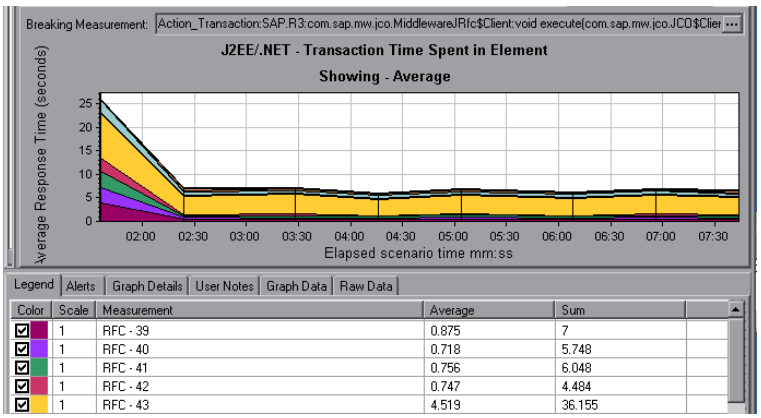
1. Go to the **J2EE/.NET Diagnostics Usage** section of the Summary Report. Next to the relevant transaction, click the color representing the **SAP.R3** layer.




The J2EE/.NET - Transaction Time Spent in Element graph opens, representing the SAP.R3 layer.

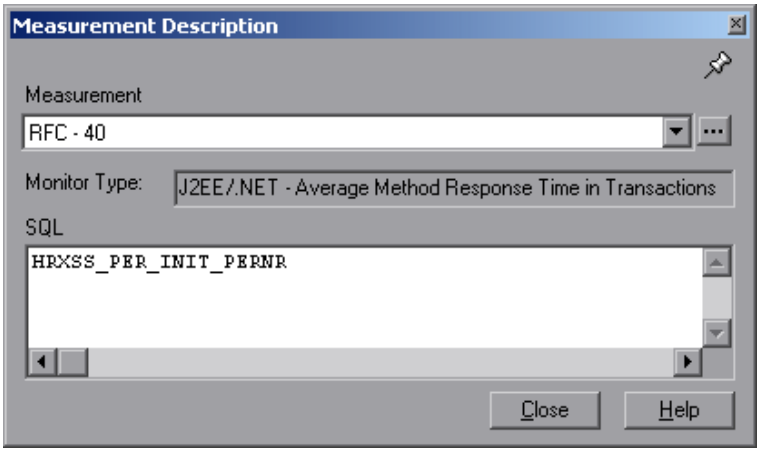
2. Right click the graph and choose **J2EE/.NET Diagnostics > Break down the class to methods**.
3. Break down the graph further by right clicking the graph and choosing **J2EE/.NET Diagnostics > Break down the method to SQLs**.

   The graph is broken down into the different RFC functions.




4. To view the name of each RFC function, right click an RFC measurement in the **Measurement** column in the graph legend and choose **Show measurement description**.

   The Measurement Description dialog box opens. The name of the RFC function is displayed in the **SQL** box.




## View RFC function information in the Chain Of Calls window

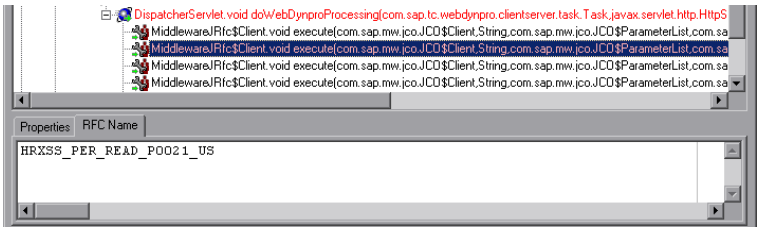
1. Go to the **J2EE/.NET Diagnostics Usage** section of the Summary Report. Next to the relevant transaction, click the color representing the **SAP.R3** layer.

   The J2EE/.NET - Transaction Time Spent in Element graph opens, representing the SAP.R3 layer.

2. Right click the graph and choose **J2EE/.NET Diagnostics > Show chain of calls**.

   The Transaction chain of calls window opens. When you click any of the RFC functions, in the

**Measurement** column, the name of the function is displayed in the lower pane in the **RFC Name** tab.




## J2EE and .NET Diagnostics Data

The J2EE and .NET Diagnostics graphs provide an overview of the entire chain of activity on the server side of the system. At the same time, you can break down J2EE/.NET layers into classes and methods to enable you to pinpoint the exact location where time is consumed. In addition, you can view custom classes or packages that you set the J2EE/.NET probe to monitor. You can also view the transaction chain of calls and call stack statistics to track the percentage of time spent on each part of the transaction.

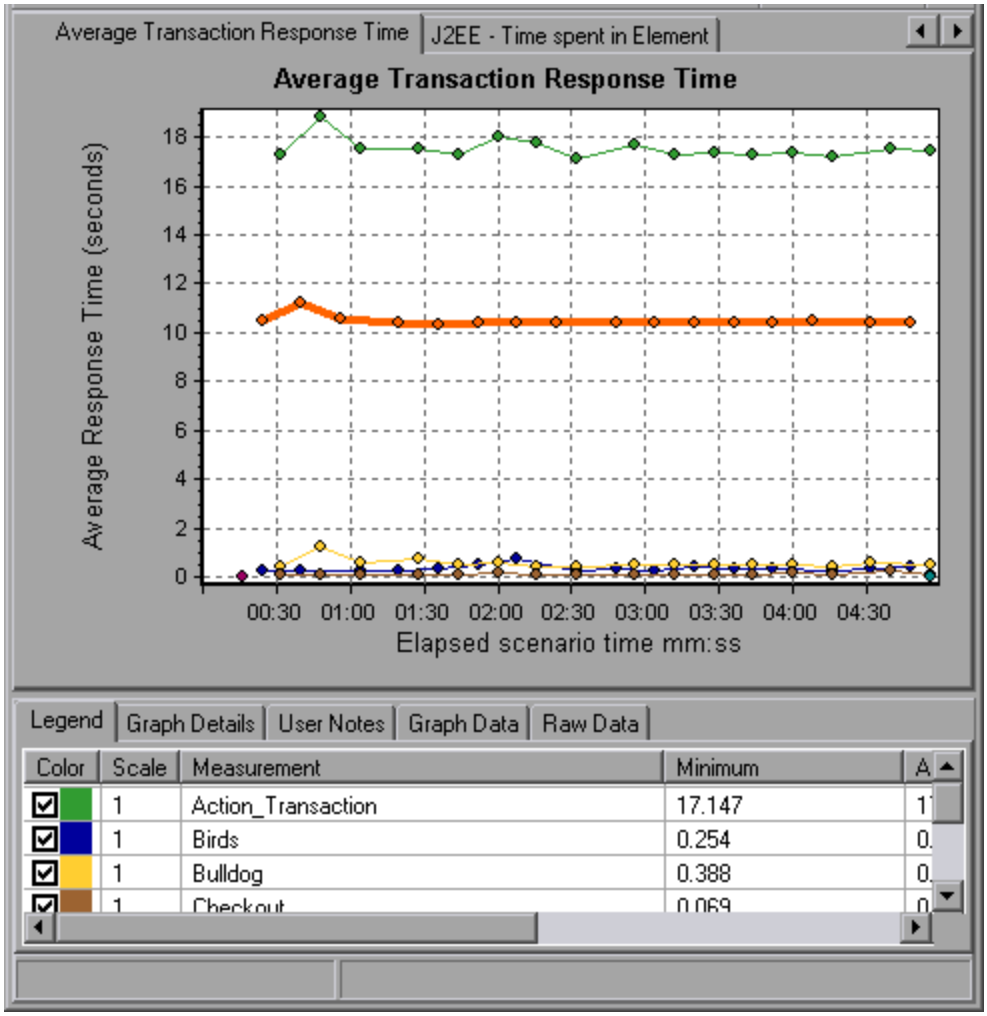
You can correlate the end user response time with the Web server activity (Servlets and JSPs data), application server activity (JNDIs), and back-end activity of database requests (JDBC methods and SQL queries).

## Example Transaction Breakdown

The following graphs illustrate the breakdown of a transaction to its layers, classes, and methods.

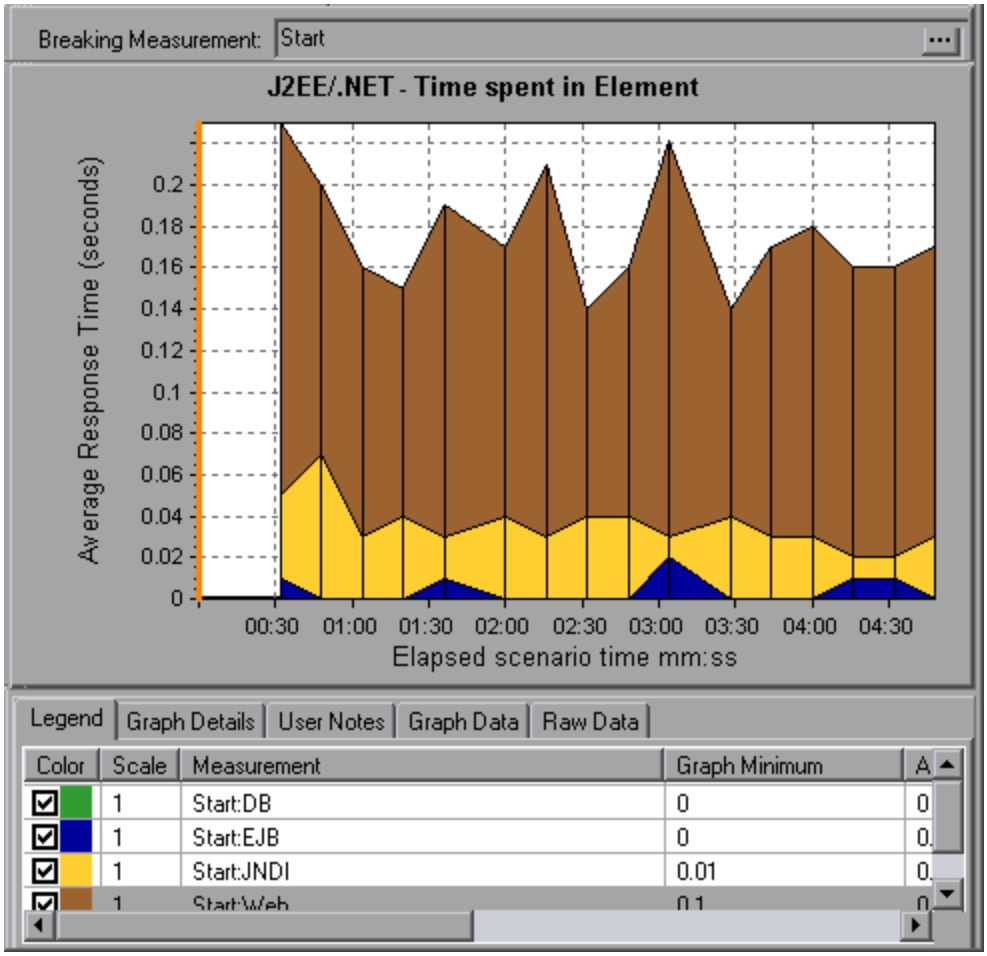
### Transaction Level

The following figure shows the top level Average Transaction Response Time graph. The graph displays several transactions: **Birds**, **Bulldog**, **Checkout**, **Start**, and so on.



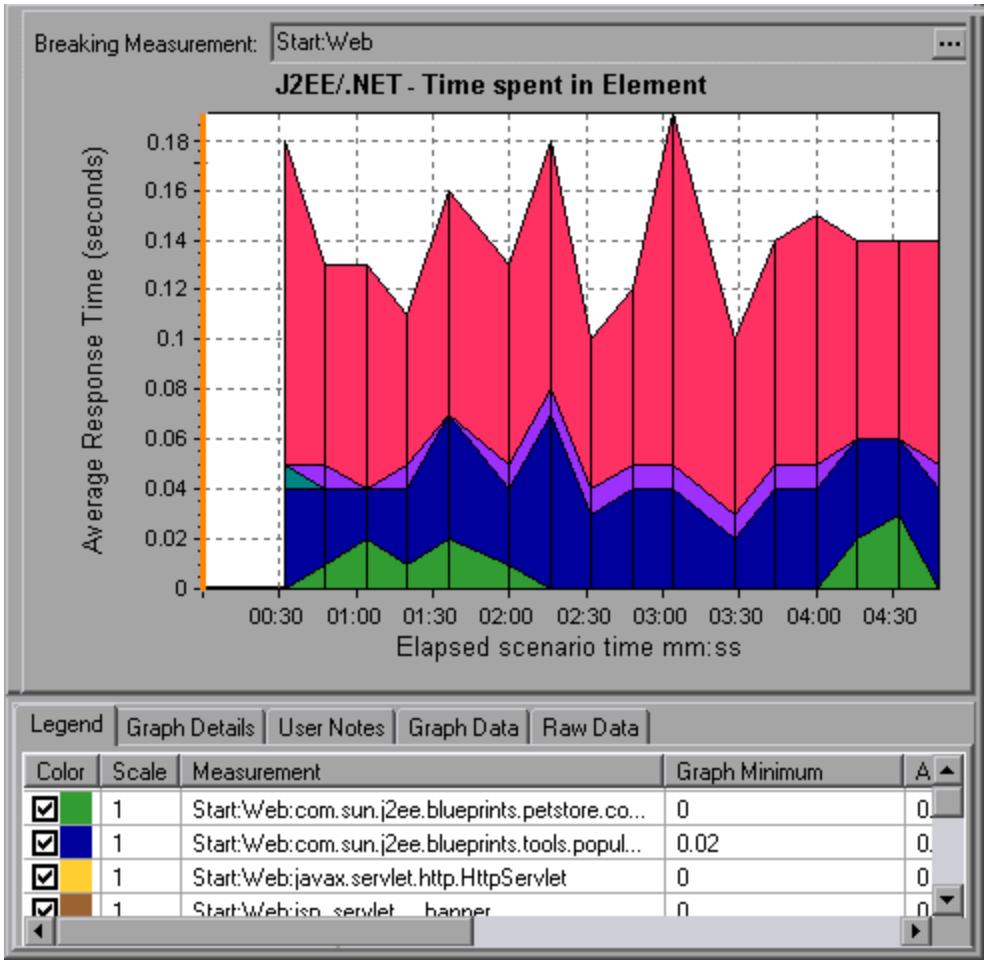

## Layer Level

In the following figure, the **Start** transaction has been broken down to its layers (DB, EJB, JNDI, and Web). In J2EE/.NET transactions, the Web layer is generally the largest.



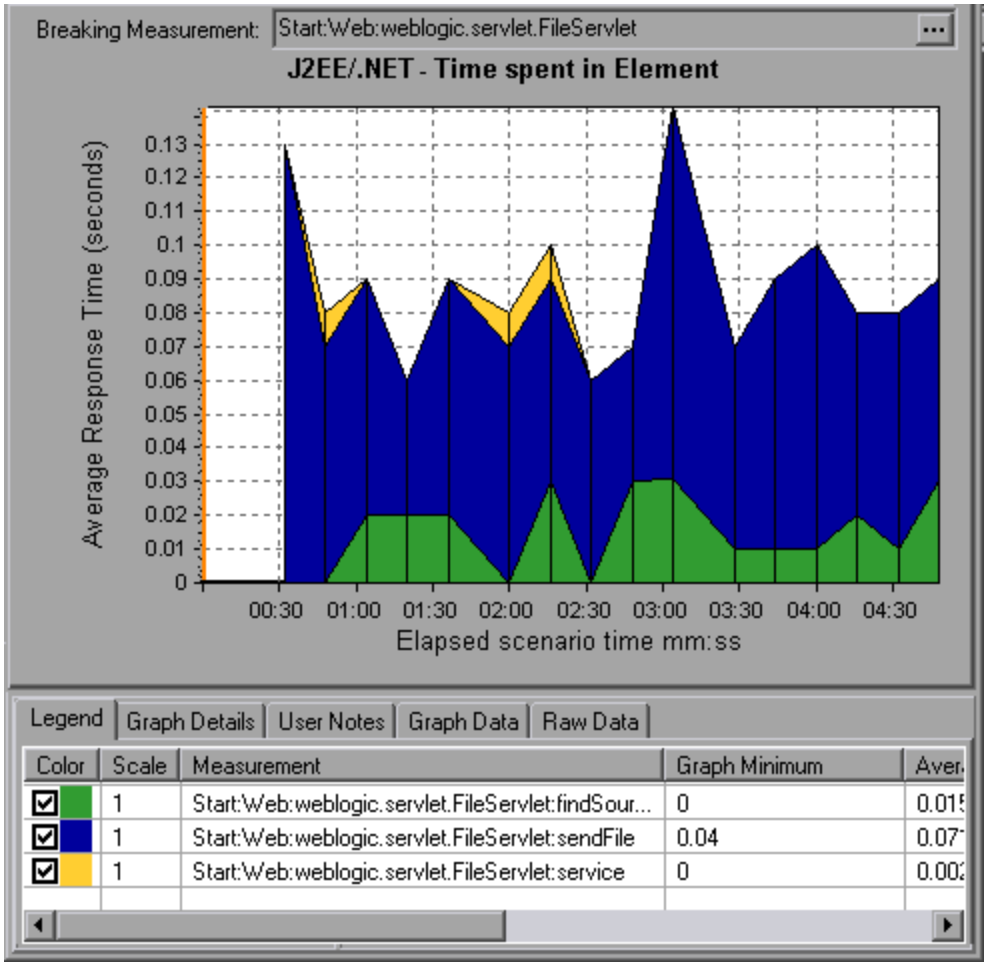

## Class Level

In the following figure, the Web layer of the **Start** transaction has been broken down to its classes.




## Method/Query Level

In the following figure, the **weblogic.servlet.FileServlet** component of the **Web** layer of the **Start** transaction has been broken down to its methods.




**Note:** Some JDBC methods can invoke SQLs which can be broken down further. In this case there is another level of breakdown, that is SQL Statements. For the methods that cannot be further broken down into SQL statements when reaching this level of breakdown, you see **NoSql**.

## Cross VM Analysis

When a server request makes a remote method invocation, the J2EE & .NET Diagnostics graphs display certain measurements relating to the classes and methods involved in these requests. These measurements are displayed at a layer, class and method level. The VM making the call is referred to as the *caller VM*, and the VM that executes the remote call is the *callee VM*.

### Descriptions of the measurements

| Measurements | Description |
|---|---|
| **Cross VM Layer** | A measurement that represents a dummy layer that integrates the data from the remote classes and methods in server requests that take place across two or more virtual machines. |
| **Remote-Class** | A measurement that represents a dummy class that integrates the data from the remote methods in server requests that take place across two or more virtual machines. |
| **Remote-Class: Remote Method** | A measurement that represents a dummy method. Remote-Class: Remote Method measures the total time, call count, exclusive latency, minimum and maximum values, standard deviation, and so on of the methods that are executed remotely, relative to the caller virtual machine. |

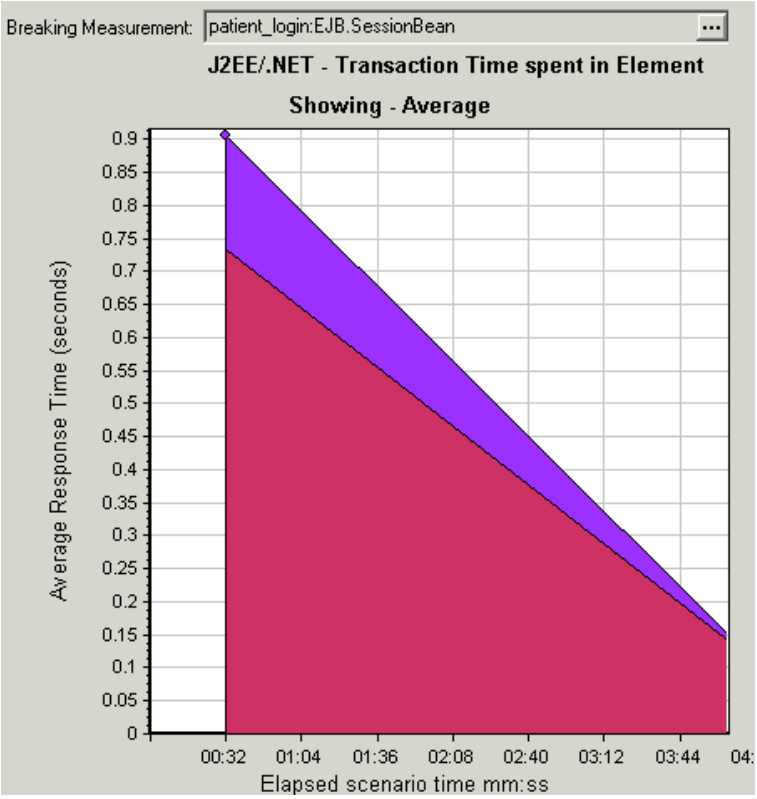
**Note:** Since this data is measured on the caller virtual machine the exclusive latency will include all of the time required for making the remote method invocation such as network latency.

## Using the J2EE and .NET Breakdown Options

J2EE and .NET breakdown options are described.

| | |
|---|---|
| **To access** | Use one of the following to access breakdown options:<br>• **<J2EE & .NET Graphs> View > J2EE & .NET Diagnostics**<br>• **<J2EE & .NET Diagnostics Graphs> >** select transaction **>** short-cut menu **> J2EE & .NET Diagnostics**<br>• See toolbar options for each breakdown level |
| **Notes** | • The breakdown menu options and buttons are not displayed until an element (transaction, server request, layer) is selected.<br>• If there is no URI in the SQL, URI-None appears in front of the full measurement description in the Measurement Description dialog box. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

User interface elements are described below :

| UI Element | Description |
|---|---|
| **<Right-click> transaction in Average Response Time Graph** | Choose J2EE/.NET Diagnostics > Show Server Requests. A new graph opens showing the breakdown of the selected transaction. The name of the transaction is displayed in the Breaking Measurement box.<br>  <br>You can view the full SQL statement for a selected SQL element by choosing **Show measurement description** from the Legend window right-click menu. The Measurement Description dialog box opens displaying the name of the selected measurement and the full SQL statement. |
| | To view transaction properties for the breakdown measurement, click the **Breaking Measurement** button. To disable this feature, choose **View > Display Options**, and clear the **Show Breaking Measurement** check box. |
| | Select **View > J2EE/.NET Diagnostics > Break down the server request to layers**, or click the measurement breakdown button in the toolbar above the graph.<br>**Note**: The option in the J2EE/.NET Diagnostics menu, and the tool tip of the measurement breakdown button, vary according to the element that you want to break down. For example, when you select a server request, the menu option and tool tip are **Break down server request to layers**. |

| UI Element | Description |
|---|---|
| | Select **View > J2EE/.NET Diagnostics > Show VM**, or click the **Show VM** button in the toolbar above the graph. This breaks the data down to the application host name (VM). |
| | Select **View > J2EE/.NET Diagnostics > Undo Break down the server request to layers**, or click the **Undo <***Measurement Breakdown***>** button in the toolbar above the graph.<br><br>**Note:** The option in the J2EE/.NET Diagnostics menu, and the tool tip of the measurement breakdown button, vary according to the element whose breakdown you want to undo. For example, when you select a layer, the menu option and tool tip are Undo break down server request to layers. |
| | Select **View > J2EE/.NET Diagnostics > Hide VM**, or click the **Hide VM** button in the toolbar above the graph. |
| | Display the chain of call or call stack statistics in the measurements tree window: Drag the orange time line on to the graph to the time specifying the end of the period for which you want to view data, and select<br>**View > J2EE/.NET Diagnostics > Show Chain of Calls**, or click the Show Chain of Calls button in the toolbar above the graph.<br><br>**Note:** A measurement that is broken down in the Average Method Response Time in Transactions graph will be different from the same measurement broken down in the J2EE/.NET - Transaction Time Spent in Element graph. This is because the J2EE/.NET - Average Method Response Time in Transactions graph displays the average transaction time, whereas the J2EE/.NET - Transaction Time Spent in Element graph displays the average time per transaction event (sum of method execution time). |

## Viewing Chain of Calls and Call Stack Statistics

You can view the chain of calls for transactions and methods. The chain of calls answers the question "Whom did I call?"

You can also view the call stack statistics for methods. Call stack statistics answer the question "Who called me?"

### The Chain of Calls window

Chain of call and call stack statistics data are shown in the measurements tree window. The title of the window changes depending on which kind of data you are viewing.

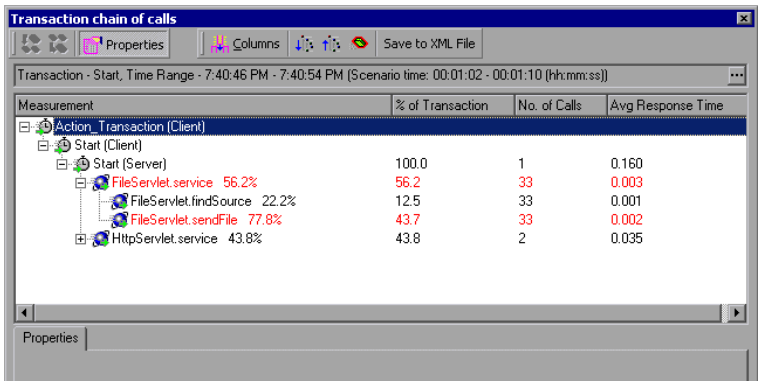
- To set the point to which the measurements tree window relates, drag the orange time line to the desired spot.
- To view transaction call chains, right-click a component and choose **J2EE/.NET Diagnostics > Show Chain of Calls**. The Chain of Calls window opens displaying the chain

of calls from the parent transaction downwards.

- To view method statistics, in the Chain of Calls window right-click a method and choose **Show Method Chain of Calls** or **Show Method Call Stack Statistics**.

You use the Chain of Calls window to view the components that the selected transaction or method called.

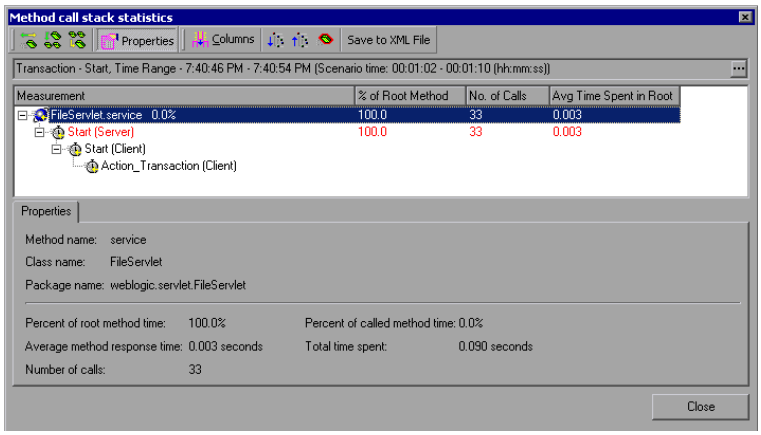
In the following figure, all the calls in the critical path of the Start server-side transaction are displayed.




**Note:** Each red node signifies the most time consuming child of its parent.

You use the Call Stack Statistics window to view which components called the selected component.

In the following figure, the **FileServlet.service** was called by Start (Server), which was called by Start (Client), and so on, down to the transaction at the bottom of the chain.




## Icons in the Chain of Calls window

User interface elements are described below:

| UI Element | Description |
|---|---|
| | **Switch to Method Chain of Calls.** When the call stack statistics data is displayed, displays the method chain of calls data (only if the root is a method). |
| | **Switch to Method Call Stack Statistics.** When the method chain of calls data is displayed, displays the method call stack statistics data (only if the root is a method). |
| | **Show Method Chain of Calls.** Displays the Chain of Calls window. |
| | **Show Method Call Stack Statistics.** Displays the Call Stack Statistics window. |
| | **Properties.** Hides or displays the properties area (lower pane). |
| | **Columns.** Enables you to select the columns shown in the Calls window. To display additional fields, drag them to the desired location in the Calls window. To remove fields, drag them from the Calls window back to the Columns chooser. |
| | **Expand All.** Expands the entire tree. |
| | **Collapse All.** Collapses the entire tree. |
| | **Expand Worst Path.** Expands only the parts of the path on the critical path. |
| **Save to XML File** | Saves the tree data to an XML file. |
| **Method Properties** | Area**.** Displays the full properties of the selected method. |
| **SQL Query** | Displays the SQL query for the selected method. (For Database only.)The following columns are available in the Chain of Calls window: |

## Columns in the Chain of Calls window

The following columns are available in the Chain of Calls window:

| Column | Description |
|---|---|
| **Measurement** | Name of the method, displayed as **ComponentName:MethodName**. In the case of a database call, query information is also displayed. The percent shown indicates the percentage of calls to this component from its parent. |

| Column | Description |
|---|---|
| **% of Root Method** | Percentage of the total time of the method from the total time of the root tree item. |
| **No of Calls** | Displays the amount of times this transaction or method was executed. |
| **Avg Response Time** | Response time is the time from the beginning of execution until the end. Average response time is the total response time divided by the number of divided by the number of instances of the method. |
| **STD Response Time** | The standard deviation of the response times. |
| **Min Response Time** | The minimum response time. |
| **Max Response Time** | The maximum response time. |
| **% of Caller** | Displays the percentage of method time in relation the parent method time. |
| **Total time** | Displays the total method execution time, including the child execution time. |

## Columns in the Call Stack Statistics window

The following columns are available in the Call Stack Statistics window:

| Column | Description |
|---|---|
| **Measurement** | Name of the method, displayed as **ComponentName.MethodName**. In the case of a database call, query information is also displayed. The percent shown indicates the percentage of calls to this component from its child. |
| **% of Root Method** | Percentage of the total time of the transaction (or method) from the total time of the root tree item. |
| **No. of Calls to Root** | Displays the amount of times this transaction or method was executed. |
| **Avg Time Spent in Root** | Time spent in root is the time that the sub-area spent in the root sub-area/area/transaction.<br>Average Time Spent in Root time is the total time spent in the root divided by the number of instances of the method. |

| Column | Description |
|---|---|
| **STD Time Spent in Root** | The standard deviation of the durations of time spent in the root. |
| **Min Time Spent in Root** | The minimum time spent in the root. |
| **Max Time Spent in Root** | The maximum time spent in the root. |
| **% of Called** | Displays the percentage of method time in relation the child method time. |
| **Total Time Spent in Root** | Displays the total method execution time, including the child execution time. |

## Graph Filter Properties

You can filter the J2EE & .NET Diagnostics graphs so that the displayed data is more suitable to your needs. You can filter using the following methods:

- Before opening a graph, enter filter criteria in the **Graph Properties** box of the **Open Graph** dialog box. For more information, see "Open a New Graph Dialog Box" on page 1371.
- From an open graph, enter filter criteria in the **Filter condition** fields in a filter dialog box. For more information, see "Filter Dialog Boxes" on page 1351 and "Drilling Down in a Graph" on page 1328.

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Class Name** | Shows data for specified classes. |
| **Layer Name** | Shows data for specified layers. |
| **Scenario Elapsed Time** | Shows data for transactions that ended during the specified time. |
| **SQL Logical Name** | Shows data for specified SQL logical names. Due to the length of some SQL names, after you choose an SQL statement it is assigned a "logical name." This logical name is used in the filter dialog, legend, grouping, and other places in place of the full SQL statement. You can view the full SQL statement in the Measurement Description dialog box (**View > Show Measurement Description**). |
| **Transaction Name - J2EE/.NET** | Shows data for a specified transaction. |

Some JDBC methods have the ability to invoke SQL's (each method can invoke several different SQL's) so there is another level of breakdown which is the SQL statements.

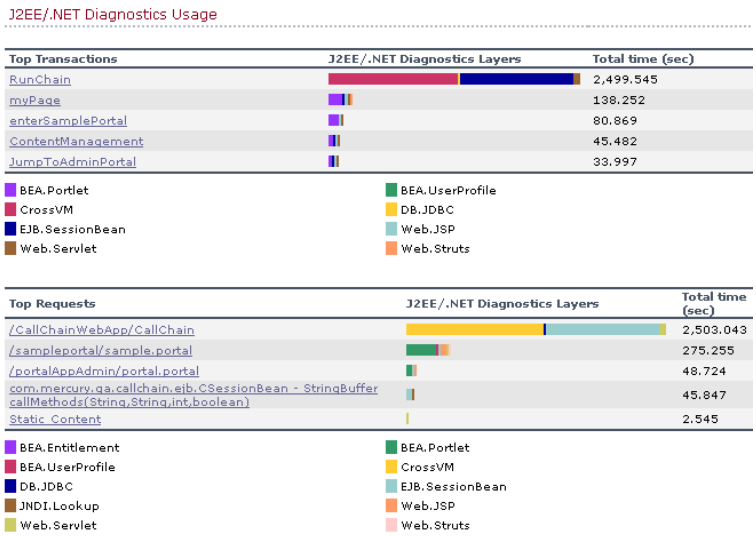
> **Note:** For the methods that do not have SQL statement when reaching this level of breakdown you see **NoSql**.

## J2EE and .NET Diagnostics Graphs Summary Report

The J2EE & .NET Diagnostics Usage section of the Summary report provides general information about load test scenario execution and a usage chart for the J2EE & .NET Diagnostics and server request layers.

| | |
|---|---|
| **Breakdown options** | The J2EE & .NET Diagnostics Usage section breaks transactions and server requests into:<br>• Web server activity (Servlets and JSPs data)<br>• Application server activity (JNDIs)<br>• Back-end activity of database requests (JDBC methods and SQL queries)<br>• Total usage time for each transaction and request |
| **Tips** | **Click on transaction**<br>In the J2EE/.NET Diagnostics Usage section of the Summary Report, click the transaction or J2EE /.NET layer on which you want to perform breakdown. The J2EE/.NET - Transaction Time Spent in Element graph or J2EE/.NET - Server Request Time Spent in Element graph opens.<br>**Click on layer**<br>Clicking a layer displays the specific layer breakdown in the transaction or server request. For more information, see "J2EE/.NET - Transaction Time Spent in Element Graph" on page 1445 and "J2EE/.NET - Average Number of Timeouts on Server Graph" on page 1436. |
| **Note** | If you do not see diagnostics data on the Summary Report, check if you are using a user-defined template. To view relevant data, choose a different template from the list of templates, or create and apply a new template. For more information about using templates, see "Apply/Edit Template Dialog Box" on page 1323. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

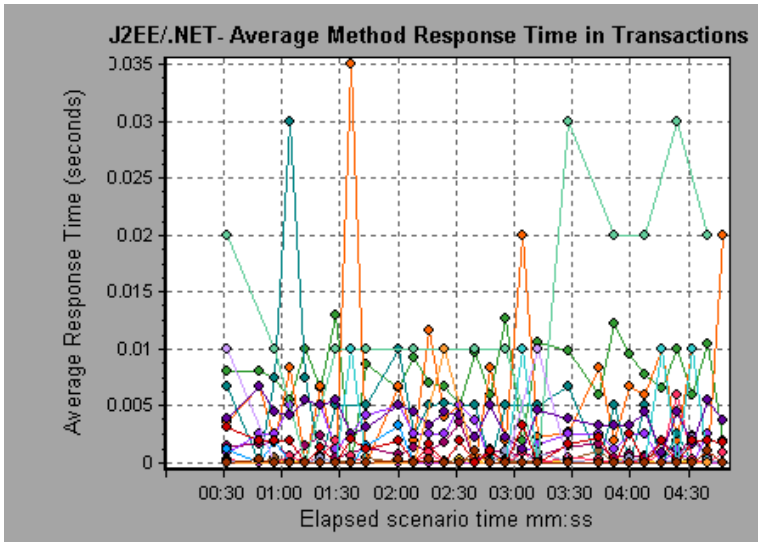



# J2EE/.NET - Average Method Response Time in Transactions Graph

This graph displays the average response time for the server side methods, computed as Total Method Response Time/Number of Method calls. For example, if a method was executed twice by an instance of transaction A and once by another instance of the same transaction, and it took three seconds for each execution, the average response time is 9/3, or 3 seconds. The method time does not include calls made from the method to other methods.

| | |
|---|---|
| **X-axis** | Elapsed time. |
| **Y-axis** | Average response time (in seconds) per method |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

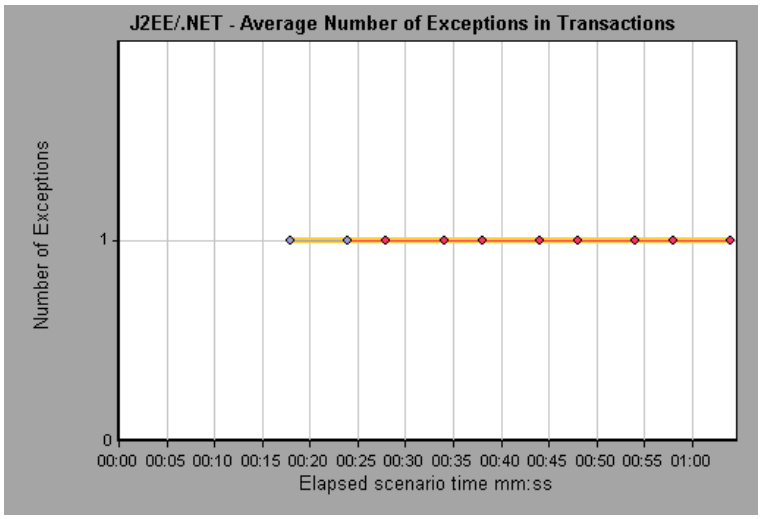



## J2EE/.NET - Average Number of Exceptions in Transactions Graph

This graph displays the average number of code exceptions per method, transaction, or request that were monitored during the selected time range.

| | |
|---|---|
| **X-axis** | Elapsed time. |
| **Y-axis** | Represents the number of events. |
| **Breakdown options** | To break the displayed elements down further, see "Using the J2EE and .NET Breakdown Options" on page 1425. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

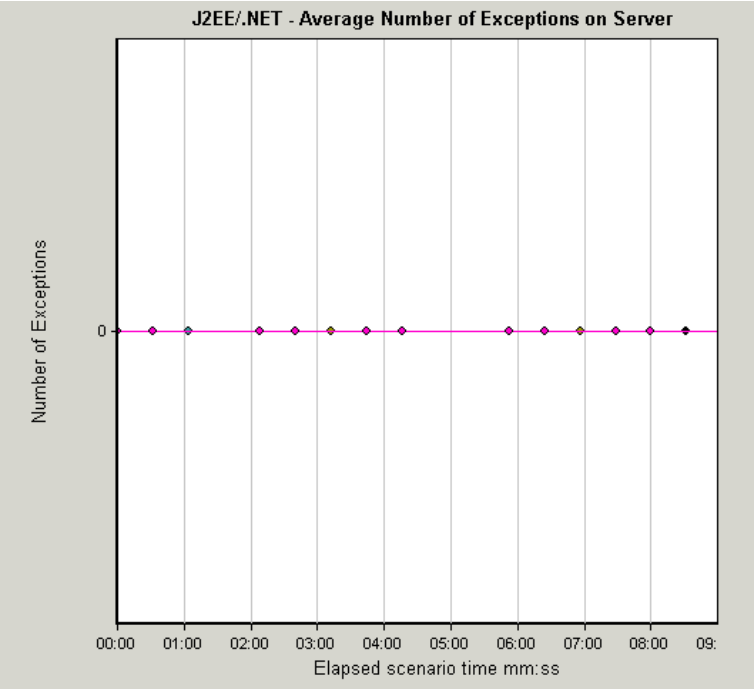


## J2EE/.NET - Average Number of Exceptions on Server Graph

This graph displays the average number of code exceptions per method that were monitored during the selected time range.

| | |
|---|---|
| **X-axis** | Elapsed time of the scenario run. |
| **Y-axis** | Number of events. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

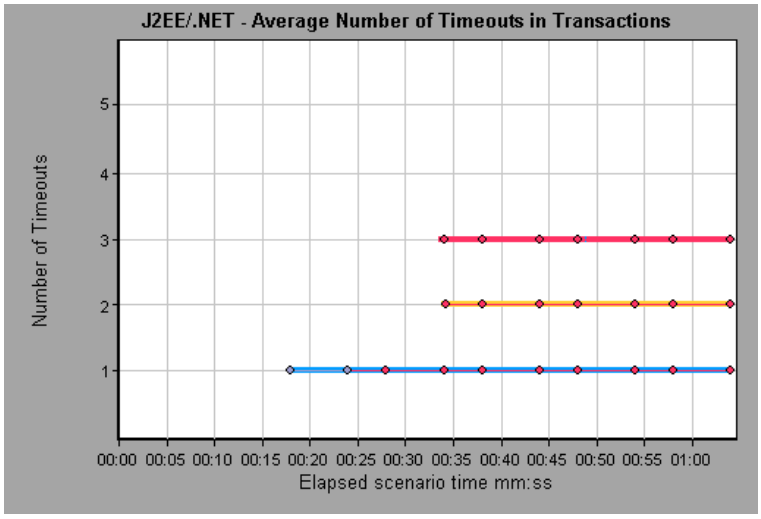



## J2EE/.NET - Average Number of Timeouts in Transactions Graph

This graph displays the average number of timeouts per method, transaction, or request that were monitored during the selected time range.

| | |
|---|---|
| **X-axis** | Elapsed time since the scenario run. |
| **Y-axis** | Represents number of events. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

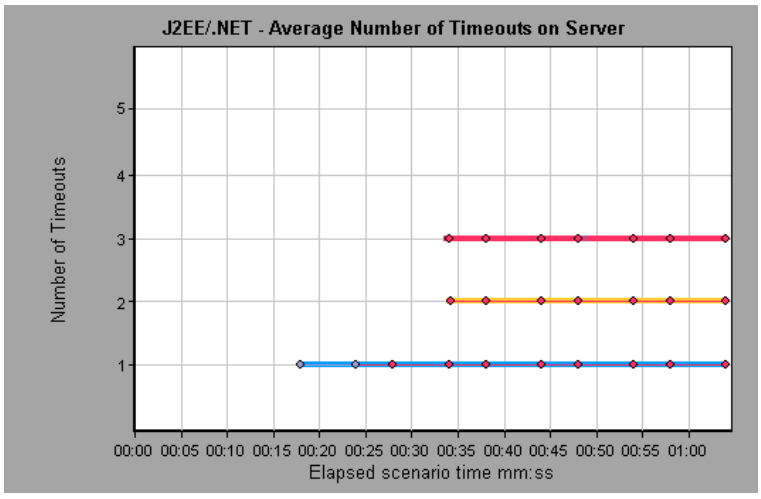



## J2EE/.NET - Average Number of Timeouts on Server Graph

This graph displays the average number of timeouts per method that were monitored during the selected time range.

| | |
|---|---|
| **X-axis** | Elapsed time since the scenario run. |
| **Y-axis** | Number of events. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

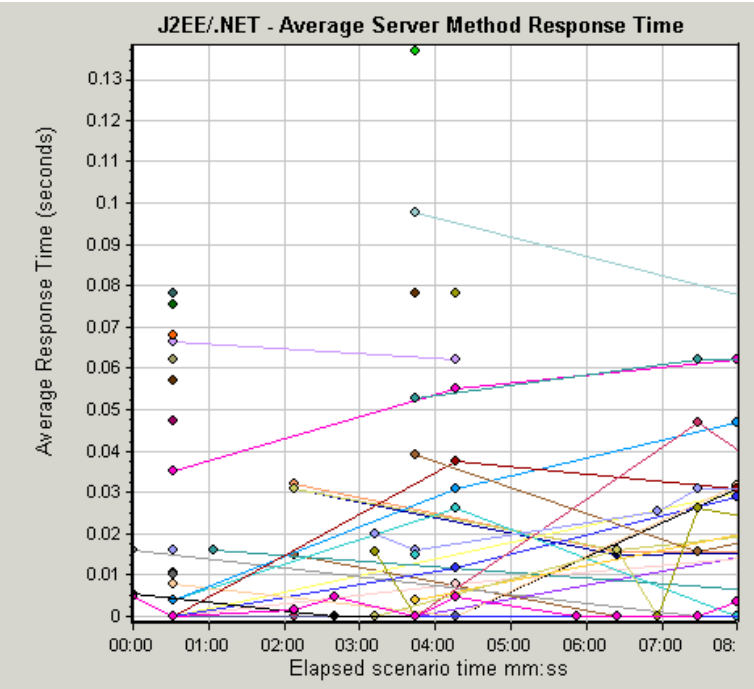


# J2EE/.NET - Average Server Method Response Time Graph

This graph displays the average response time for the server side methods, computed as Total Method Response Time/Number of Method calls.

| | |
|---|---|
| **X-axis** | Elapsed time since the scenario run. |
| **Y-axis** | Average response time (in seconds) per method. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **Note** | The method time does not include calls made from the method to other methods. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**



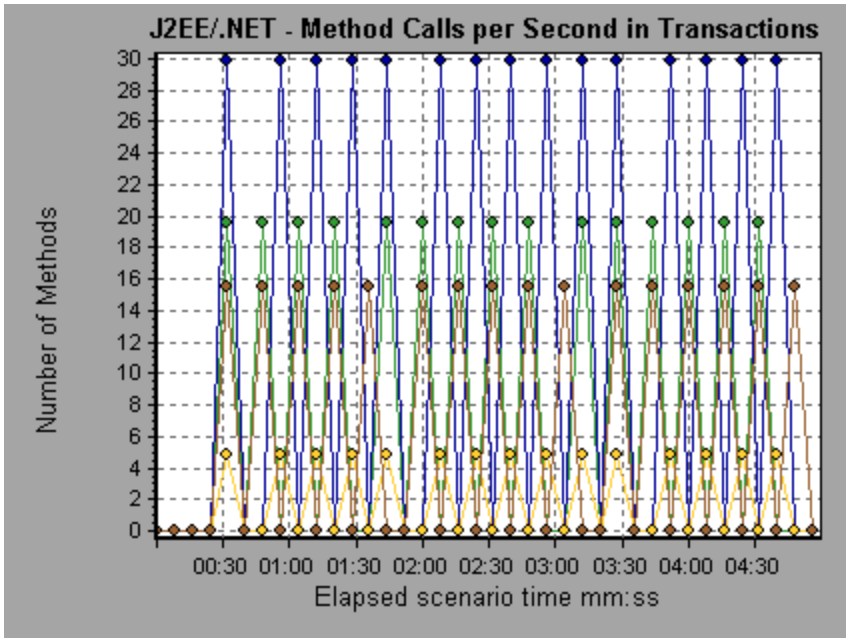

# J2EE/.NET - Method Calls per Second in Transactions Graph

This graph displays the number of completed sampled transactions during each second of a load test scenario run.

The number of transactions included in the sample is determined by the sampling percentage set in the Diagnostics Distribution dialog box in Controller (**Diagnostics > Configuration**).

| | |
|---|---|
| **X-axis** | Elapsed time. |
| **Y-axis** | Represents the number of completed sampled transactions per second. |
| **Breakdown options** | To break the displayed elements down further, see "Using the J2EE and .NET Breakdown Options" on page 1425. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**




## J2EE/.NET - Probes Metrics Graph

This graph displays performance metrics collected by J2EE/.NET Diagnostics probes. Metrics include JVM related data such as Heap usage and Garbage Collection, application server specific metrics, JDBC (Java Database Connectivity) metrics, and more.

| | |
|---|---|
| **X-axis** | Elapsed time since the scenario run. |
| **Y-axis** | Resource usage. The following probe metric data is provided for offline analysis:<br>• **HeapUsed**<br>• **GC Collections/sec**<br>• **GC time Spent in Collections**<br>To include additional Probe metric data in offline Analysis, you use the Diagnostics configuration file, **etc./offline.xml**. For more information, see the *Diagnostics Server Installation and Administration Guide*. |

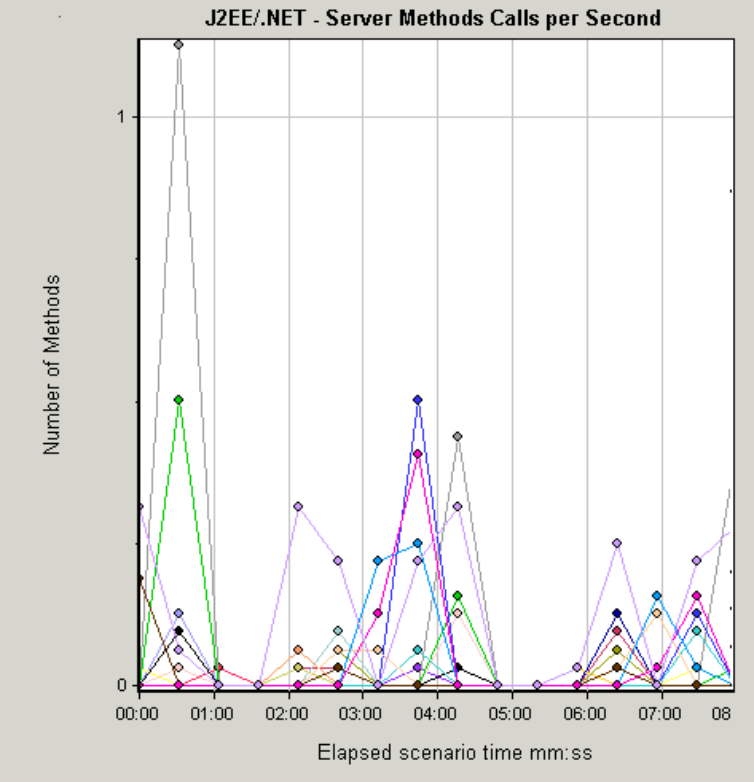
| | |
|---|---|
| **Data Grouping** | By default, the data in the graph is grouped by Category Name (the Diagnostics metric category name) and Probe Name. As a result, the default format for the measurement name is the graph is:<br><Name of metric from Diagnostics (unit of metric)>:<Diagnostics metric category name>:<Probe name><br>If the measurement unit is a count, no unit name is displayed in parentheses. |
| **Important Information** | By default, the following probe metric data is provided for offline analysis: **HeapUsed**, **GC Collections/sec**, and **GC time Spent in Collections**. To include additional Probe metric data in offline Analysis, you use the Diagnostics configuration file, **etc/offline.xml**. For more information, see the *LoadRunner and Performance Center - Diagnostics Integration Guide* .<br>For example, for the following measurement name:<br>• the name of the metric is **GC Time Spent in Collections**.<br>• the value is measured as a percentage.<br>• the metric category name is **GC**.<br>• the Probe name is **MyJBossDev**<br>In addition to the regular Analysis filter criteria, you can also filter and group by the Diagnostics metrics collector name and the host name. |
| **Note** | You need to synchronize the operating system time settings on the Controller machine and the Diagnostics Servers to ensure accurate display of the elapsed scenario time in the Probe Metrics graph. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

## J2EE/.NET - Server Methods Calls per Second Graph

This graph displays the number of completed sampled methods during each second of a load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time of the scenario run. |
| **Y-axis** | Number of completed sampled methods per second. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **Note** | The number of methods included in the sample is determined by the sampling percentage set in the Diagnostics Distribution dialog box in Controller (**Diagnostics > Configuration**). |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

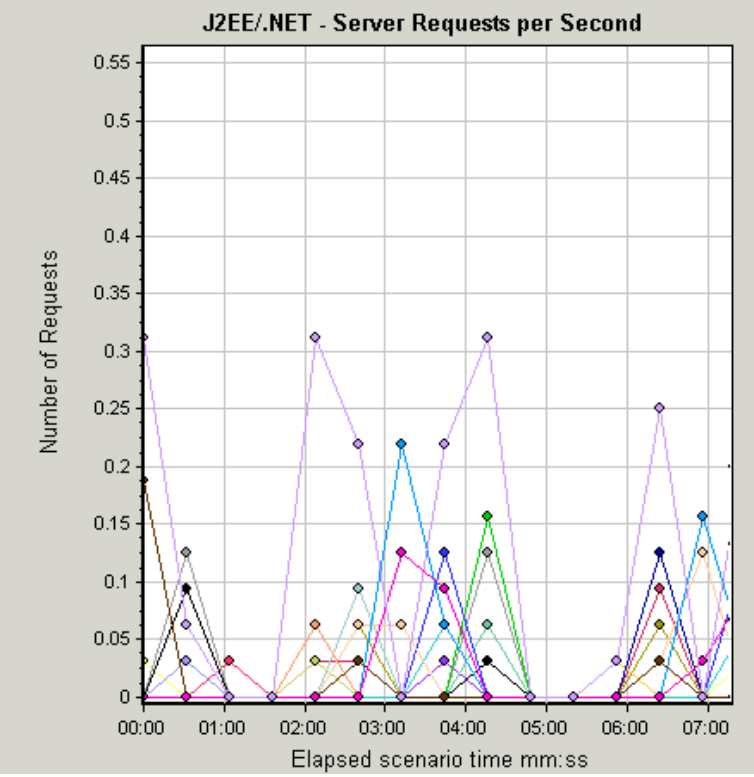



## J2EE/.NET - Server Requests per Second Graph

This graph displays the number of completed sampled requests during each second of a load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time of the scenario run. |
| **Y-axis** | Number of completed sampled requests per second. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **Note** | The number of requests included in the sample is determined by the sampling percentage set in the Diagnostics Distribution dialog box in Controller (**Diagnostics > Configuration**). For more information, see the section on online monitors in the LoadRunner Controller documentation. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

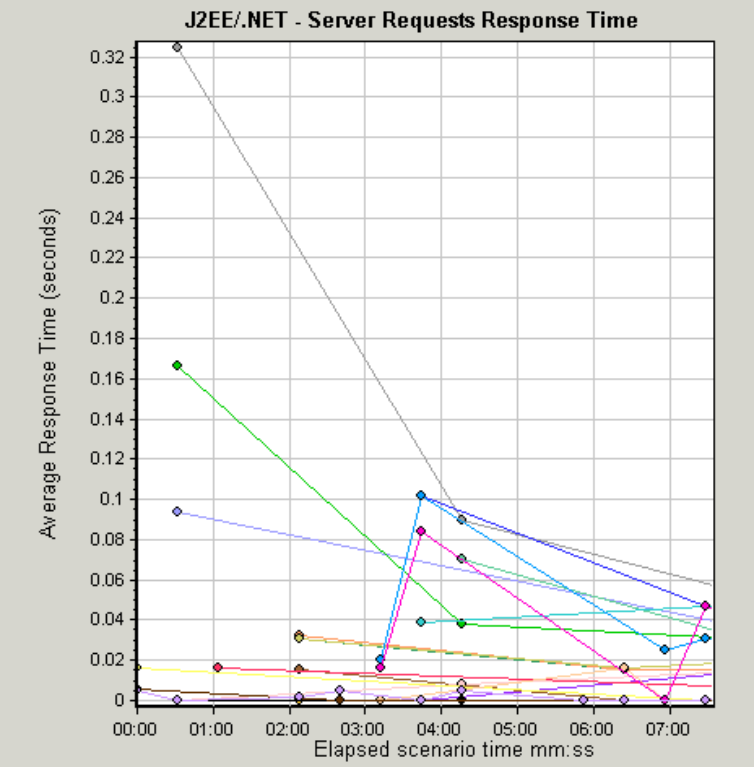



## J2EE/.NET - Server Request Response Time Graph

This graph displays the server response time of requests that include steps that cause activity on the J2EE/.NET backend.

| | |
|---|---|
| **X-axis** | Elapsed time of the scenario time. |
| **Y-axis** | Average time (in seconds) taken to perform each request. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **Note** | The reported times, measured from the point when the request reached the Web server to the point it left the Web server, include only the time that was spent in the J2EE/.NET backend. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**



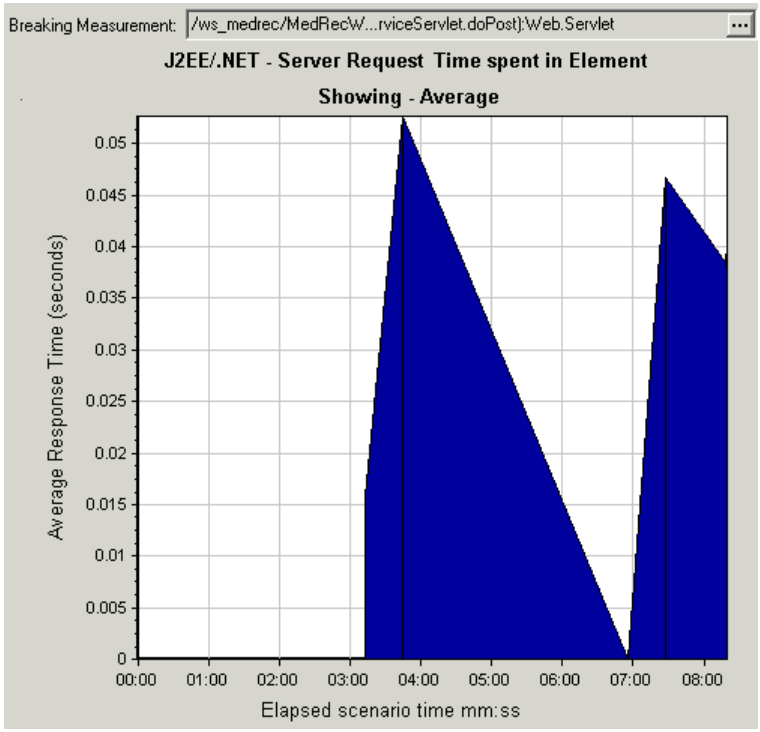

## J2EE/.NET - Server Request Time Spent in Element Graph

This graph displays the server response time for the selected element (layer, class, or method) within each server request.

| | |
|---|---|
| **Purpose** | The time is computed as Total Response Time/Total Number of Server Requests. For example, if a method was executed twice by an instance of server request A and once by another instance of the same server request, and it took three seconds for each execution, the average response time is 9/2, or 4.5 seconds. The server request time does not include the nested calls from within each server request. |
| **X-axis** | Elapsed time of the scenario run. |
| **Y-axis** | Average response time (in seconds) per element within the server request. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |

| | |
|---|---|
| **Filtering properties** | The display of the graph is determined by the Graph Properties selected when the graph is opened, as described:<br>• **None.** Time spent in each server request<br>• **Server request.** Filtered by server request. Grouped by layer.<br>• **Server request and layer.** Filtered by server request and layer. Grouped by class.<br>• **Server request, layer, and class.** Filtered by server request, layer, and class. Grouped by method. |
| **Tips** | To obtain data for this graph, you must first install J2EE/.NET Diagnostics. Before you can view Diagnostics for J2EE & .NET data in a particular load test scenario, you need to configure the Diagnostics parameters for that scenario. For more information, see the section on online monitors in the LoadRunner Controller documentation. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**



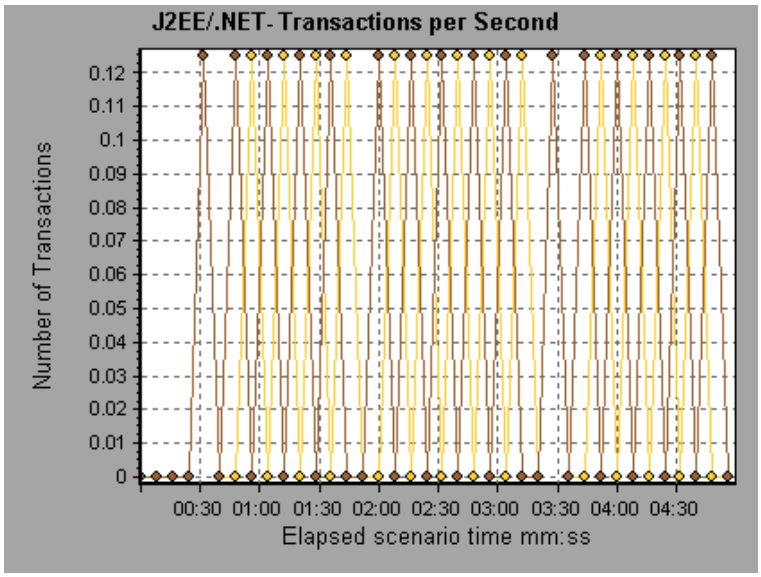

## J2EE/.NET - Transactions per Second Graph

This graph displays the number of completed sampled transactions during each second of a load test scenario run.

The number of transactions included in the sample is determined by the sampling percentage set in the Diagnostics Distribution dialog box in Controller (**Diagnostics > Configuration**). For more information, see the section on online monitors in the LoadRunner Controller documentation.

| | |
|---|---|
| **X-axis** | Elapsed time. |
| **Y-axis** | Number of completed sampled transactions per second |
| **Breakdown options** | To break the displayed elements down further, see "Using the J2EE and .NET Breakdown Options" on page 1425. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**

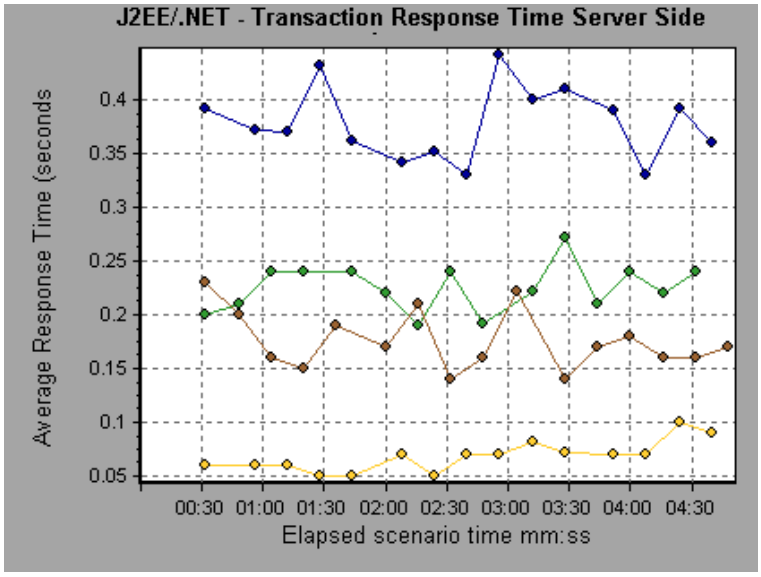



## J2EE/.NET - Transaction Response Time Server Side Graph

This graph displays the transaction server response time of transactions that include steps that cause activity on the J2EE/.NET backend. The reported times, measured from the point when the transaction reached the Web server to the point it left the Web server, include only the time that was spent in the J2EE/.NET backend.

| | |
|---|---|
| **X-axis** | Elapsed time. |
| **Y-axis** | Average response time (in seconds) of each transaction. |
| **Breakdown options** | "Using the J2EE and .NET Breakdown Options" on page 1425 |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417 |

**Example**



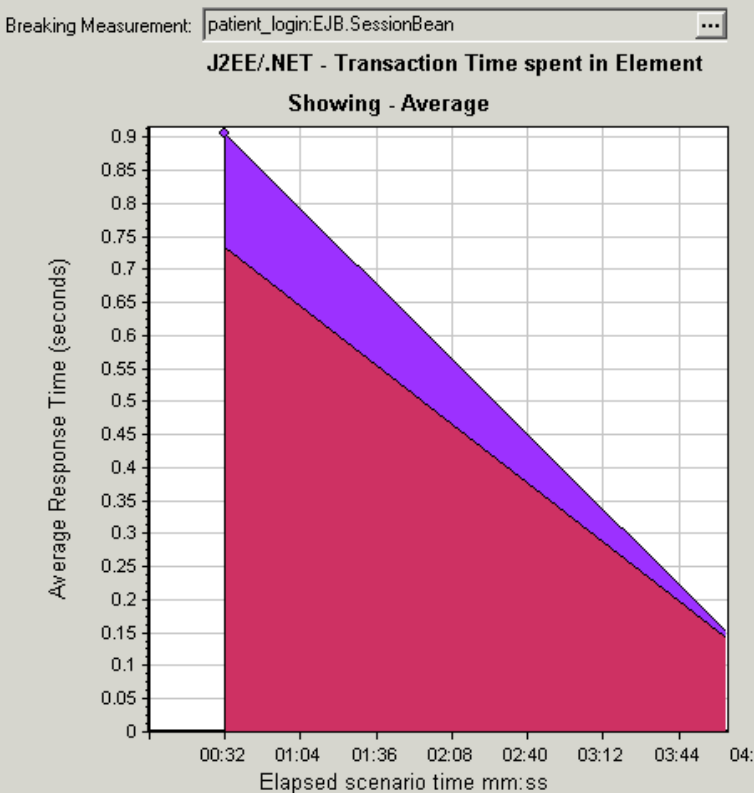

# J2EE/.NET - Transaction Time Spent in Element Graph

This graph displays the server response time for the selected element (layer, class, or method) within each transaction.

## Graph UI Elements

| | |
|---|---|
| **X-axis** | Elapsed time. |
| **Y-axis** | Average response time (in seconds) per element within the transaction. |
| **Breakdown options** | The display of graph data is determined by the graph properties selected when the graph was opened, as described in the following table: For information on filtering on graph data, see "Filtering Graph Data Overview" on page 1342.<br>You can break down the displayed elements. For more information, see "Using the J2EE and .NET Breakdown Options" on page 1425. |
| **Tips** | To obtain data for this graph, you must enable the J2EE & .NET Diagnostics module (from Controller) before running the load test scenario. |
| **Note** | The time is computed as Total Response Time/Total Number of Transactions. For example, if a method was executed twice by an instance of transaction A and once by another instance of the same transaction, and it took three seconds for each execution, the average response time is 9/2, or 4.5 seconds. The transaction time does not include the nested calls from within each transaction. |
| **See also** | "J2EE and .NET Diagnostics Graphs Overview" on page 1417<br>"Filtering and Sorting Graph Data" on page 1342 |

**Example**




## Graph data display filters

| If you filter by these properties... | The graph data is displayed like this |
|---|---|
| None | Time spent in each transaction. |
| Transaction | Filtered by transaction. Grouped by layer. |
| Transaction and layer | Filtered by transaction and layer. Grouped by class. |
| Transaction, layer, and class | Filtered by transaction, layer, and class. Grouped by method. |

# JMeter Analysis Graphs

The **JMeter** graphs provide you with statistics for the JMeter tests during the scenario run.

**Note:** There is a difference in the default data granularity displayed in the JMeter graphs provided with the standard JMeter plugin, and the Analysis JMeter graphs, causing the results to look different. The granularity of the Analysis JMeter graphs is set as 1 data point every 5 seconds, as opposed to 1 data point every 1 second in the plugin graphs. You can change the

granularity, as described in "Set up JMeter Analysis graphs" below.

## Set up JMeter Analysis graphs

You must manually add the JMeter graphs to your Analysis session.

1. With the Analysis session open, click the **Add New Graph** button.
2. In the Open a New Graph dialog box, find the **JMeter Graphs** section and select the graph you want to add. Click **Open Graph**.

   **Note:** If JMeter graphs do not appear in the list, check the troubleshooting in "JMeter Best Practices and Troubleshooting" on page 948.
3. The graph is displayed and added to the session **Graphs** list.
4. To change the granularity, right-click on the graph and select **Set Granularity**. Change to the granularity you require.

## Available JMeter graphs

| Monitor | Description |
|---|---|
| **JMeter Hits** | Shows the number of hits (y-axis) per second (x-axis) generated by the test plan.<br>It does not include child samples from transactions and embedded resources. |
| **JMeter Throughput** | Shows the amount of throughput (y-axis) during each second of the test run (x-axis). Throughput is measured in bytes and represents the amount of bytes sent and received by JMeter during the load test. |

# Middleware Performance Graphs

A primary factor in a transaction's response time is the middleware performance usage. LoadRunner's Middleware Performance monitors provide you with information about the middleware performance usage of the IBM WebSphere MQ servers during a load test scenario execution. To obtain performance data, you need to activate the online monitor for the server and specify which resources you want to measure before executing the scenario.

For more information, see the section on online monitors in the LoadRunner Controller documentation.

# IBM WebSphere MQ Counters

## Queue Performance Counters

| Measurement | Description |
|---|---|
| **Event - Queue Depth High (events per second)** | An event triggered when the queue depth reaches the configured maximum depth. |
| **Event - Queue Depth Low (events per second)** | An event triggered when the queue depth reaches the configured minimum depth. |
| **Event - Queue Full (events per second)** | An event triggered when an attempt is made to put a message on a queue that is full. |
| **Event - Queue Service Interval High (events per second)** | An event triggered when no messages are put to or retrieved from a queue within the timeout threshold. |
| **Event - Queue Service Interval OK (events per second)** | An event triggered when a message has been put to or retrieved from a queue within the timeout threshold. |
| **Status - Current Depth** | The current count of messages on a local queue. This measurement applies only to local queues of the monitored queue manager. |
| **Status - Open Input Count** | The current count of open input handles. Input handles are opened so that an application may "put" messages to a queue. |
| **Status - Open Output Count** | The current count of open output handles. Output handles are opened so that an application may "get" messages from a queue. |

## Channel Performance Counters

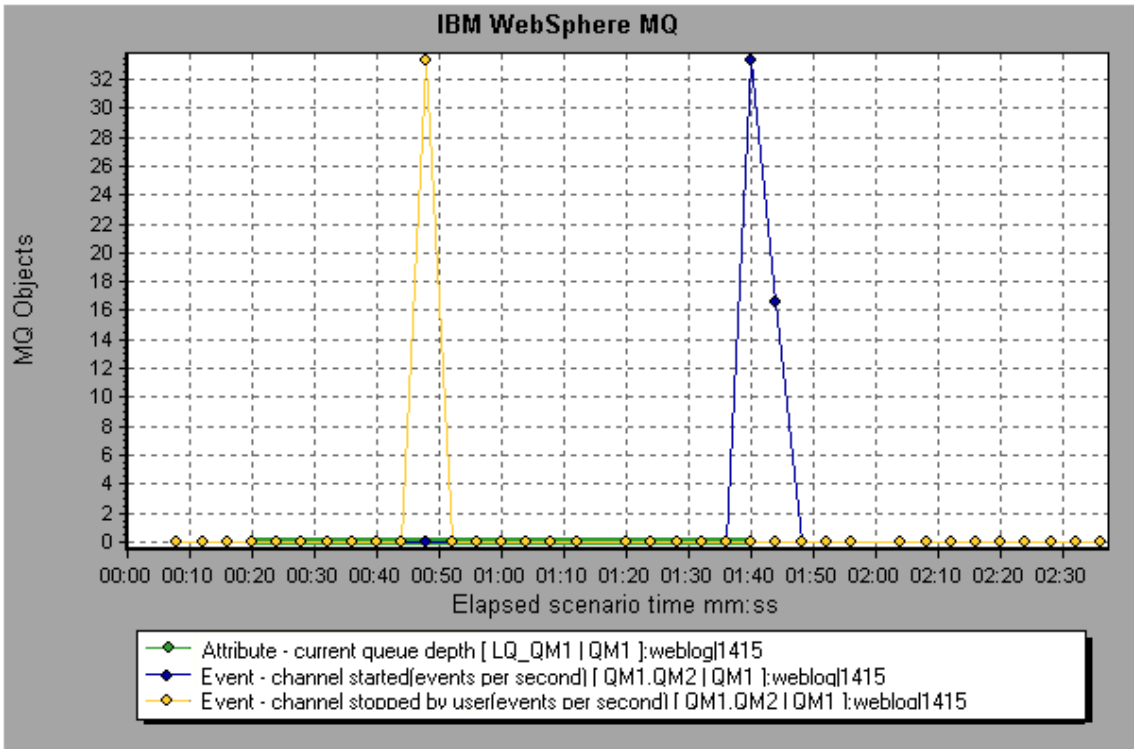
| Measurement | Description |
|---|---|
| **Event - Channel Activated (events per second)** | An event generated when a channel, waiting to become active but inhibited from doing so due to a shortage of queue manager channel slots, becomes active due to the sudden availability of a channel slot. |
| **Event - Channel Not Activated (events per second)** | An event generated when a channel attempts to become active but is inhibited from doing so due to a shortage of queue manager channel slots. |

| Measurement | Description |
|---|---|
| **Event - Channel Started (events per second)** | An event generated when a channel is started. |
| **Event - Channel Stopped (events per second)** | An event generated when a channel is stopped, regardless of source of stoppage. |
| **Event - Channel Stopped by User (events per second)** | An event generated when a channel is stopped by a user. |
| **Status - Channel State** | The current state of a channel. Channels pass through several states from stopped (inactive state) to running (fully active state). Channel states range from 0 (stopped) to 6 (running). |
| **Status - Messages Transferred** | The count of messages that have been sent over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available. |
| **Status - Buffer Received** | The count of buffers that have been received over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available. |
| **Status - Buffer Sent** | The count of buffers that have been sent over the channel. If no traffic is occurring over the channel, this measurement will be zero. If the channel has not been started since the queue manager was started, no measurement will be available. |
| **Status - Bytes Received** | The count of bytes that have been received over the channel. If no traffic is occurring over the channel, this measurement will appear as zero. If the channel has not been started since the queue manager was started, no measurement will be available. |
| **Status - Bytes Sent** | The count of bytes that have been sent over the channel. If no traffic is occurring over the channel, this measurement will appear as zero. If the channel has not been started since the queue manager was started, no measurement will be available. |

## IBM WebSphere MQ Graph

This graph shows the resource usage of IBM WebSphere MQ Server channel and queue performance counters as a function of the elapsed load test scenario time.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource usage of the IBM WebSphere MQ Server channel and queue performance counters. |
| **Note** | To obtain data for this graph, you need to enable the IBM WebSphere MQ monitor (from Controller) and select the default measurements you want to display, before running the scenario. |
| **See also** | "Middleware Performance Graphs" on page 1447<br>"IBM WebSphere MQ Counters" on page 1448 |




# MQTT Statistics Graphs

The MQTT Statistics graphs provide you with statistics about messages published from and received by the MQTT broker during a scenario run.

## Publish/Received Message Rate graphs

Display the number of messages published or received at any given time during the load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run |
| **Y-axis** | Average number of messages per second published/received |

## Publish/Received Throughput graphs

Display the amount of data (in bytes) that Vusers received from or published to the MQTT broker at any given second of the load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run |
| **Y-axis** | Throughput to or from the MQTT broker, in bytes |

## In-flight Messages graph

Displays the number of in-flight message on the MQTT broker at any given second of the load test scenario run. In-flight messages are QoS 1 or 2 messages that can be in the process of being transmitted, including messages currently going through handshakes and messages that are being retried.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run |
| **Y-axis** | Number of inflight message received |

## Average Publish Duration graph

Displays the average amount of time (in seconds) that it took to publish a message on the MQTT broker at any given second of the load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run |
| **Y-axis** | Average number of seconds it took to publish a message on the MQTT broker |

## Active Connection graph

Displays the number of active connections to the MQTT broker at any given moment of the load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run |

| | |
|---|---|
| **Y-axis** | Number of active connections |

## Average Connect Duration graph

Displays the average amount of time (in seconds) that there was an active connection to the MQTT broker at any given second of the load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run |
| **Y-axis** | The average number of seconds that there was an active connection to the MQTT broker |

## Connect Statistics graph

A pie graph that displays the number of successful versus failed connections to the MQTT broker during the load test scenario run.

### See also:

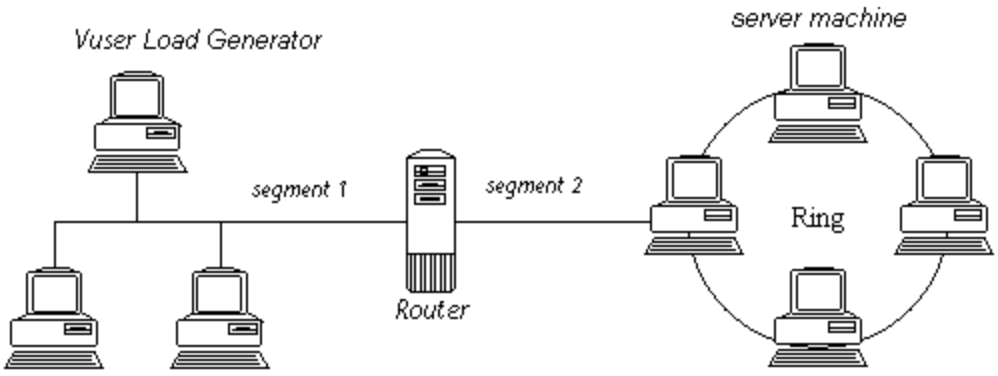
- "MQTT Statistics Monitoring" on page 1198

# Network Monitor Graphs

## Network Monitor Graphs Overview

Network configuration is a primary factor in the performance of applications and Web systems. A poorly designed network can slow client activity to unacceptable levels. In an application, there are many network segments. A single network segment with poor performance can affect the entire application.

The following diagram shows a typical network. To go from the server machine to the Vuser machine, data must travel over several segments.



To measure network performance, the Network monitor sends packets of data across the network. When a packet returns, the monitor calculates the time it takes for the packet to go to the requested node and return.

The Network Sub-Path Time graph displays the delay from the source machine to each node along the path. The Network Segment Delay graph displays the delay for each segment of the path. The Network Delay Time graph displays the delay for the complete path between the source and destination machines.

Using the Network Monitor graphs, you can determine whether the network is causing a bottleneck. If the problem is network-related, you can locate the problematic segment so that it can be fixed.

In order for Analysis to generate Network monitor graphs, you must activate the Network monitor before executing the load test scenario. In the Network monitor settings, you specify the path you want to monitor. For information about setting up the Network monitor, see "Network Delay Monitoring" on page 1201.

## Network Delay Time Graph

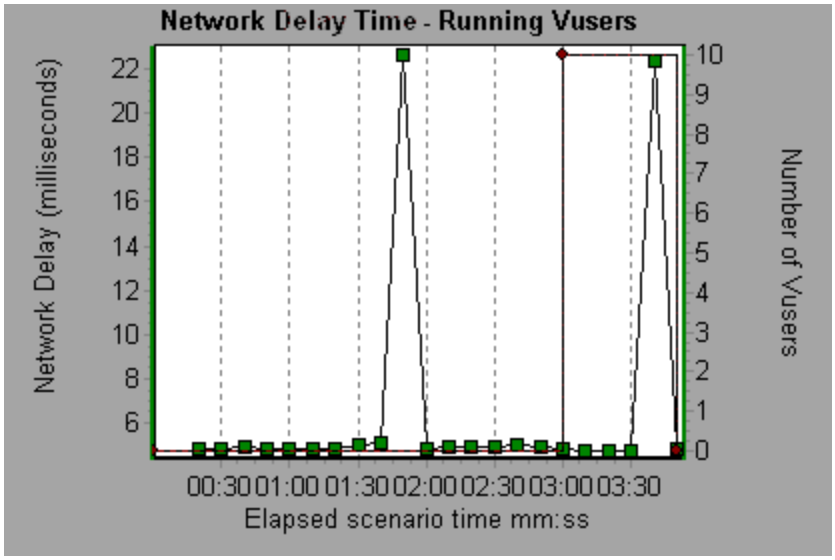
This graph shows the delays for the complete path between the source and destination machines (for example, the database server and Vuser load generator). The graph maps the delay as a function of the elapsed load test scenario time.

Each path defined in Controller is represented by a separate line with a different color in the graph.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Network delay time. |
| **Tips** | You can merge various graphs to determine if the network is a bottleneck. For example, using the Network Delay Time and Running Vusers graphs, you can determine how the number of Vusers affects the network delay. |
| **See also** | "Network Monitor Graphs Overview" on the previous page |

**Example**

In the following example of a merged graph, the network delays are compared to the running Vusers. The graph shows that when all 10 Vusers were running, a network delay of 22 milliseconds occurred, implying that the network may be overloaded.



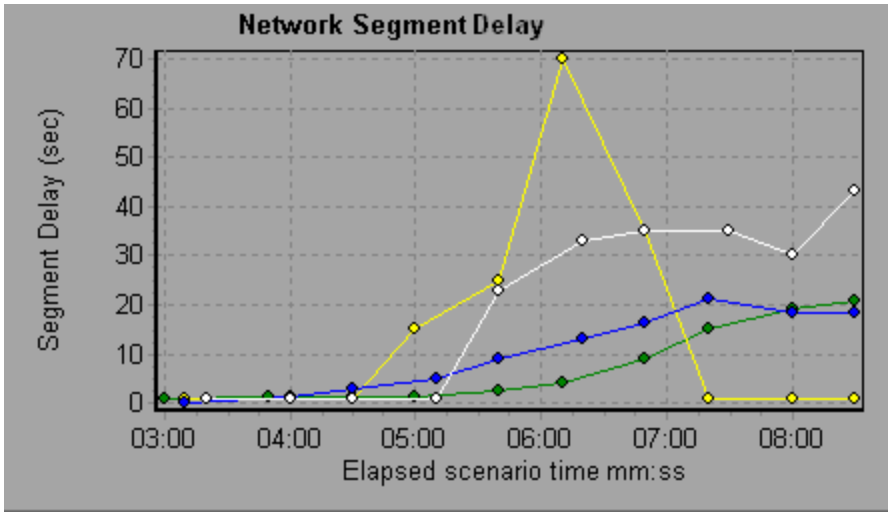

## Network Segment Delay Graph

This graph shows the delay for each segment of the path according to the elapsed load test scenario time. Each segment is displayed as a separate line with a different color.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Network delay time. |
| **Note** | The segment delays are measured approximately, and do not add up to the network path delay which is measured exactly. The delay for each segment of the path is estimated by calculating the delay from the source machine to one node and subtracting the delay from the source machine to another node. For example, the delay for segment B to C is calculated by measuring the delay from the source machine to point C, and subtracting the delay from the source machine to point B. |
| **See also** | "Network Monitor Graphs Overview" on page 1452 |

Example

In the following example, four segments are shown. The graph indicates that one segment caused a delay of 70 seconds in the sixth minute.



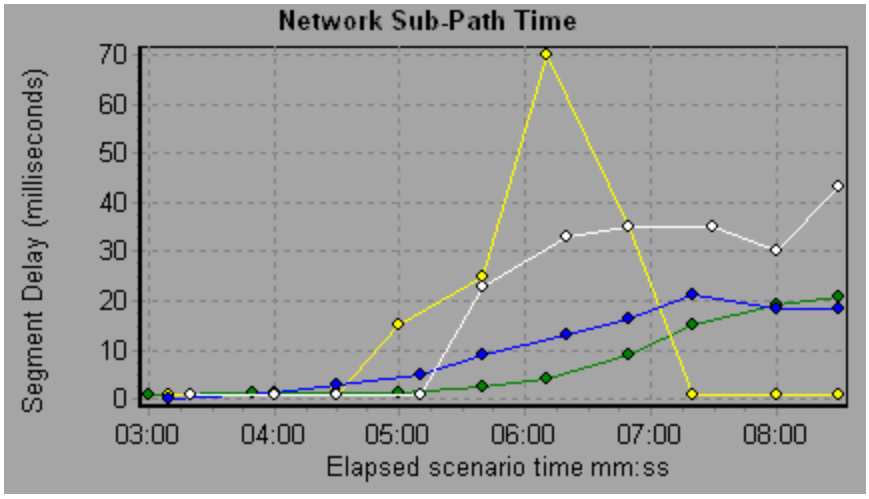

## Network Sub-Path Time Graph

This graph displays the delay from the source machine to each node along the path according to the elapsed load test scenario time. Each segment is displayed as a separate line with a different color.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Network delay time. |
| **Note** | The delays from the source machine to each of the nodes are measured concurrently, yet independently. It is therefore possible that the delay from the source machine to one of the nodes could be greater than the delay for the complete path between the source and destination machines. |
| **See also** | "Network Monitor Graphs Overview" on page 1452 |

**Example**

In the following example, four segments are shown. The graph indicates that one segment caused a delay of 70 milliseconds in the sixth minute.




# Network Virtualization Graphs

LoadRunner integrates with Network Virtualization. This enables you to test point-to-point performance of WAN or other network deployed products under real-world network conditions. By installing software on your load generators, you introduce highly probable effects such as latency, packet loss, and link faults over your network. As a result of this, your scenario performs the test in an environment that better represents the actual deployment of your application.

You can create more meaningful results by configuring multiple load generator machines or groups on a single load generator with the same unique set of network effects, and by giving each set a unique location name, such as NY- London. When viewing scenario results in Analysis, you can group the metrics according to their location names.

For details about the integration, see "Network Virtualization Integration" on page 1237.

## Packet Loss Graph

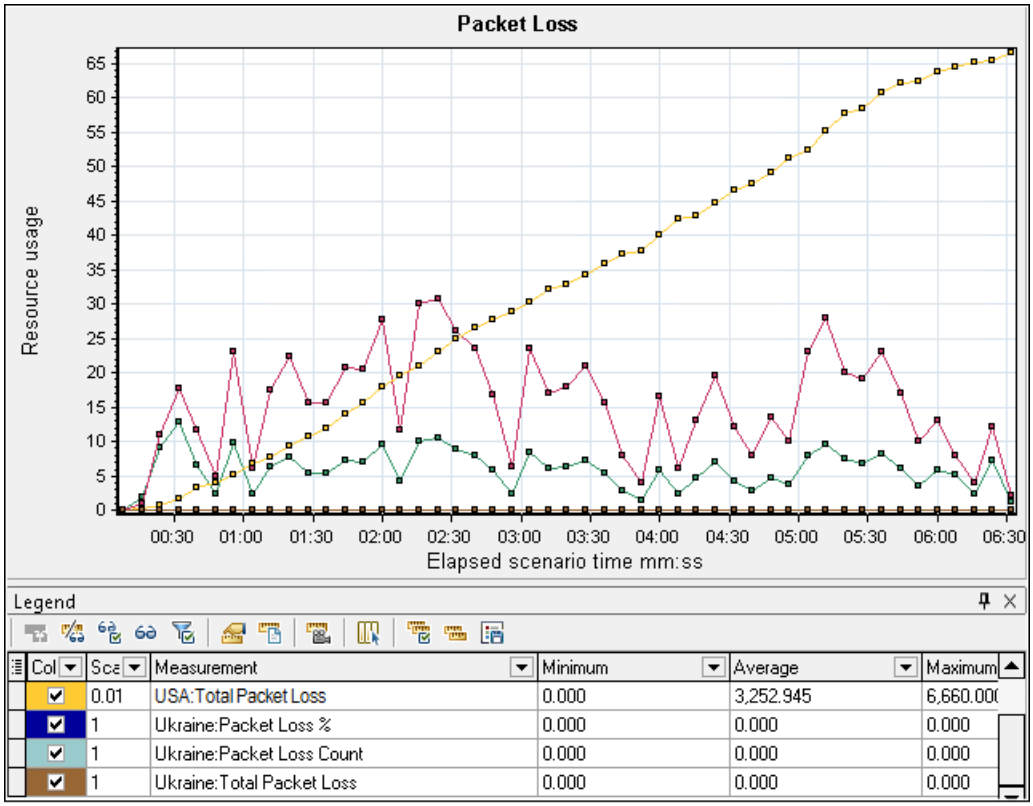
This graph shows packets lost during the last second of the scenario run. Packet loss occurs when data packets fail to reach their destination. It can result from gateway overload, signal degradation, channel congestion, or faulty hardware.

| | |
|---|---|
| **Purpose** | Helps you understand how many data packets were lost over a specific time interval. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The following measurements:<br>• The percentage of lost packets from all packets that were sent.<br>• The number of data packets that were lost over 60 seconds.<br>• The total number of packets that were lost. |

| | |
|---|---|
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **Tip** | For LoadRunner Analysis (not applicable to monitoring graphs):<br>To view information for a specific location:<br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Network Virtualization Graphs" on the previous page |

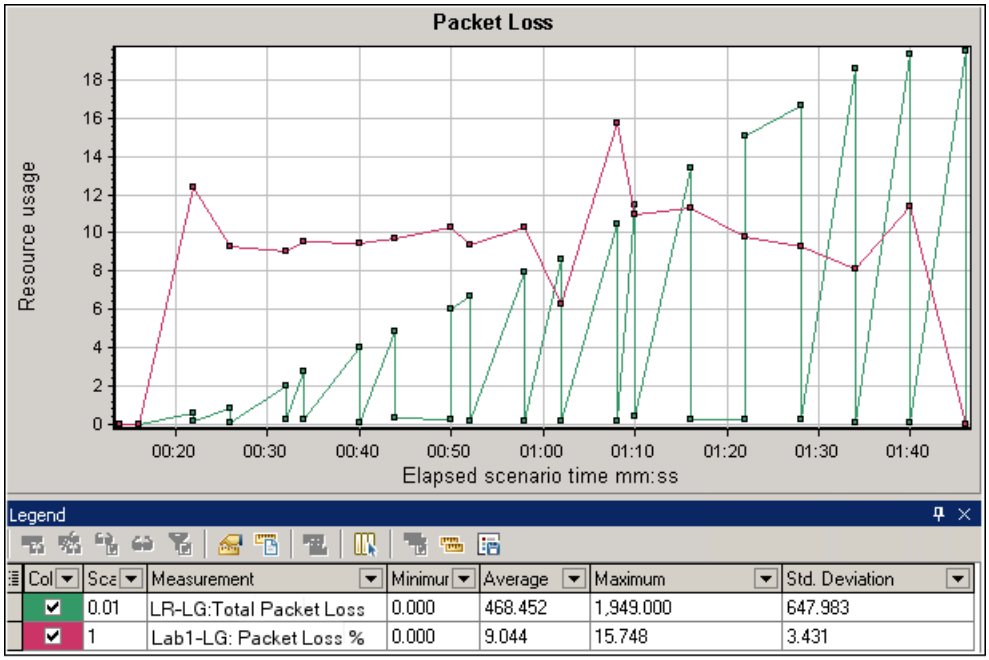
**Example - Network Virtualization Per Group**

The following example shows how the total of packet loss for the **USA** group increased as the scenario progressed.



**Example - Network Virtualization Per Load Generator**

In the following example, you can see that the packet loss is grouped by load generator. This was the mode selected when you enabled Network Virtualization for the scenario.




## Average Latency Graph

This graph shows the average recorded time required for a packet of data to travel from the indicated source point to the required destination, measured in milliseconds in the last 60 seconds.

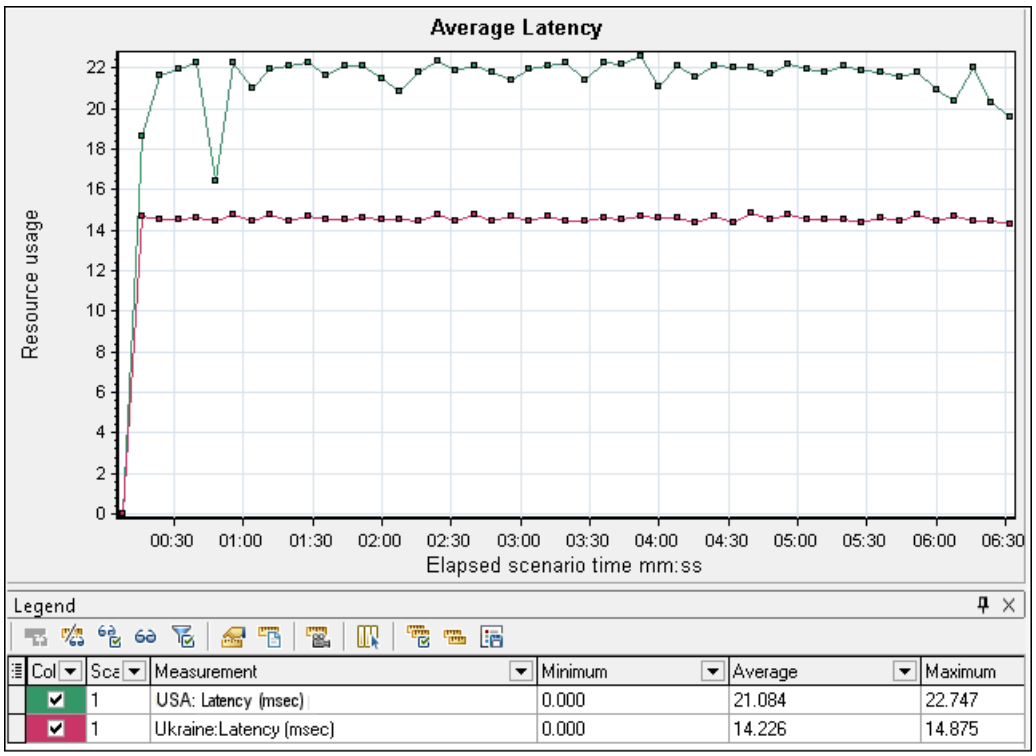
| | |
|---|---|
| **Purpose** | Helps you evaluate the time required for a packet of data to travel over the network. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The average latency—the time in milliseconds required for a packet of data to reach its destination, per 60 second intervals. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |

| | |
|---|---|
| **Tips** | For LoadRunner Analysis (not applicable to monitoring graphs):<br>To view information for a specific location:<br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | • "Network Virtualization Graphs" on page 1456<br>• "Custom Filter Dialog Box" on page 1350 |

If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Graph" on page 1456.

**Example - Network Virtualization Per Group**

In the following example, you can see that the latency for the **USA** group reached its peak at nearly 4 minutes into the scenario run, while the **Ukraine** group remained fairly constant at approximately 14 msec.



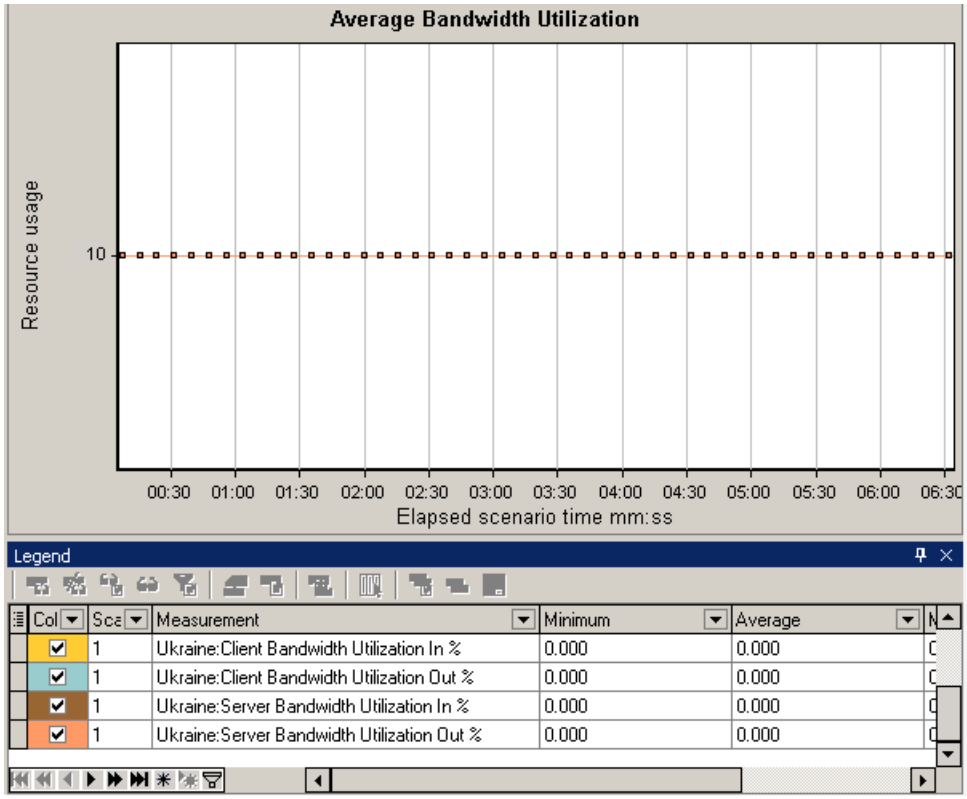
# Average Bandwidth Utilization Graph

This graph shows the average bandwidth utilized by a virtual user or a virtualized location from the maximal available bandwidth allocated for it during the last second, measured in percentages.

| | |
|---|---|
| **Purpose** | Helps you evaluate the bandwidth used over your network. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The percentage of bandwidth utilization. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **Tips** | For LoadRunner Analysis (not applicable to monitoring graphs):<br>To view information for a specific location:<br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Network Virtualization Graphs" on page 1456 |

**Example**

In the following example, you can see that the bandwidth utilization for all locations and measurements, was constant at 10%.




If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Graph" on page 1456.

## Average Throughput Graph

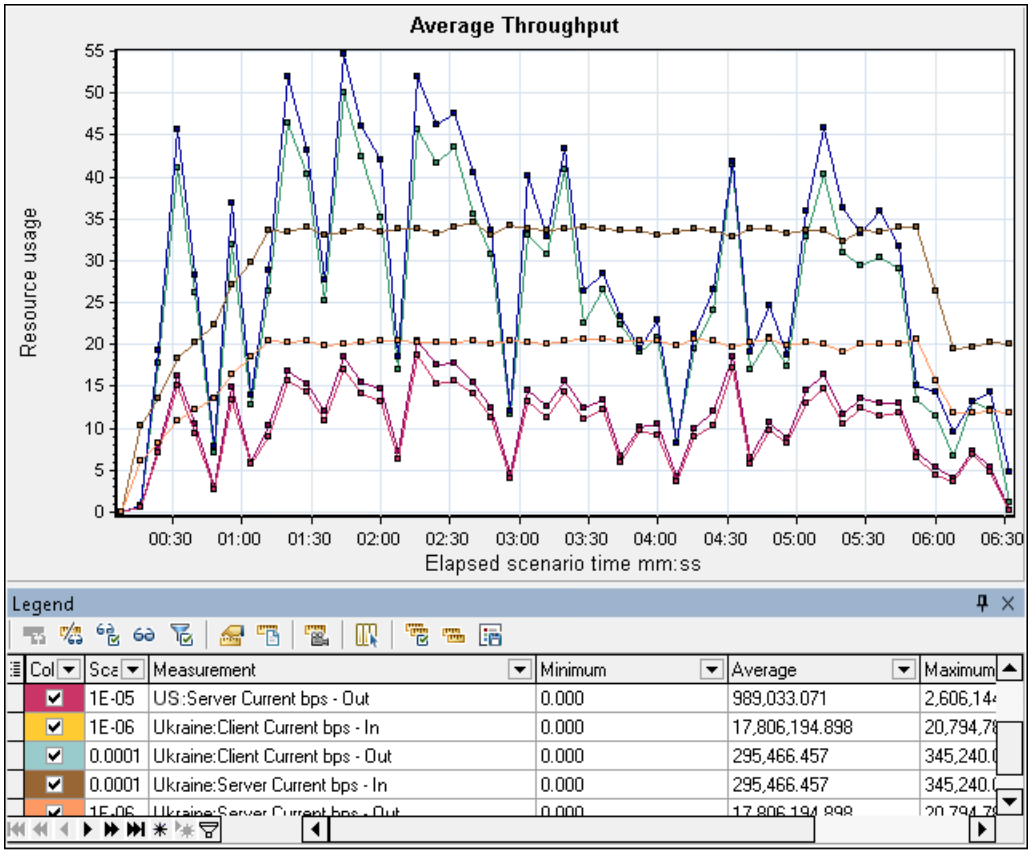
This graph shows the average data traffic passing to or from the virtualized location, measured in kilobytes per second (bps).

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of load Vusers generate, in terms of the number of server and client throughput. The graph shows metrics for input and output traffic for both the server and client machines. Use the legend below the graph to determine the line color for each metric. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The rate of data passing to and from the virtual location, in bps for the following metrics per group or load generator:<br>• Input to the client machine<br>• Output from the client machine<br>• Input to the server machine<br>• Output from the server machine |

| | |
|---|---|
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **Tips** | For LoadRunner Analysis (not applicable to monitoring graphs):<br>To view information for a specific location:<br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Total Throughput Graph" below |

**Example**

In the following example, the average server input throughput was the lowest for the **Ukraine** group.




If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Graph" on page 1456.

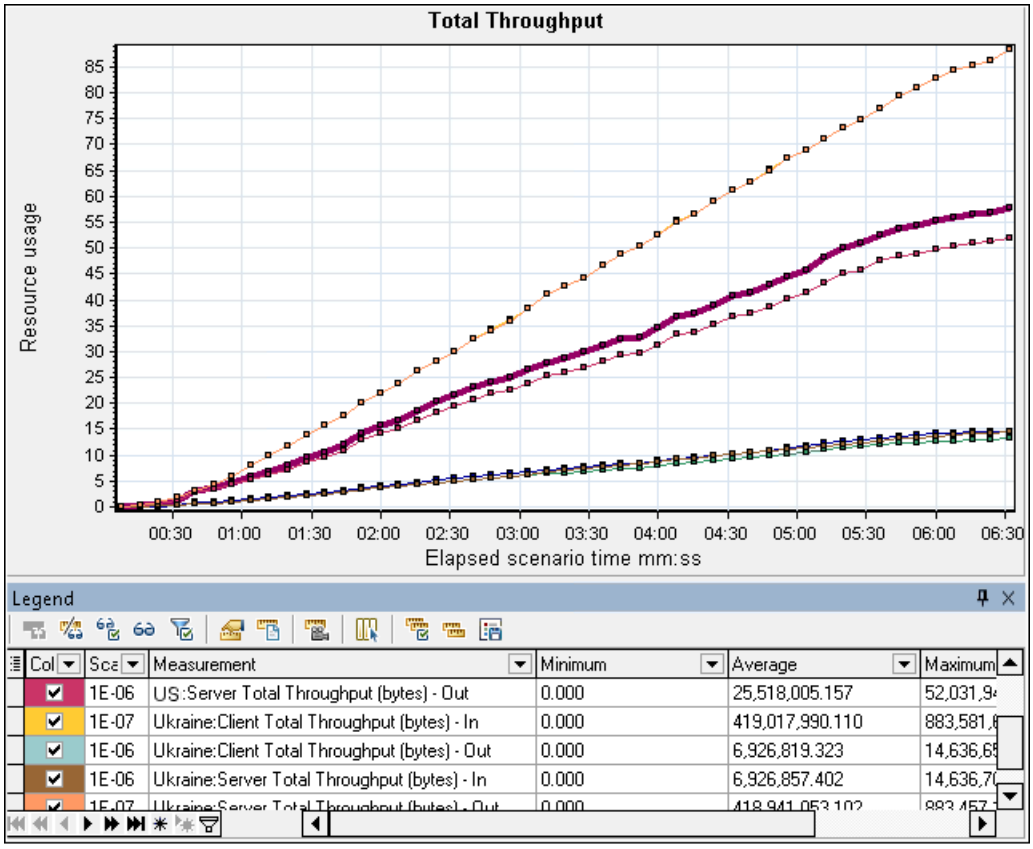
## Total Throughput Graph

Displays the total data traffic passing to or from the virtualized location, measured in bytes.

| | |
|---|---|
| **Purpose** | Helps you evaluate the total amount of load that Vusers generate while running a scenario with network virtualization.<br><br>The graph shows metrics for input and output traffic for both the server and client machines. The legend below the graph indicates the line color for each of these metrics. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Throughput of the server, in bytes per second. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the **General** tab of the Options dialog box. |
| **Tips** | For LoadRunner Analysis (not applicable to monitoring graphs):<br><br>To view information for a specific location:<br><br>1. Click within the graph.<br>2. Select **Set Filter/ Sort By** from the right-click menu to open the Graph Settings dialog box.<br>3. In the **Filter condition** section, select the *Location Name* row, and select the desired location from the drop-down list. |
| **See also** | "Average Throughput Graph" on page 1461 |

**Example**

In the following example, the highest throughput level was for the input data to the client, for the **Ukraine** group.

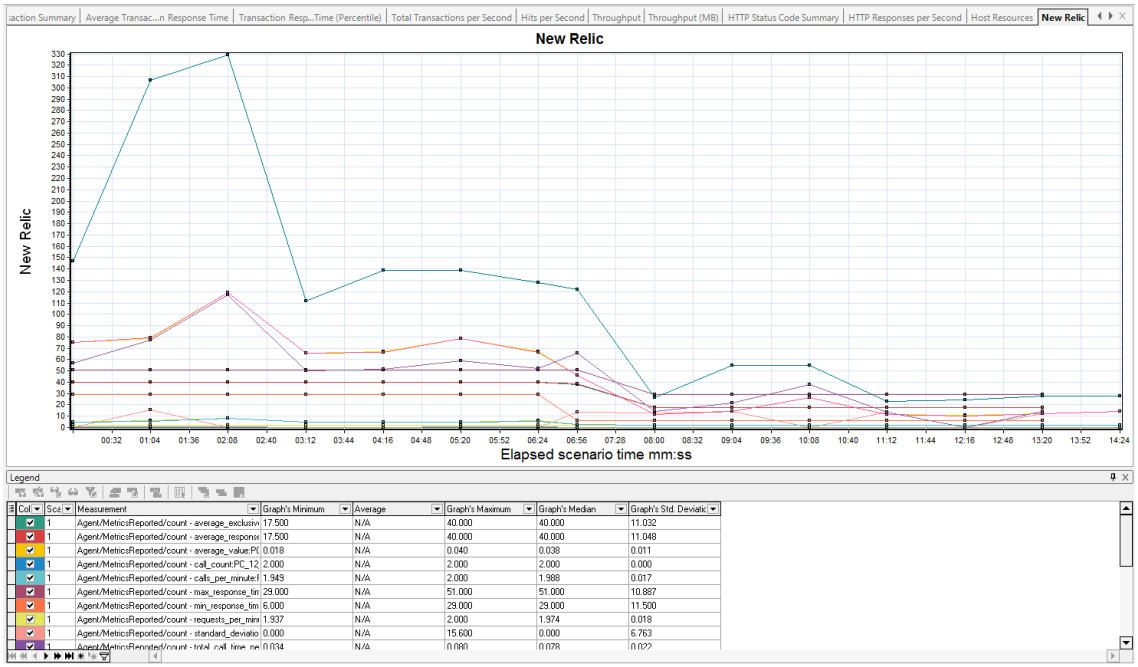



If you enabled Network Virtualization per load generator (and not per group), the graph shows the measurements per load generator, as shown in the "Packet Loss Graph" on page 1456.

# New Relic Graph

> **Note:** Available for Performance Center only.

The New Relic graph displays metrics from the New Relic server about monitored applications during a scenario run.




For details about the New Relic monitor, see the Performance Center help.

For more information, see the New Relic documentation.

# Service Virtualization Graphs

The Service Virtualization graphs are similar to the corresponding monitors used by the LoadRunner Controller. For details, see "Service Virtualization Monitors" on page 1255.

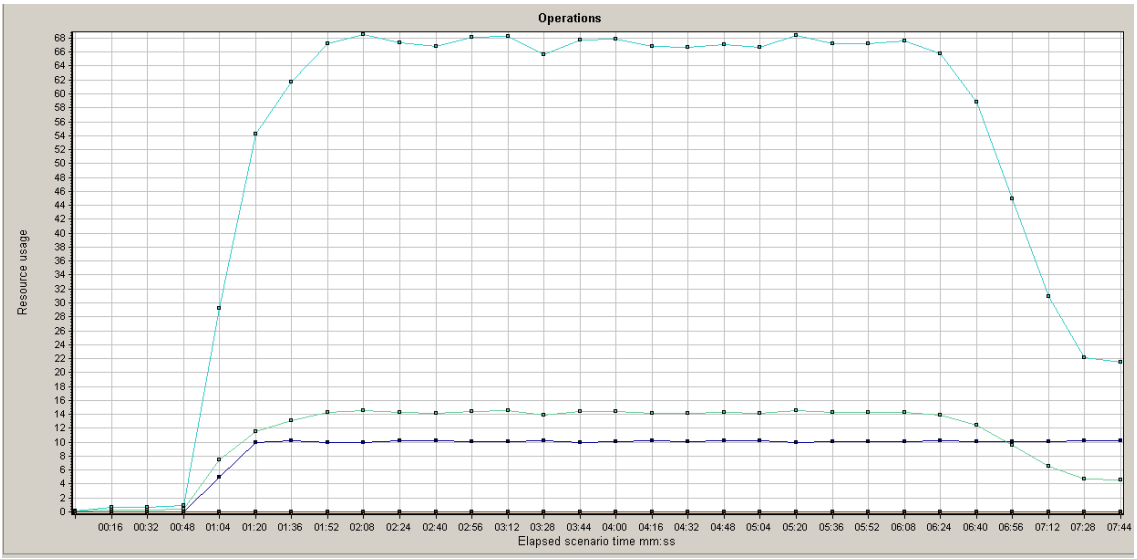
## Service Virtualization Operations Graph

This graph displays a summary for Service Virtualization - Operations.

| | |
|---|---|
| **X-axis** | The elapsed time from the beginning of the scenario run. |
| **Y-axis** | The number of resources used. |
| **Tips** | • To isolate the measurement with the most problems, it may be helpful to sort the legend window according to the average number of resources used. To sort the legend by average, double-click the **Average** column heading.<br>• To identify a measurement in the graph, you can select it. The corresponding line in the legend window is selected. |
| **Note** | To use this graph, you must first open a Service Virtualization project in Controller. |
| **See also** | Web Page Diagnostics Graph |

**Example**

Using the graph, you can track which resources were most problematic, and at which point(s) during the scenario the problem(s) occurred.



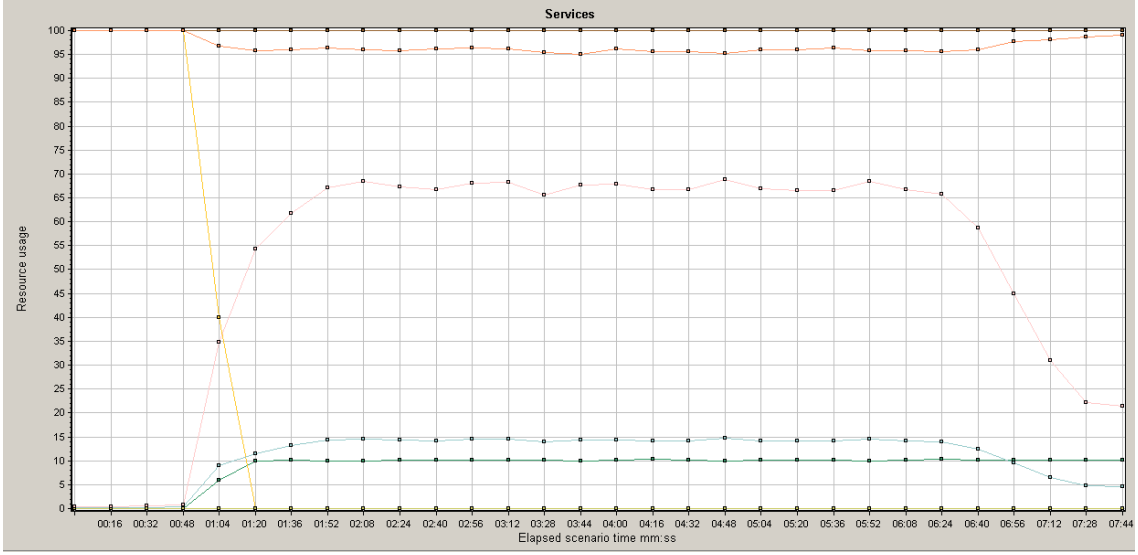

## Service Virtualization Services Graph

This graph displays a summary for Service Virtualization - Services.

| | |
|---|---|
| **X-axis** | The elapsed time from the beginning of the scenario run. |
| **Y-axis** | The number of resources used. |
| **Tips** | • To isolate the measurement with the most problems, it may be helpful to sort the legend window according to the average number of resources used. To sort the legend by average, double-click the **Average** column heading.<br>• To identify a measurement in the graph, you can select it. The corresponding line in the legend window is selected. |
| **Note** | To use this graph, you must first open a Service Virtualization project in the Controller scenario. |
| **See also** | Web Page Diagnostics Graph |

**Example**

Using the graph, you can track which resources were most problematic, and at which point(s) during the scenario the problem(s) occurred.




# Streaming State Statistics Graphs

The Streaming State Statistics graphs display statistics about the streaming videos from your scenario run.

> **Note:** For a list of video streaming statistics metrics, see "Streaming State Statistics Measurements" on page 1222.

| Graph | Description |
|---|---|
| **Streaming Time** | Displays the average start and buffering times of all the videos at any given moment of the scenario run. |
| **Streaming Network** | Displays the average download speed and video quality of all the videos at any given moment of the scenario run. |
| **Streaming Events** | Displays the number of play, pause, and seek events that occurred at any given moment of the scenario run. |
| **Streaming Video** | Displays the number of videos that played or buffered at any given moment of the scenario run. |

## See also:

- "Streaming State Statistics Monitoring" on page 1221

# System Resource Graphs

System Resource graphs display the system resource usage measured by the online monitors during the load test scenario run. These graphs require that you specify the resources you want to measure *before* running the scenario. For more information, see the section on online monitors in the LoadRunner Controller documentation.

## Server Resources Performance Counters

The following table describes the available counters:

| Monitor | Measurements | Description |
|---|---|---|
| **CPU Monitor** | Utilization | Measures CPU utilization. |
| **Disk Space Monitor** | Disk space | Measures the amount (in MB) free disk space and the percentage of disk space used. |
| **Memory Monitor** | MB free | Measures the amount of free memory (in MB). |
| | Pages/sec | Measures the number of virtual memory pages that are moved between main memory and disk storage. |
| | Percent used | Measures the percentage of memory and paging file space used. |
| **Services Monitor** | | Monitors processes locally or on remote systems. Can be used to verify that specific processes are running. |

## UNIX Resources Default Measurements

The following default measurements are available for UNIX machines:

| Measurement | Description |
|---|---|
| **Average load** | Average number of processes simultaneously in `Ready' state during the last minute. |
| **Collision rate** | Collisions per second detected on the Ethernet. |
| **Context switches rate** | Number of switches between processes or threads, per second. |
| **CPU utilization** | Percent of time that the CPU is utilized. |
| **Disk rate** | Rate of disk transfers. |

| Measurement | Description |
|---|---|
| **Incoming packets error rate** | Errors per second while receiving Ethernet packets. |
| **Incoming packets rate** | Incoming Ethernet packets per second. |
| **Interrupt rate** | Number of device interrupts per second. |
| **Outgoing packets errors rate** | Errors per second while sending Ethernet packets. |
| **Outgoing packets rate** | Outgoing Ethernet packets per second. |
| **Page-in rate** | Number of pages read to physical memory, per second. |
| **Page-out rate** | Number of pages written to pagefile(s) and removed from physical memory, per second. |
| **Paging rate** | Number of pages read to physical memory or written to pagefile(s), per second. |
| **Swap-in rate** | The rate by which disk content is swapped into the machine's memory in kBps. |
| **Swap-out rate** | The rate by which the machine's memory is being swapped out to disk in kBps. |
| **System mode CPU utilization** | Percent of time that the CPU is utilized in system mode. |
| **User mode CPU utilization** | Percent of time that the CPU is utilized in user mode. |

## Windows Resources Default Measurements

The following default measurements are available for Windows Resources:

| Object | Measurement | Description |
|---|---|---|
| **System** | **% Total Processor Time** | The average percentage of time that all the processors on the system are busy executing non-idle threads. On a multi-processor system, if all processors are always busy, this is 100%, if all processors are 50% busy this is 50% and if 1/4 of the processors are 100% busy this is 25%. It can be viewed as the fraction of the time spent doing useful work. Each processor is assigned an Idle thread in the Idle process which consumes those unproductive processor cycles not used by any other threads. |
| **Processor** | **% Processor Time** | The percentage of time that the processor is executing a non-idle thread. This counter was designed as a primary indicator of processor activity. It is calculated by measuring the time that the processor spends executing the thread of the idle process in each sample interval, and subtracting that value from 100%. (Each processor has an idle thread which consumes cycles when no other threads are ready to run.) It can be viewed as the percentage of the sample interval spent doing useful work. This counter displays the average percentage of busy time observed during the sample interval. It is calculated by monitoring the time the service was inactive, and then subtracting that value from 100%. |
| **System** | **File Data Operations/sec** | The rate at which the computer issues read and write operations to file system devices. This does not include File Control Operations. |
| **System** | **Processor Queue Length** | The instantaneous length of the processor queue in units of threads. This counter is always 0 unless you are also monitoring a thread counter. All processors use a single queue in which threads wait for processor cycles. This length does not include the threads that are currently executing. A sustained processor queue length greater than two generally indicates processor congestion. This is an instantaneous count, not an average over the time interval. |
| **Memory** | **Page Faults/sec** | This is a count of the page faults in the processor. A page fault occurs when a process refers to a virtual memory page that is not in its Working Set in the main memory. A page fault will not cause the page to be fetched from disk if that page is on the standby list (and hence already in main memory), or if it is in use by another process with which the page is shared. |
| **PhysicalDisk** | **% Disk Time** | The percentage of elapsed time that the selected disk drive is busy servicing read or write requests. |

| Object | Measurement | Description |
|---|---|---|
| **Memory** | **Pool Nonpaged Bytes** | The number of bytes in the non-paged pool, a system memory area where space is acquired by operating system components as they accomplish their appointed tasks. Non-paged pool pages cannot be paged out to the paging file. They remain in main memory as long as they are allocated. |
| **Memory** | **Pages/sec** | The number of pages read from the disk or written to the disk to resolve memory references to pages that were not in memory at the time of the reference. This is the sum of Pages Input/sec and Pages Output/sec. This counter includes paging traffic on behalf of the system cache to access file data for applications. This value also includes the pages to/from non-cached mapped memory files. This is the primary counter to observe if you are concerned about excessive memory pressure (that is, thrashing), and the excessive paging that may result. |
| **System** | **Total Interrupts/sec** | The rate at which the computer is receiving and servicing hardware interrupts. The devices that can generate interrupts are the system timer, the mouse, data communication lines, network interface cards, and other peripheral devices. This counter provides an indication of how busy these devices are on a computer-wide basis. See also **Processor:Interrupts/sec**. |
| **Objects** | **Threads** | The number of threads in the computer at the time of data collection. Notice that this is an instantaneous count, not an average over the time interval. A thread is the basic executable entity that can execute instructions in a processor. |
| **Process** | **Private Bytes** | The current number of bytes that the process has allocated that cannot be shared with other processes. |

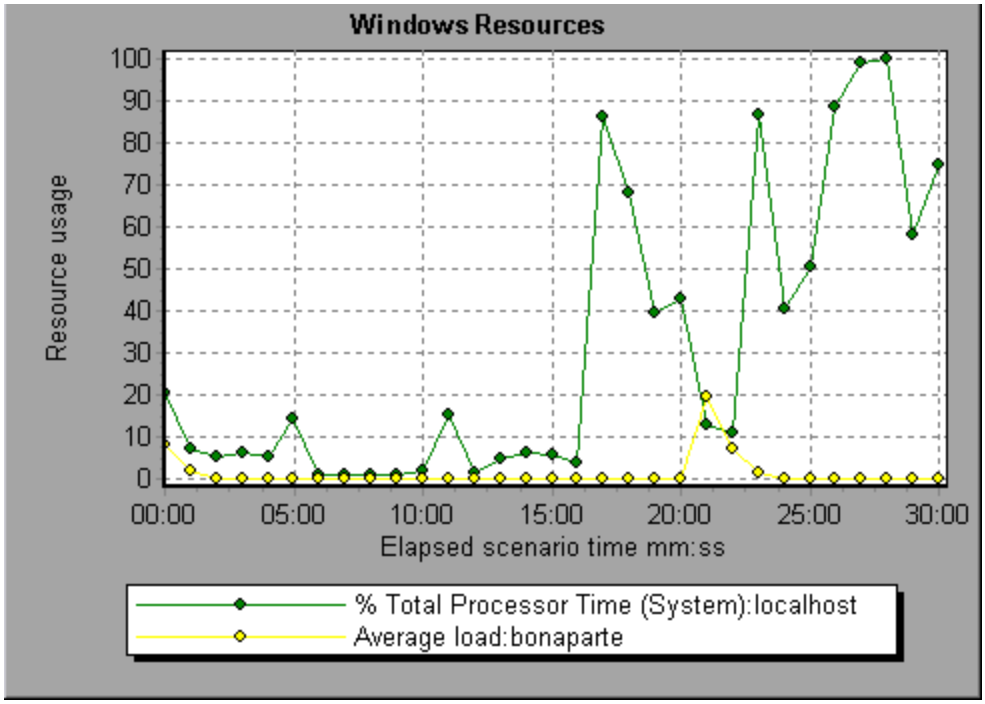
## Server Resources Graph

This graph shows the resources (CPU, disk space, memory, or services) used on remote UNIX servers measured during the load test scenario.

| | |
|---|---|
| **Purpose** | This graph helps you determine the impact of Vuser load on the various system resources. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The usage of resources on the UNIX server. |
| **See also** | "System Resource Graphs" on page 1468<br>"Server Resources Performance Counters" on page 1468 |

**Example**

In the following example, Windows resource utilization is measured during the load test scenario. It is shown as a function of the elapsed scenario time.



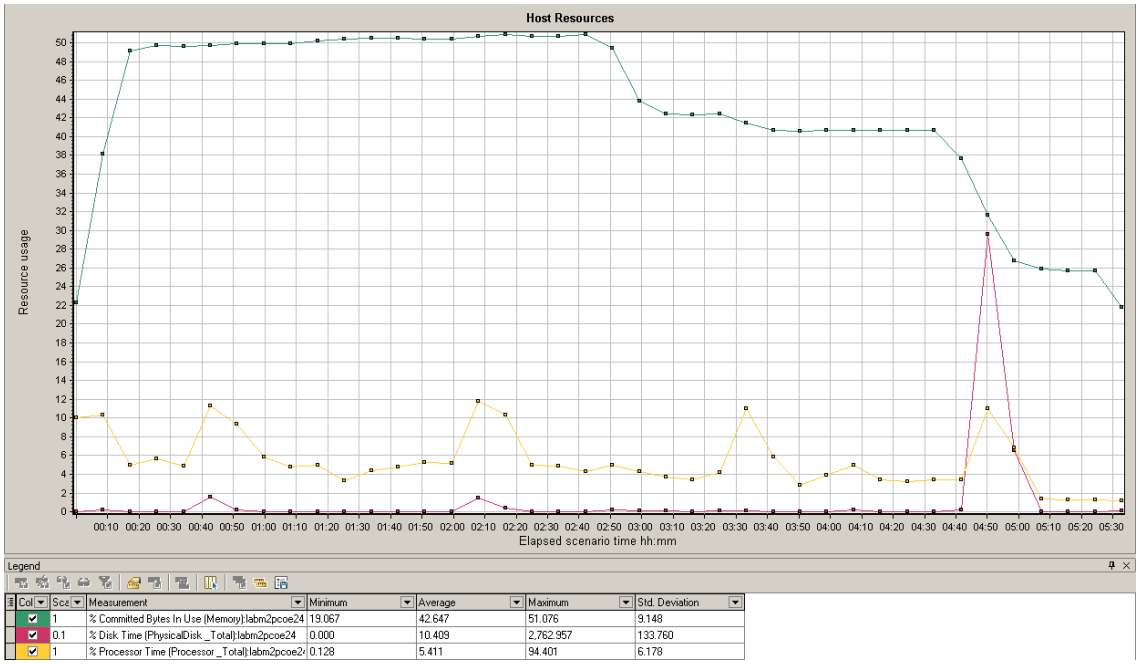

## Host Resources Graph

This graph displays a summary of the System Resources usage for each Windows based Performance Center host (Controller and load generators) measured during the load test scenario.

| | |
|---|---|
| **Purpose** | This graph helps you determine the impact of Vuser load on the various host resources. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The usage of resources on the Windows hosts. |
| **See also** | "System Resource Graphs" on page 1468 |

**Example**

In the following example, you can see a peak in the usage of Disk Time and Processor Time as the Memory Usage gets less towards the end of the load test.



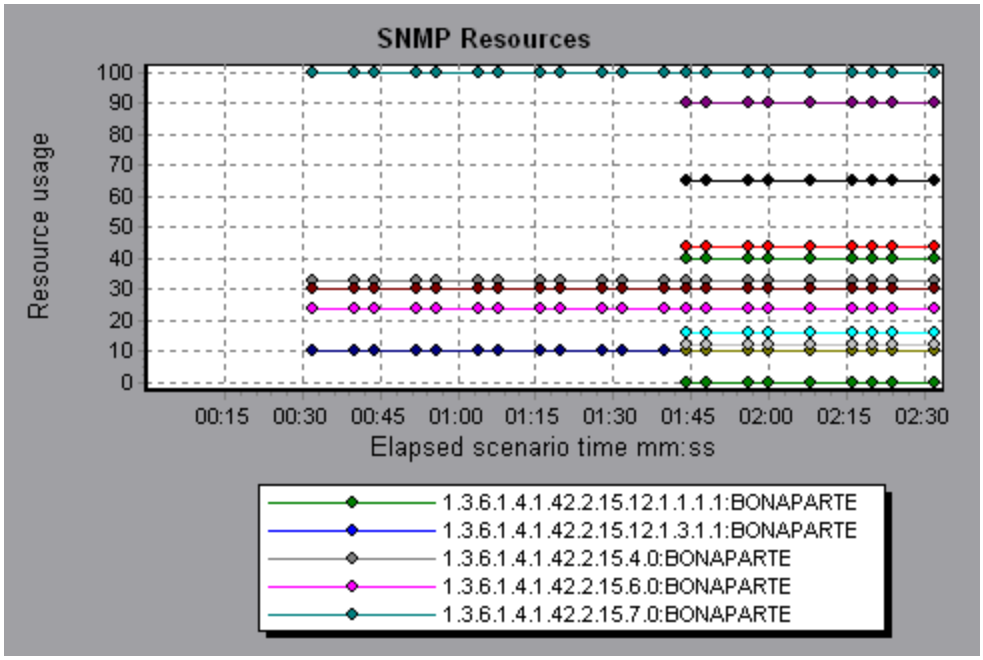

## SNMP Resources Graph

This graph shows statistics for machines running an SNMP agent, using the Simple Network Management Protocol (SNMP).

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The usage of resources on a machine running the SNMP agent. |
| **Note** | To obtain data for this graph, you need to enable the SNMP monitor (from Controller) and select the default measurements you want to display, before running the scenario. |
| **See also** | "System Resource Graphs" on page 1468 |

**Example**

In the following example SNMP measurements are displayed for a machine called bonaporte.



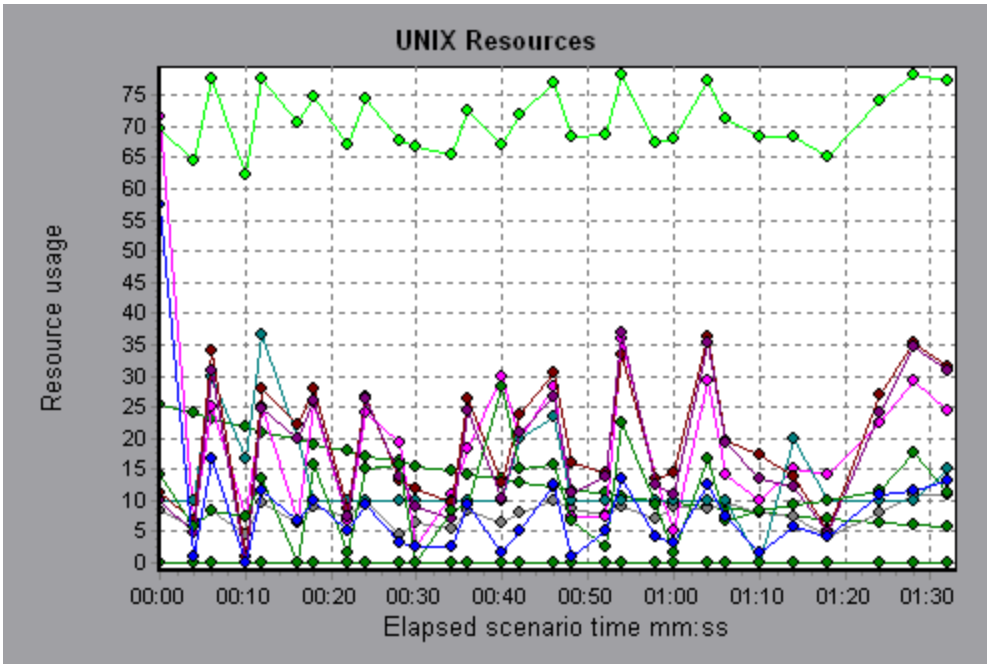

## UNIX Resources Graph

This graph shows the UNIX resources measured during the load test scenario. The UNIX measurements include: average load, collision rate, context switch rate, CPU utilization, incoming packets error rate, incoming packets rate, interrupt rate, outgoing packets error rate, outgoing packets rate, page-in rate, page-out rate, paging rate, swap-in rate, swap-out rate, system mode CPU utilization, and user mode CPU utilization.

| | |
|---|---|
| **Purpose** | This graph helps you determine the impact of Vuser load on the various system resources. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The usage of resources on the UNIX machine. |
| **Note** | To obtain data for this graph, you need to select the desired measurements for the online monitor (from Controller) before running the scenario. |
| **See also** | "UNIX Resources Default Measurements" on page 1468 |

**Example**

In the following example UNIX resources are measured during the load test scenario.



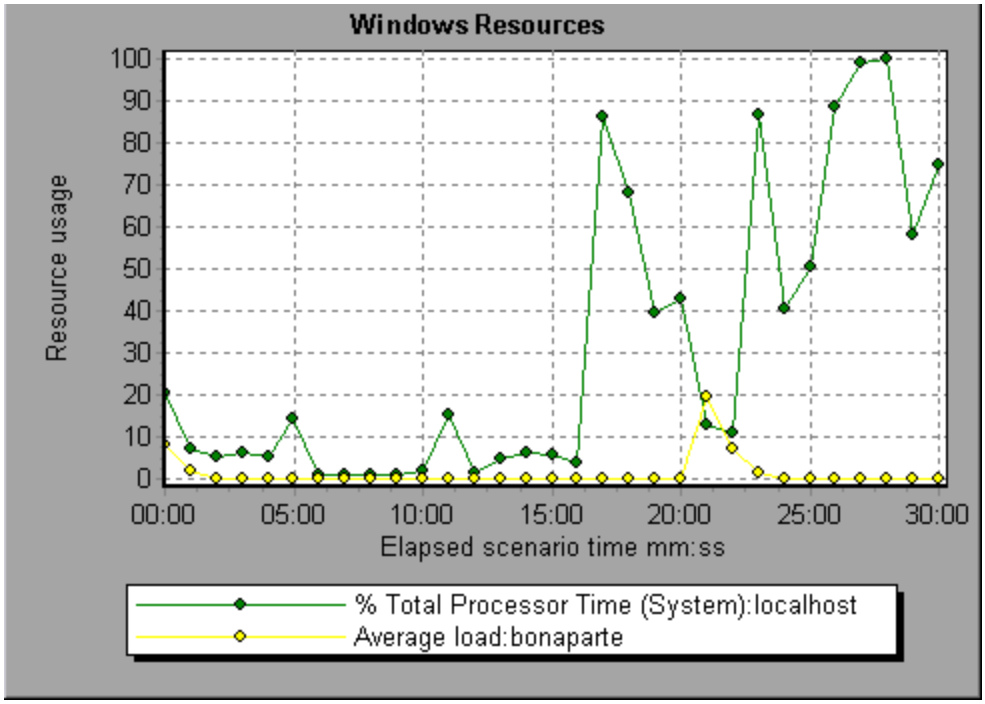

## Windows Resources Graph

This graph shows the Windows resources measured during the load test scenario. The Windows measurements correspond to the built-in counters available from the Windows Performance Monitor.

| | |
|---|---|
| **Purpose** | This graph helps you determine the impact of Vuser load on the various system resources. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The usage of resources on the Windows machine running the load test scenario. |
| **Note** | To obtain data for this graph, you need to select the desired measurements for the online monitor (from Controller) before running the scenario. |
| **See also** | "System Resource Graphs" on page 1468<br>"Windows Resources Default Measurements" on page 1469 |

**Example**

In the following example Windows resources are measured on the server running the load test scenario.




# Transaction Graphs

During load test scenario execution, Vusers generate data as they perform transactions. Analysis enables you to generate graphs that show the transaction performance and status throughout script execution.

In addition, when working with Network Virtualization, you can view the transaction response times per virtual location.

You can use additional Analysis tools such as merging and crossing results to understand your transaction performance graphs. You can also sort the graph information by transactions and the locations in which they were performed.

For more information, see the transaction graphs below.

## Transaction Breakdown Tree

The Transaction Breakdown Tree displays a tree view of the transactions and sub-transactions in the current session. From the tree, you can breakdown transactions and view the results of the breakdown in either the Average Transaction Response Time or Transaction Performance Summary graph.

| | |
|---|---|
| **To access** | In either the Average Transaction Response Time or Transaction Performance Summary graph, right-click in the graph and select **Show Transaction Breakdown Tree.** |
| **Important information** | After you breakdown a transaction, you can return to the original transaction graph by reapplying the global filter (**File > Set Global Filter**) or by undoing your breakdown actions using **Edit > Undo Last Action**. |

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **<Right-click menu>** | • **Break Down From Highest Level.** Displays data for the highest level hierarchical path of a transaction.<br>• **Break Down <transaction name>.** Displays data for the sub-transactions in the Average Transaction Response Time or Transaction Performance Summary graph.<br>• **Show Only <transaction name>.** Displays data only for the selected transaction/sub-transaction.<br>• **Web Page Diagnostics for <page name>.** Displays a breakdown of the Web page(s) included in a transaction or sub-transaction in the Web Page Diagnostics graphs. For details, see "Web Page Diagnostics Graphs" on page 1494. |

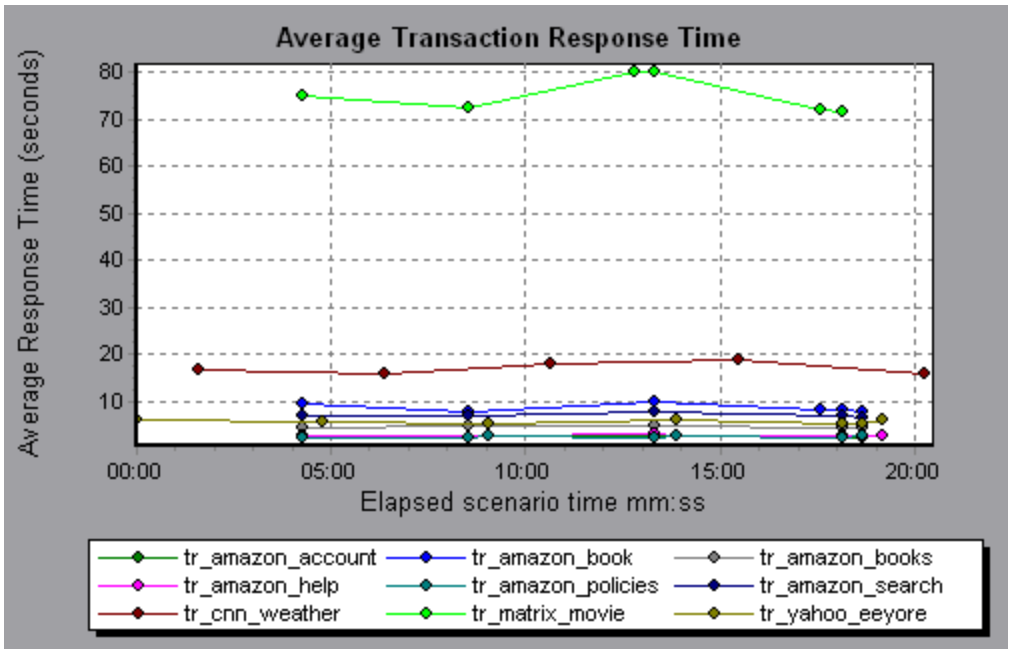
## Average Transaction Response Time Graph

This graph displays the average time taken to perform transactions during each second of the load test scenario run.

| | |
|---|---|
| **Purpose** | If you have defined acceptable minimum and maximum transaction performance times, you can use this graph to determine whether the performance of the server is within the acceptable range. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Average response time (in seconds) of each transaction |
| **Breakdown options** | • **Transaction Breakdown.** You can view a breakdown of a transaction by right-clicking the transaction in the graph and selecting **Show Transaction Breakdown Tree**.<br>In the Transaction Breakdown Tree, right-click the transaction you want to break down, and select **Break Down <transaction name>**. The Average Transaction Response Time graph displays data for the sub-transactions.<br>For more details, see "Transaction Breakdown Tree" on the previous page.<br>• **Web Page Breakdown.** To view a breakdown of the Web page(s) included in a transaction or sub-transaction, right-click it and select **Web Page Diagnostics for <transaction name>**.<br>For more information on the Web Page Diagnostics graphs, see "Web Page Diagnostics Graphs" on page 1494. |

| | |
|---|---|
| **Tips** | • **Granularity.** This graph is displayed differently for each granularity. The lower the granularity, the more detailed the results.<br>However, it may be useful to view the results with a higher granularity to study the overall Vuser behavior throughout the scenario.<br>For example, using a low granularity, you may see intervals when no transactions were performed. However, by viewing the same graph with a higher granularity, you will see the graph for the overall transaction response time.<br>For more information on setting the granularity, see "Getting more information from Graph Data" on page 1333.<br>• **Compare with Running Vusers.** You can compare the Average Transaction Response Time graph to the Running Vusers graph to see how the number of running Vusers affects the transaction performance time.<br>For example, if the Average Transaction Response Time graph shows that performance time gradually improved, you can compare it to the Running Vusers graph to see whether the performance time improved due to a decrease in the Vuser load. |
| **Note** | By default, only transactions that passed are displayed. |
| **See also** | "Transaction Graphs" on page 1476 |

## Example

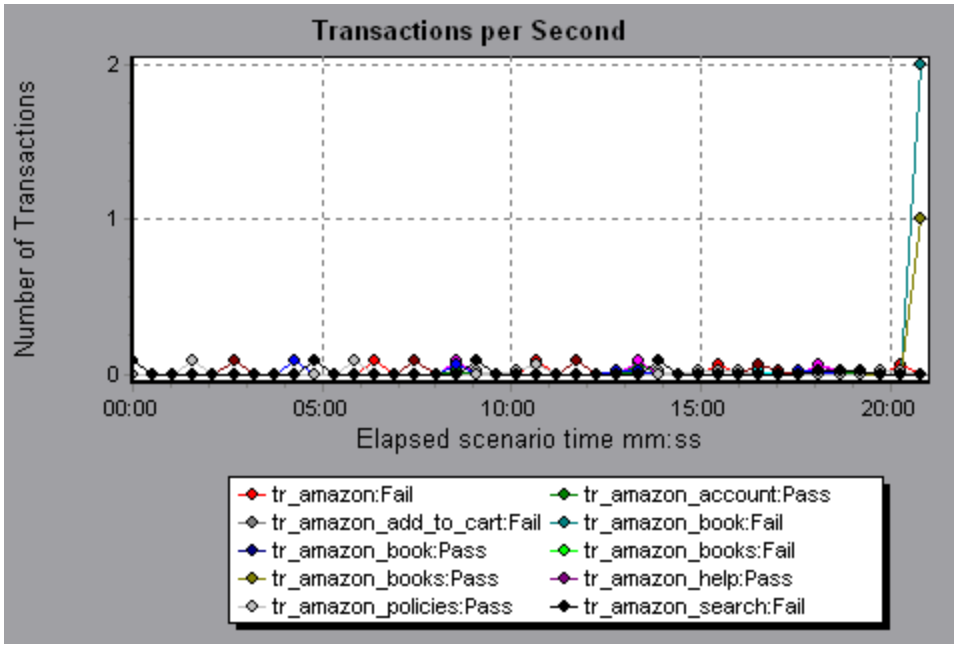


# Transactions per Second Graph

This graph displays, for each transaction, the number of times it passed, failed, and stopped during each second of a load test scenario run.

| | |
|---|---|
| **Purpose** | Helps you determine the actual transaction load on your system at any given moment. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of transactions performed during each second of the scenario run. |
| **Tips** | Compare with the **Average Transaction Response Time** Graph. Doing this helps you to determine the effect of the number of transactions upon the transaction performance time. |
| **See also** | "Transaction Graphs" on page 1476 |

## Example



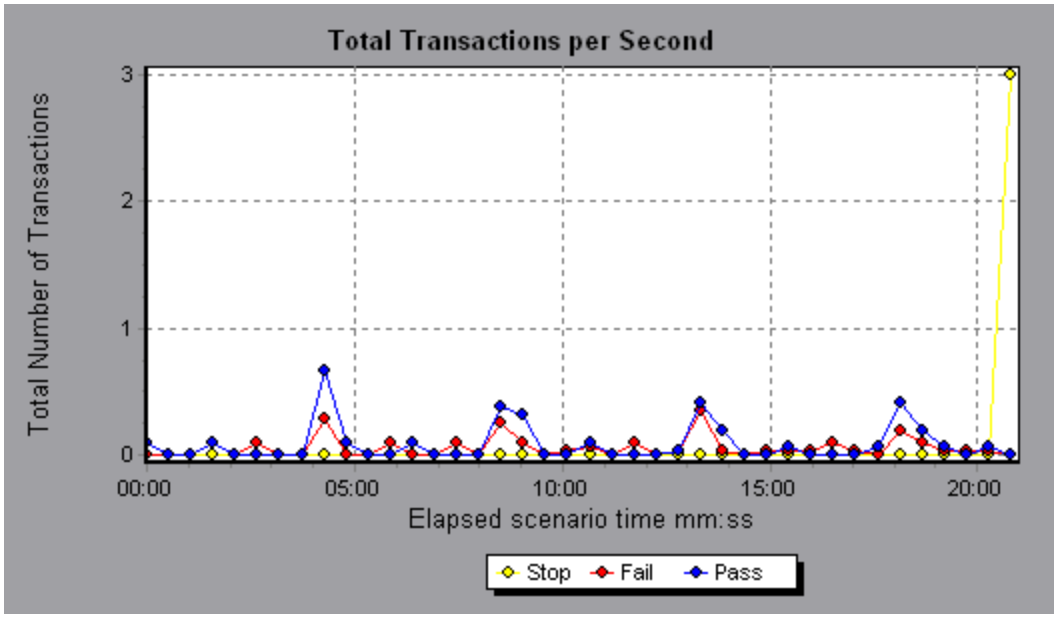

# Total Transactions per Second Graph

This graph displays the total number of transactions that passed, the total number of transactions that failed, and the total number of transactions that were stopped, during each second of a load test scenario run.

| | |
|---|---|
| **Purpose** | Helps you determine the actual transaction load on your system at any given moment. |

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Total number of transactions performed during the scenario run. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example

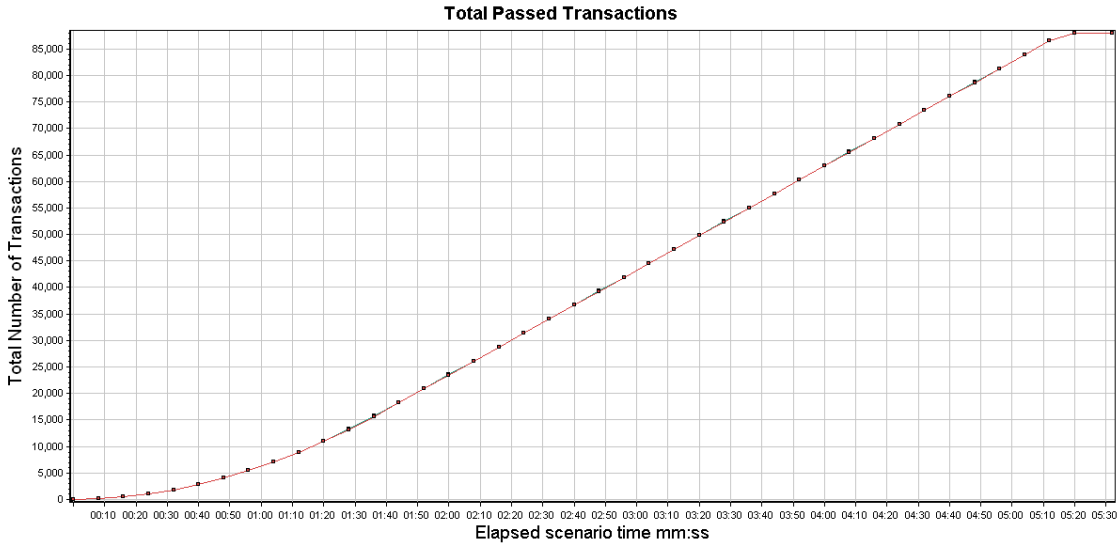



## Total Passed Transactions Graph

This graph displays the cumulative sum of passed transactions over the scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The transactions that passed during the scenario run. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example

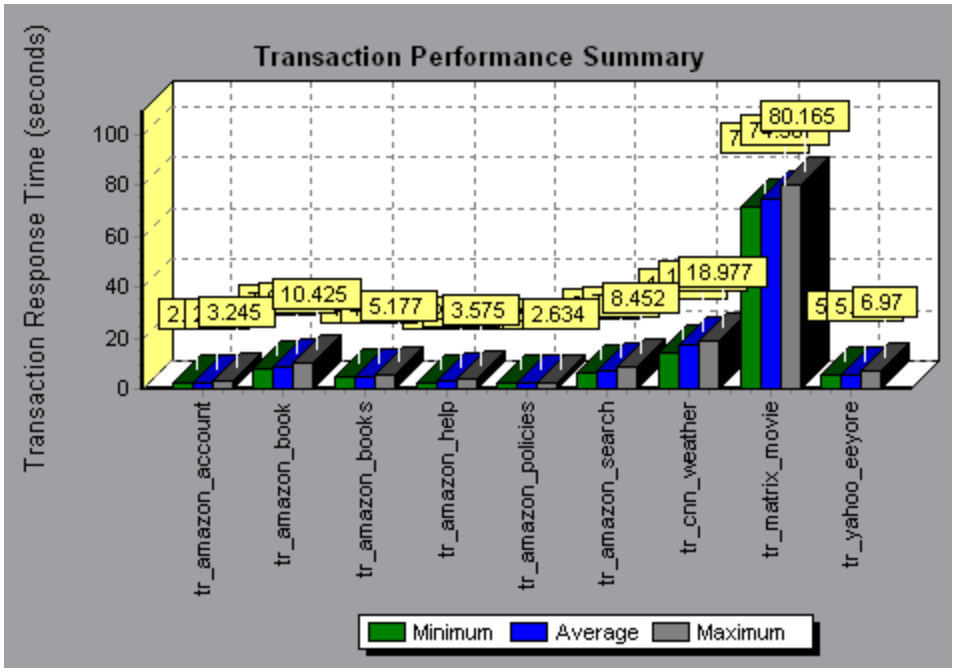



## Transaction Performance Summary Graph

This graph displays the minimum, maximum and average performance time for all the transactions in the load test scenario.

| | |
|---|---|
| **X-axis** | Name of the transaction. |
| **Y-axis** | Response time—rounded off to the nearest second—of each transaction. |
| **Breakdown options** | • **Transaction Breakdown.** You can view breakdown of a transaction in the Transaction Performance Summary graph by right-clicking the transaction in the graph and selecting **Show Transaction Breakdown Tree**. In the Transaction Breakdown Tree, right-click the transaction you want to break down, and select **Break Down <transaction name>**.<br>The Transaction Performance Summary graph displays data for the sub-transactions. For more details, see "Transaction Breakdown Tree" on page 1476.<br>• **Web Page Breakdown.** To view a breakdown of the Web page(s) included in a transaction or sub-transaction, right-click it and select **Web Page Diagnostics for <transaction name>**. For more, see "Web Page Diagnostics Graphs" on page 1494. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example



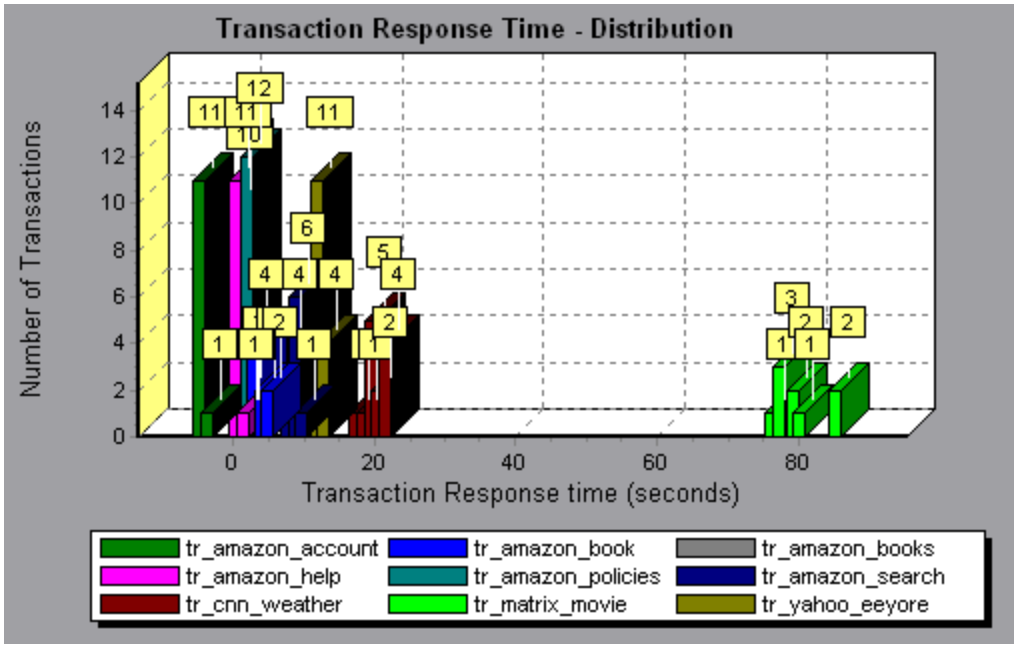

## Transaction Response Time (Distribution) Graph

This graph displays the distribution of the time taken to perform transactions in a load test scenario.

| | |
|---|---|
| **Purpose** | If you have defined acceptable minimum and maximum transaction performance times, you can use this graph to determine whether the performance of the server is within the acceptable range. |
| **X-axis** | Transaction response time (rounded down to the nearest second). |
| **Y-axis** | Number of transactions executed during the scenario. |
| **Tips** | Compare with **Transaction Performance Summary** Graph to see how the average performance was calculated. |
| **Note** | This graph can only be displayed as a bar graph. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example

In the following example, most of the transactions had a response time of less than 20 seconds.



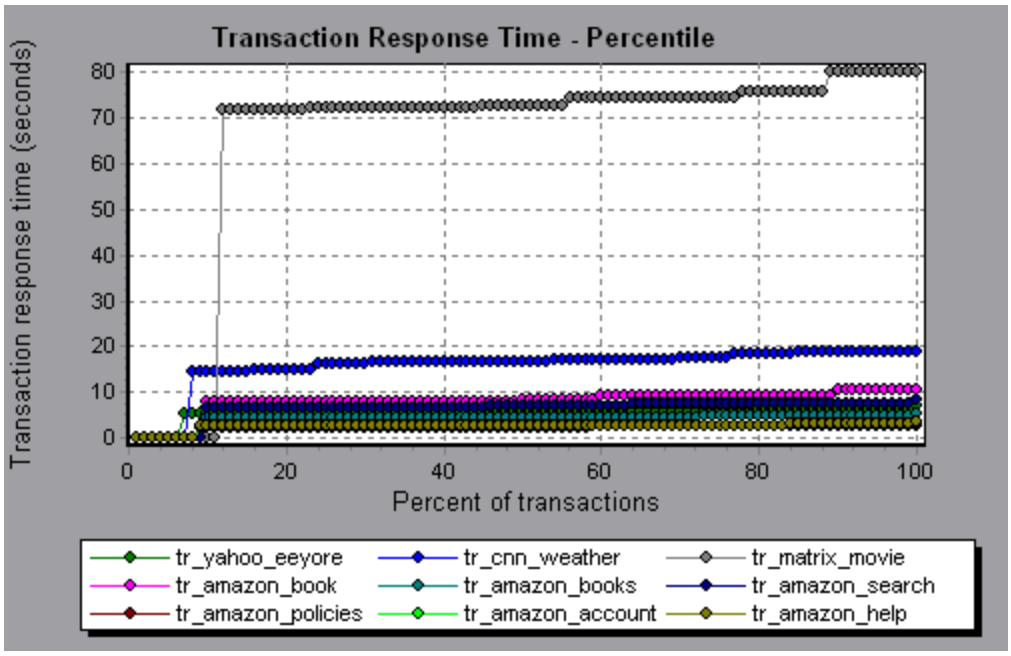

## Transaction Response Time (Percentile) Graph

This graph analyzes the percentage of transactions that were performed within a given time range.

| | |
|---|---|
| **Purpose** | Helps you determine the percentage of transactions that met the performance criteria defined for your system. In many instances, you need to determine the percent of transactions with an acceptable response time. The maximum response time may be exceptionally long, but if most transactions have acceptable response times, the overall system is suitable for your needs. |
| **X-axis** | Percentage of the total number of transactions measured during the load test scenario run. |
| **Y-axis** | Maximum transaction response time (in seconds).<br><br>**Note:** Analysis approximates the transaction response time for each available percentage of transactions. The y-axis values, therefore, may not be exact. |
| **Tips** | Compare with the **Average Response Time** Graph.<br><br>A high response time for several transactions may raise the overall average. However, if the transactions with a high response time occurred less than five percent of the time, that factor may be insignificant. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example

In the following example, fewer than 20 percent of the tr_matrix_movie transactions had a response time less than 70 seconds.

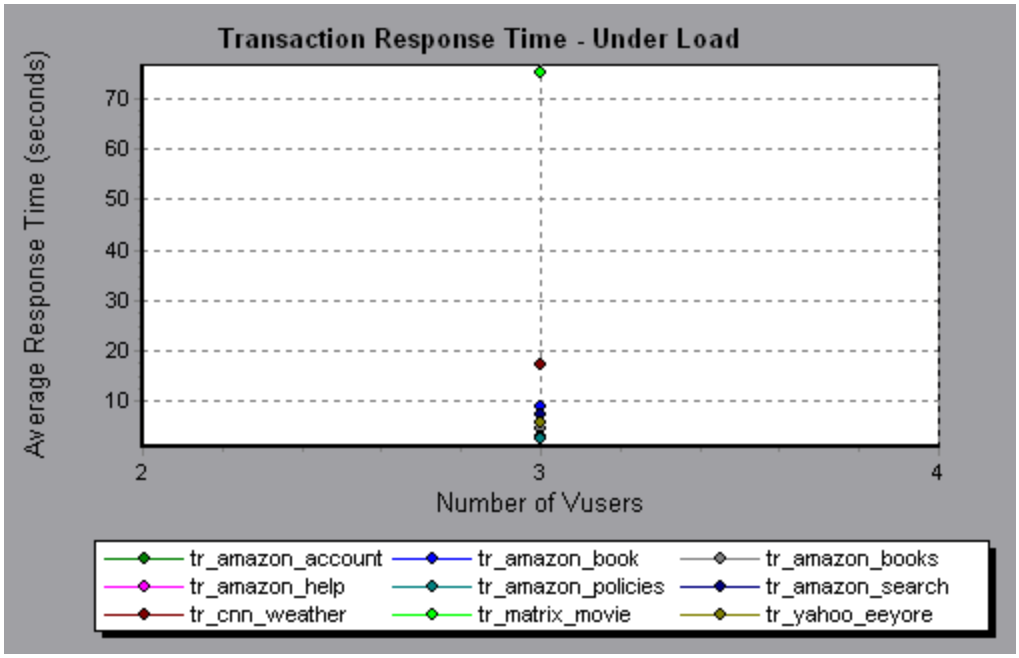



## Transaction Response Time (Under Load) Graph

This graph is a combination of the Running Vusers and Average Transaction Response Time graphs and indicates transaction times relative to the number of Vusers running at any given point during the load test scenario.

| | |
|---|---|
| **Purpose** | Helps you view the general impact of Vuser load on performance time and is most useful when analyzing a scenario with a gradual load. |
| **X-axis** | Number of running Vusers |
| **Y-axis** | Average response time (in seconds) of each transaction. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example




## Transaction Response Time by Location Graph

This graph shows the average transaction response time for each of the virtual locations with which the transactions were performed.

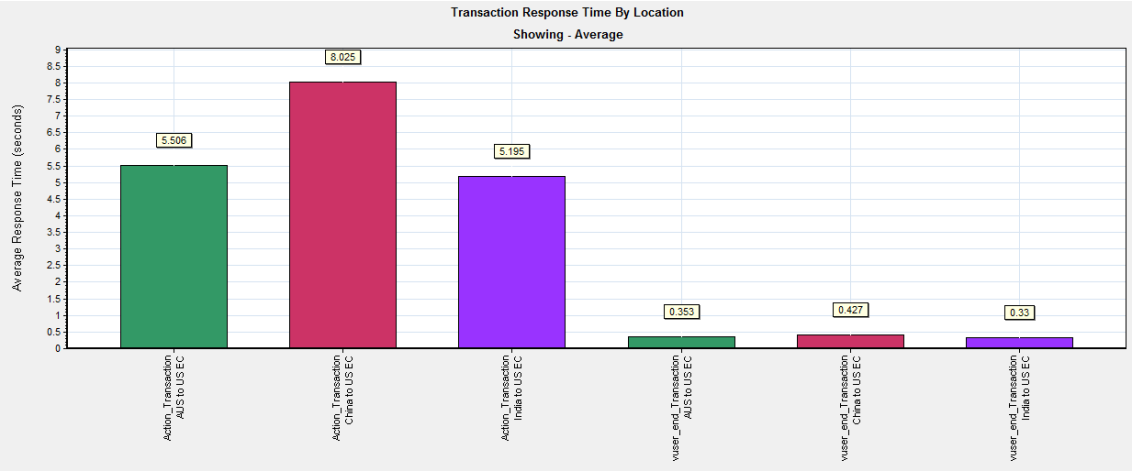
This graph is used in conjunction with Network Virtualization. Using Network Virtualization, you set up a scenario that runs Vusers on several virtual locations. This graph lets you compare the transaction response times for the various virtual locations. For details, see "Network Virtualization Integration" on page 1237Network Virtualization Integration.

| | |
|---|---|
| **Purpose** | Helps you view the general impact of Vuser load on performance time for each virtual location.<br><br>The bars in the graph are color-coded by virtual location – the same color is displayed for all transactions that use the same virtual location. |
| **X-axis** | Transactions performed with a specific virtual location. |
| **Y-axis** | Average response time (in seconds) for each transaction, per virtual location. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example

The following example shows the average transaction response times for three virtual locations. **Action_Transaction** took 8.025 seconds to perform (on average) when using the **China to US EC** virtual

location, the longest time for any of the three virtual locations.

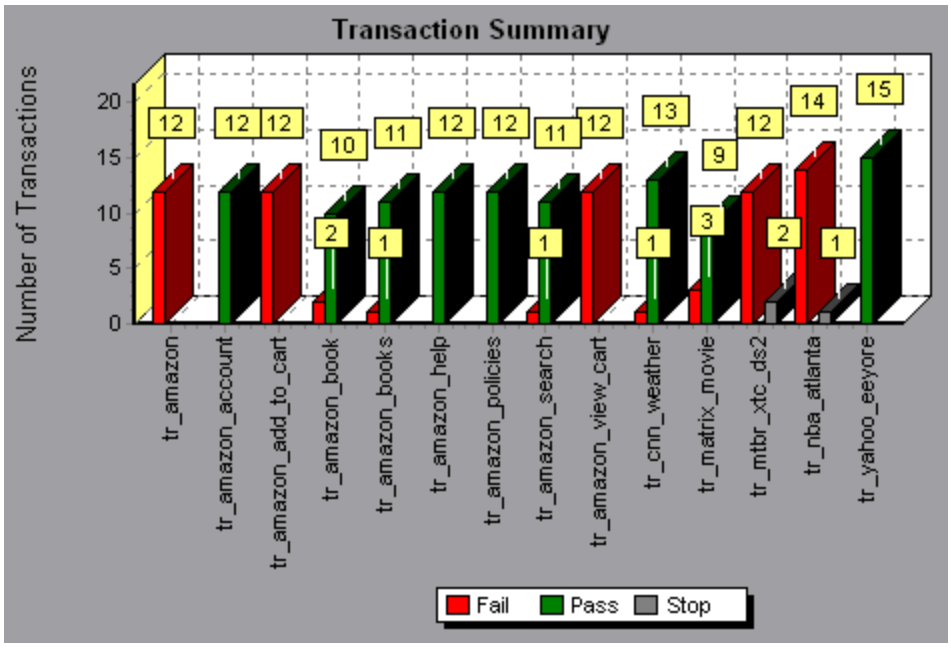



## Transaction Summary Graph

This graph summarizes the number of transactions in the load test scenario that failed, passed, stopped, and ended in error.

| | |
|---|---|
| **X-axis** | Name of the transaction. |
| **Y-axis** | Number of transactions performed during the scenario run. |
| **See also** | "Transaction Graphs" on page 1476 |

### Example



# TruClient - Native Mobile Graphs

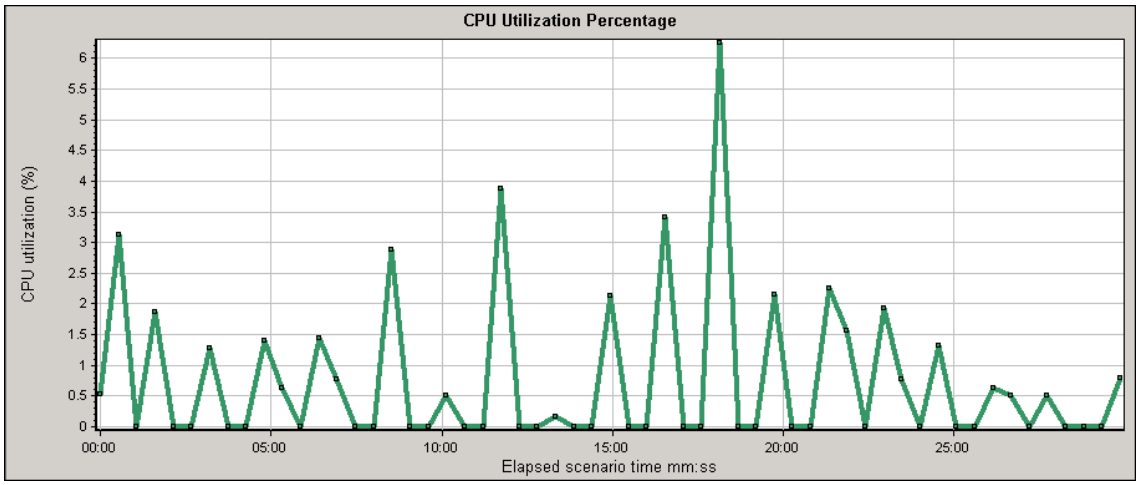
## TruClient CPU Utilization Percentage Graph

This graph displays the percentage of the CPU utilized during the test run for TruClient Native Mobile Vuser scripts.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of CPU utilized by an application. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | The percentage of the CPU utilized during the test run. |

### Example

In the following example, the CPU utilization peaked to approximately 6% at 18 minutes into the test run.



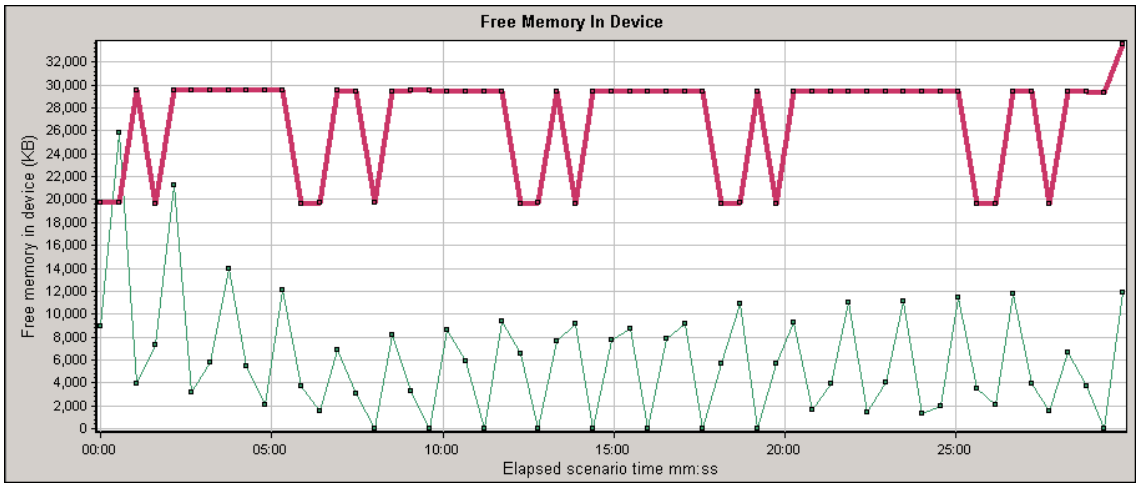

## TruClient Free Memory In Device Graph

This graph displays the free memory on a mobile device as a function of time, for TruClient Native Mobile scripts.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of memory available on the device during the test run. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | The amount of free memory in KBs. |

## Example

In the following example, the graph shows a free memory of over 33 MBs, at 30 minutes into the test run for one of the transactions.



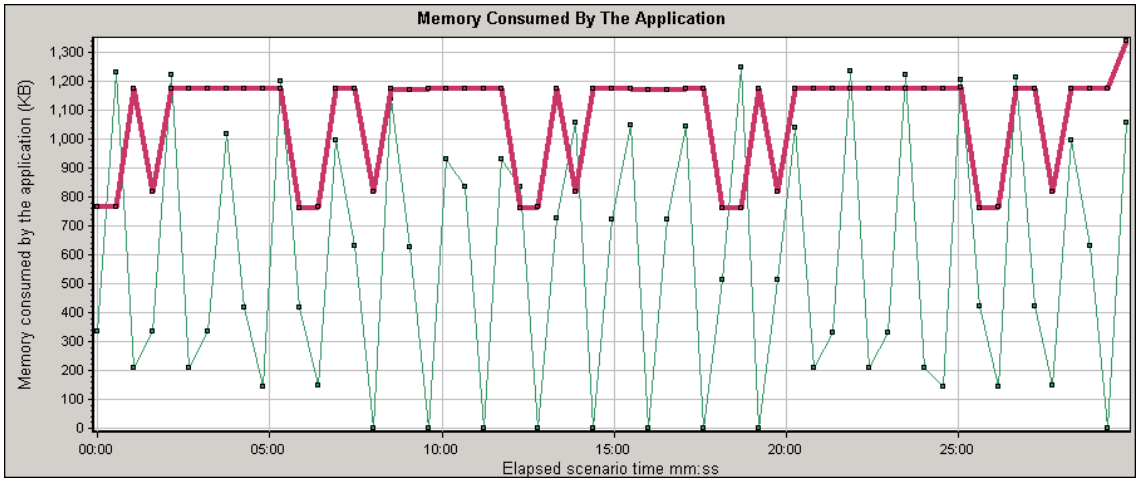

# TruClient Memory Consumed by Application Graph

This graph displays the memory consumed by the application, as a function of time.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of memory used by the application. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | The memory consumed by the application in KBs. |

## Example

In the following example, the memory consumption peaked to 1337 KBs at 30 minutes into the test, for one of the transactions.



# User-Defined Data Point Graphs

## User-Defined Data Point Graphs Overview

The User-Defined Data Point graphs display the values of user-defined data points. You define a data point in your Vuser script by inserting an **lr_user_data_point** function at the appropriate place (**user_data_point** for GUI Vusers and **lr.user_data_point** for Java Vusers).

```
Action1()
{
    lr_think_time(1);
    lr_user_data_point ("data_point_1",1);
    lr_user_data_point ("data_point_2",2);
    return 0;
}
```

For Vuser protocols that support the graphical script representations such as Web and Oracle NCA, you insert a data point as a User Defined step. Data point information is gathered each time the script executes the function or step. For more information about data points, refer to the Function Reference.

Data points, like other Analysis data, are aggregated every few seconds, resulting in less data points shown on the graph than actually recorded. For more information, see "Changing the Granularity of the Data" on page 1330.

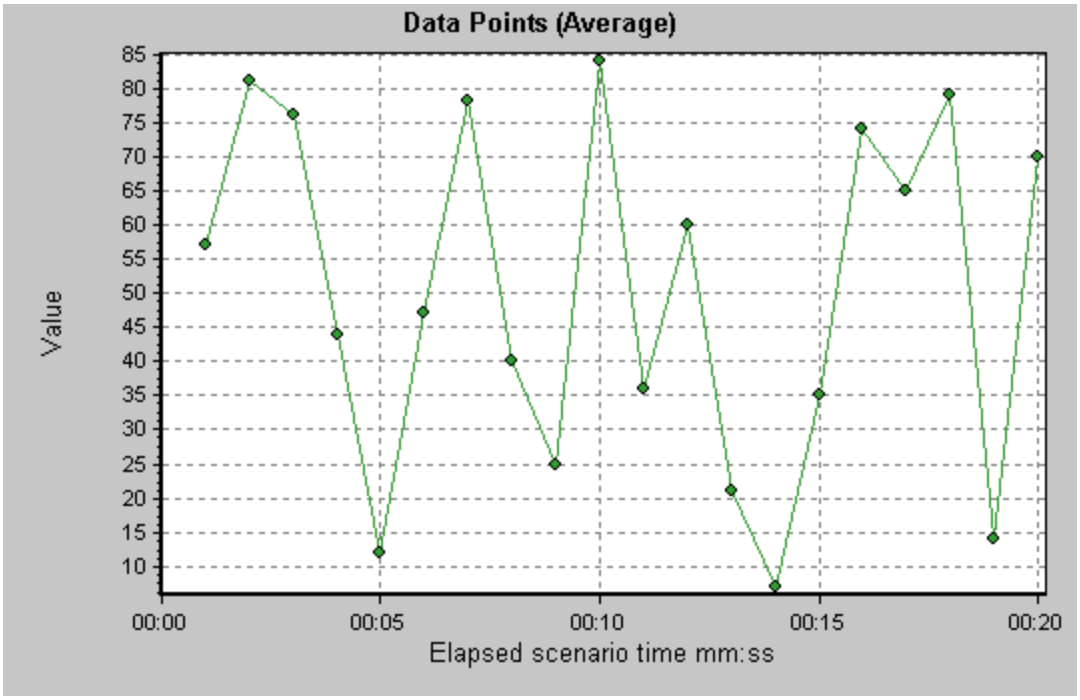
## Data Points (Average) Graph

This graph shows the average values that were recorded for user-defined data points during the load test scenario run.

| | |
|---|---|
| **Purpose** | This graph is typically used in cases where the actual value of the measurement is required. Suppose that each Vuser monitors CPU utilization on its machine and records it as a data point. In this case, the actual recorded value of CPU utilization is required. The Average graph displays the average value recorded throughout the scenario. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The average values of the recorded data point statements. |
| **See also** | "User-Defined Data Point Graphs Overview" above |

**Example**

In the following example, the CPU utilization is recorded as the data point **user_data_point_val_1**. It is shown as a function of the elapsed scenario time.




## Data Points (Sum) Graph

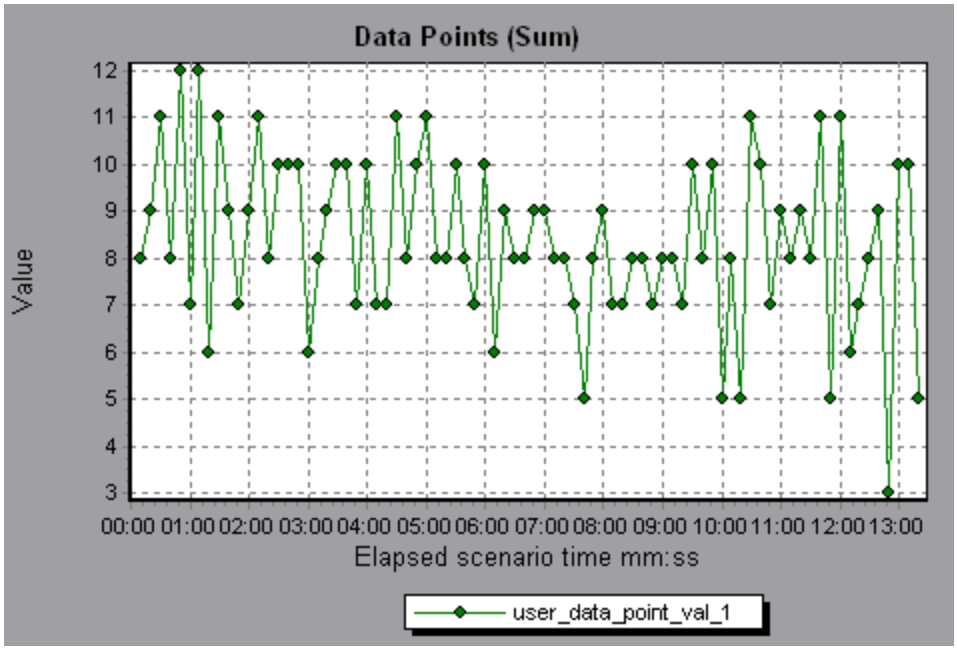
This graph shows the sum of the values for user-defined data points throughout the load test scenario run.

This graph typically indicates the total amount of measurements which all Vusers are able to generate. For example, suppose only a certain set of circumstances allow a Vuser to call a server. Each time it does, a data point is recorded. In this case, the Sum graph displays the total number of times that Vusers call the function.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The sum of the recorded data point values. |
| **See also** | "User-Defined Data Point Graphs Overview" on the previous page |

**Example**

In the following example, the call to the server is recorded as the data point user_data_point_val_1. It is shown as a function of the elapsed scenario time.




# Vuser Graphs

During load test scenario execution, Vusers generate data as they perform transactions. The Vuser graphs let you determine the overall behavior of Vusers during the scenario. They display the Vuser states, the number of Vusers that completed the script, and rendezvous statistics. Use these graphs in conjunction with Transaction graphs to determine the effect of the number of Vusers on transaction response time. For more information about Transaction graphs, see "Transaction Graphs" on page 1476.

## Rendezvous Graph (Vuser Graphs)

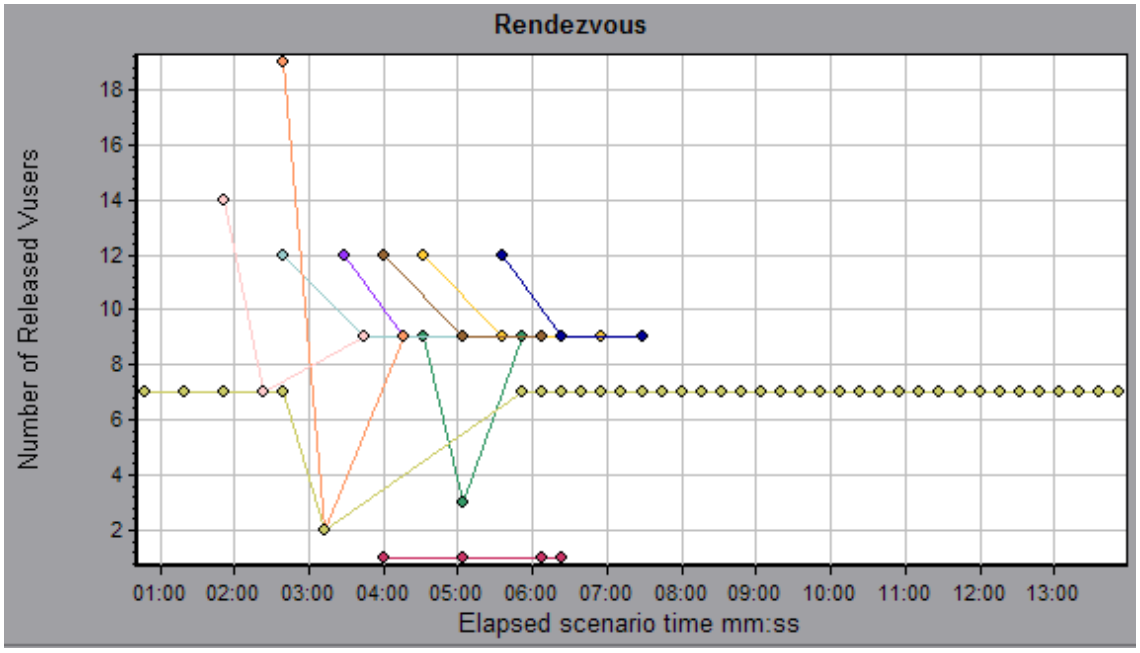
During a scenario run, you can instruct multiple Vusers to perform tasks simultaneously by using **rendezvous points**. A rendezvous point creates intense user load on the server and enables LoadRunner to measure server performance under load. For more information about using rendezvous points, see the *Virtual User Generator User Guide*.

This graph indicates when Vusers were released from rendezvous points, and how many Vusers were released at each point.

| | |
|---|---|
| **Purpose** | Helps you understand transaction performance times. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of Vusers that were released from the rendezvous. |
| **See also** | "Vuser Graphs" above |

> **Tip:** Compare this to the **Average Transaction Response Time** graph. When you so this, you can see how the load peak created by a rendezvous influences transaction times.

**Example**

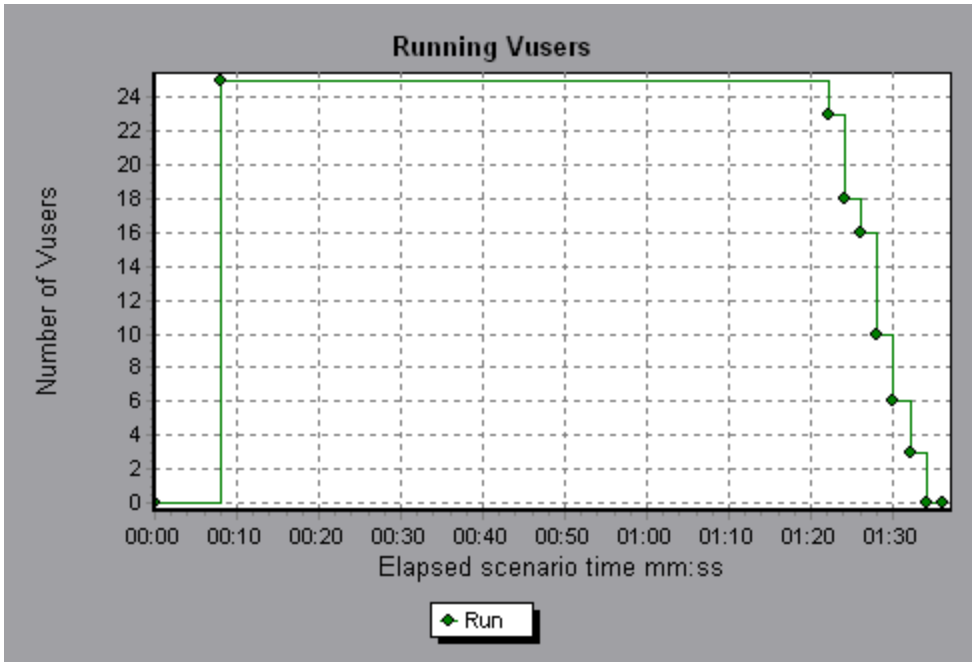



## Running Vusers Graph

This graph displays the number of Vusers (or active JMeter threads) that executed scripts and their status during each second of the test.

| | |
|---|---|
| **Purpose** | Helps you determine the Vuser load on your server at any given moment. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of Vusers in the scenario. |
| **Note** | By default, this graph only shows the Vusers with a **Run** status. To view another Vuser status, set the filter conditions to the desired status. For more information, see "Filtering and Sorting Graph Data" on page 1342. |
| **See also** | "Vuser Graphs" on the previous page |

**Example**

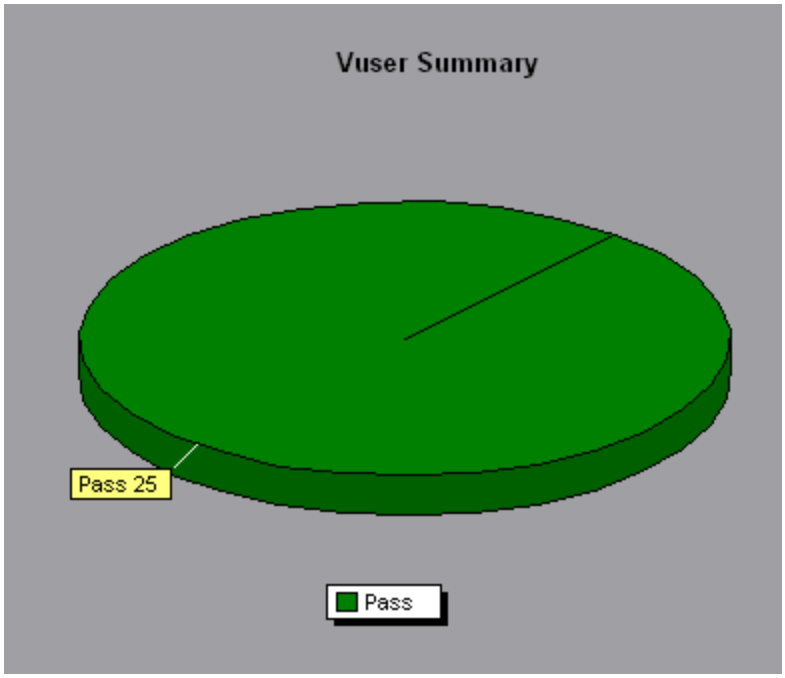



## Vuser Summary Graph

This graph displays a summary of Vuser performance.

| | |
|---|---|
| **Purpose** | Lets you view the number of Vusers that successfully completed the load test scenario run relative to those that did not. |
| **Note** | This graph can only be viewed as a pie. |
| **See also** | "Vuser Graphs" on page 1491 |

**Example**




# Web Page Diagnostics Graphs

## Web Page Diagnostics Tree View Overview

The Web Page Diagnostics tree view displays a tree view of the transactions, sub-transactions, and Web pages for which you can view Web Page Diagnostics graphs. For more information about Web Page Diagnostics graphs, see "Web Page Diagnostics Graph" on page 1498.

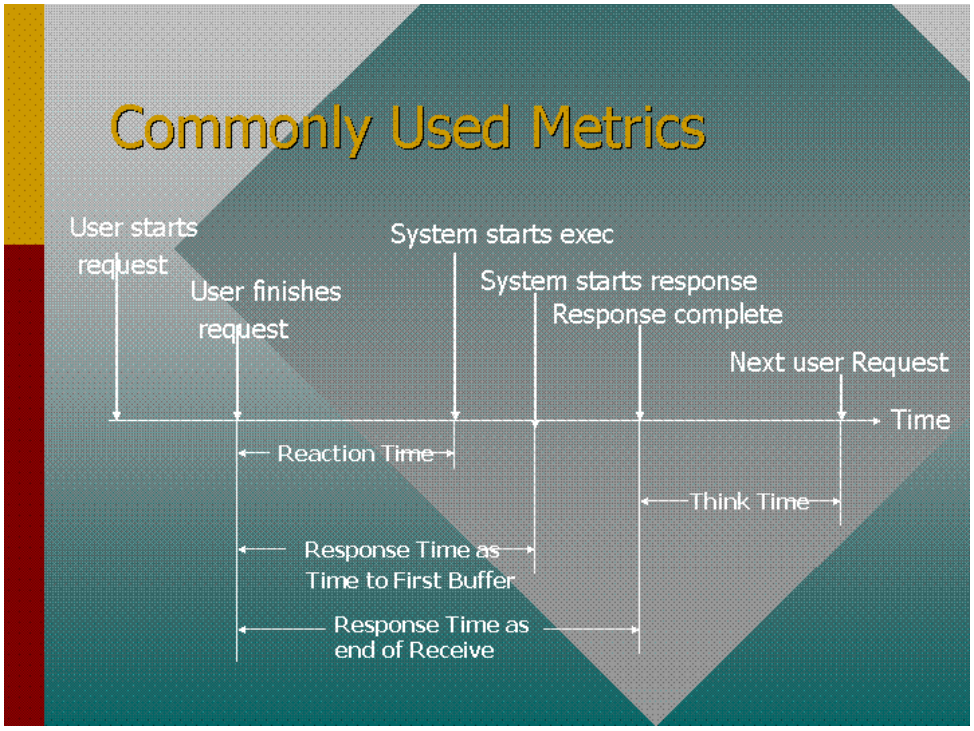
The Web Page Diagnostics graphs enable you to assess whether transaction response times were affected by page content. Using the Web Page Diagnostics graphs, you can analyze problematic elements—for example, images that download slowly, or broken links—of a Web site.

## Web Page Diagnostics Graphs Overview

Web Page Diagnostics graphs provide you with performance information for each monitored Web page in your script. You can view the download time of each page in the script and its components, and identify at what point during download time problems occurred. In addition, you can view the relative download time and size of each page and its components. Analysis displays both average download time and download time over time data.

You correlate the data in the Web Page Diagnostics graphs with data in the Transaction Performance Summary and Average Transaction Response Time graphs in order to analyze why and where problems are occurring, and whether the problems are network- or server-related.

The following diagram illustrates the sequence of events from the time an HTTP request is sent:




**Note:** Because server time is being measured from the client, network time may influence this measurement if there is a change in network performance from the time the initial HTTP request is sent until the time the first buffer is sent. The server time displayed, therefore, is estimated server time and may be slightly inaccurate.

You begin analyzing the Transaction Performance Summary and Average Transaction Response Time graphs with the Web Page Diagnostics graph, which displays the average download time (in seconds) for each monitored Web page during each second of the load test scenario run. The x-axis represents the elapsed time from the beginning of the scenario run. The y-axis represents the average download time (in seconds) for each Web page.

These graphs can also be used for analyzing mobile applications using the Mobile Application - HTTP/HTML protocol.

In order for Analysis to generate Web Page Diagnostics graphs, you must enable the Web Page Diagnostics feature in Controller before running your scenario.

1. In Controller, from the menu, choose **Diagnostics > Configuration** and select the **Enable the following diagnostics** check box.
2. In the **Offline Diagnostics** section, if the button to the right of **Web Page Diagnostics (Max. Vuser Sampling: 10%)** says **Enable**, click it.

**Note:** When preparing a Web HTTP/HTML Vuser script for which you want to perform Web diagnostics, it is recommended that you create an **HTML-based script** (using the **Recording** tab

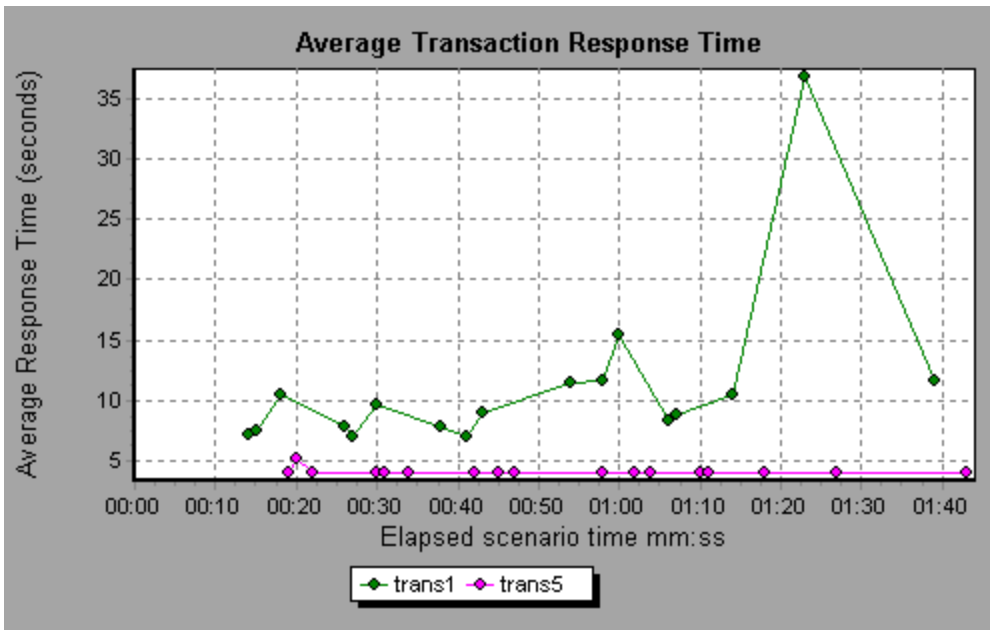
in the Recording Options).

### See also:

- "Recording - Overview" on page 154

## How to View the Breakdown of a Transaction

The Web Page Diagnostics graphs are most commonly used to analyze a problem detected in the Transaction Performance Summary or Average Transaction Response Time graphs. For example, the Average Transaction Response Time graph below demonstrates that the average transaction response time for the trans1 transaction was high.




Using the Web Page Diagnostics graphs, you can pinpoint the cause of the delay in response time for the trans1 transaction.

This task describes how to breakdown a transaction.

1. Right-click **trans1** and select **Web Page Diagnostics for trans1**. The Web Page Diagnostics graph opens and the Web Page Diagnostics tree appear. An icon appears next to the page name indicating the page content. See "Web Page Diagnostics Content Icons" on the next page.
2. In the Web Page Diagnostics tree, right-click the problematic page you want to break down, and select **Break Down <component name>**. Alternatively, select a page in the **Select Page to Break Down** box that appears under the Web Page Diagnostics graph. The Web Page Diagnostics graph for that page appears.

> **Note:** You can open a browser displaying the problematic page by right-clicking the page in the Web Page Diagnostics tree and selecting **View page in browser**.

3. Select one of the following available breakdown options:
   - **Download Time.** Displays a table with a breakdown of the selected page's download time. The size of each page component (including the component's header) is displayed. See the "Page Download Time Breakdown Graph" on page 1502 for more information about this display.
   - **Component (Over Time).** Displays the "Page Component Breakdown (Over Time) Graph" on page 1501 for the selected Web page.
   - **Download Time (Over Time).** Displays the "Page Download Time Breakdown (Over Time) Graph" on page 1504 for the selected Web page.
   - **Time to First Buffer (Over Time).** Displays the "Time to First Buffer Breakdown (Over Time) Graph" on page 1508 for the selected Web page.

   To display the graphs in full view, click the button. You can also access these graphs, as well as additional Web Page Diagnostics graphs, from the Open a New Graph dialog box.

## Web Page Diagnostics Content Icons

The following icons appear in the Web Page Diagnostics tree. They indicate the HTTP content of the page.

| Name | Description |
|---|---|
| | **Transaction**. Specifies that the ensuing content is part of the transaction. |
| | **Page Content**. Specifies that the ensuing content, which may include text, images, and so on, is all part of one logical page. |
| | **Text content.** Textual information. Plain text is intended to be displayed as-is. Includes HTML text and style sheets. |
| | **Multipart content.** Data consisting of multiple entities of independent data types. |
| | **Message content.** An encapsulated message. Common subtypes are news, or external-body which specifies large bodies by reference to an external data source. |
| | **Application content.** Some other kind of data, typically either uninterpreted binary data or information to be processed by an application. An example subtype is Postscript data. |
| | **Image content.** Image data. Two common subtypes are the jpeg and gif format. |

| Name | Description |
|---|---|
| | **Resource content.** Other resources not listed above. Also, content that is defined as "not available" is likewise included. |

## Web Page Diagnostics Graph

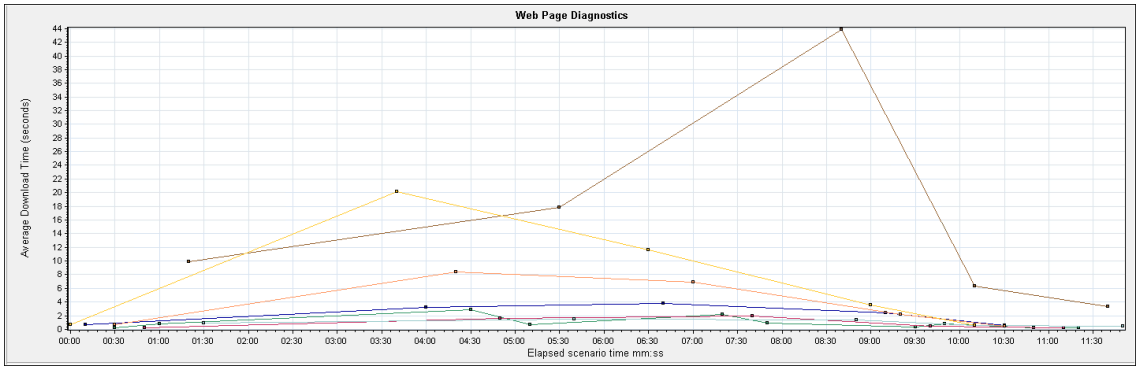
The Web Page Diagnostics graph provides you with performance information for each monitored Web page in your script. You can view the download time of each page in the script and its components, and identify at what point during download time problems occurred. In addition, you can view the average download time of each page and its components.

| | |
|---|---|
| **Purpose** | This graph enables you to determine at what point during scenario execution a network or server problem occurred, that may have affected access to the Web page. |
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | The download time (in seconds) for each Web page in the download process. |
| **Tips** | • Choose a page in the **Select Page to Break Down** drop-down box.<br>• To isolate the most problematic components, you can sort the legend window according to the average number of seconds taken to download a component. To sort the legend by average, double-click the **Average** column heading. |
| **Diagnostic Options** | You can choose one of the following options to drill down on the results. For sample graphs, see below.<br>• **Download Time** - as a bar graph<br>• **Component (Over Time)** - as a line graph<br>• **Download Time (Over Time)** - as an area graph<br>• **Time to First Buffer (Over Time)** - as an area graph |
| **See also** | "Web Page Diagnostics Tree View Overview" on page 1494 |

### Examples

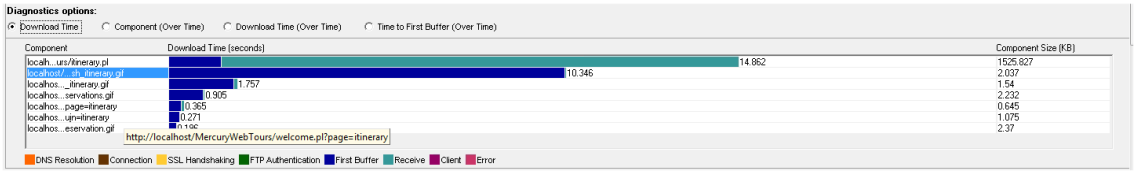
#### Average Download Time

This graph enables you to monitor the download time during the scenario execution, to determine at what point network or server problems occurred.



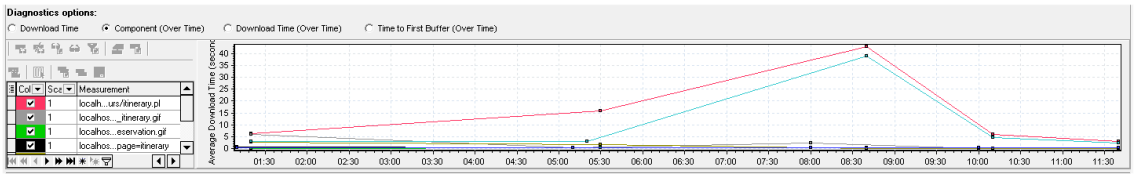

### Download Time

In the following example, the download time for the **itinerary.pl** page was the greatest during the **Receive** stage.



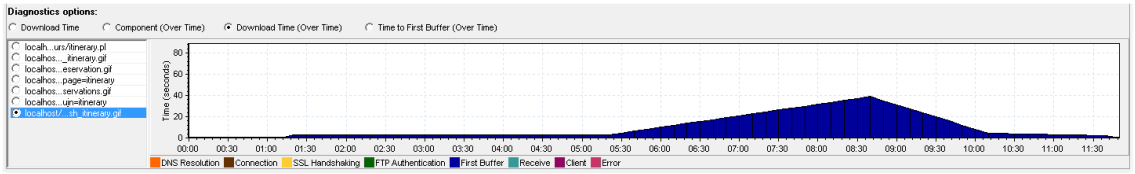

### Component(Over Time)

In the following example, the download time for the **itinerary.pl** component was the greatest at approximately 8:40 into the scenario.



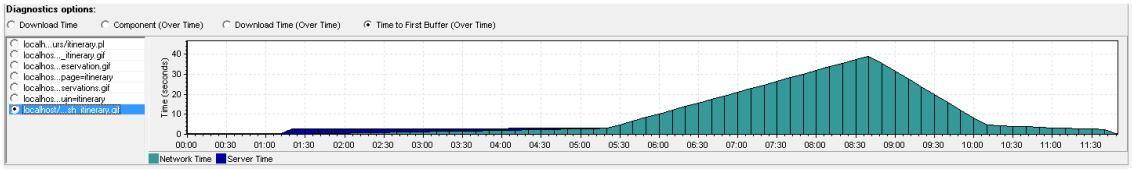

### Download Time (Over Time)

The following graph shows the download time for the **itinerary.pl** page as an area graph.




### Time to First Buffer (Over Time)

In the following example, the download time for the **splash_itinerary.gif** file was the greatest approximately 8:40 into the scenario.



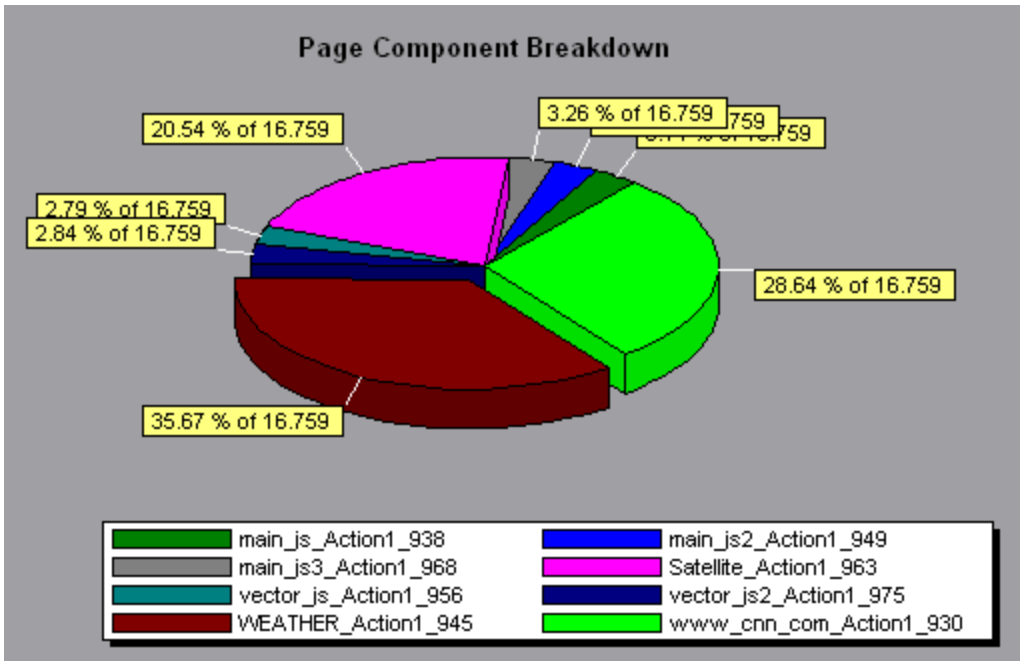

# Page Component Breakdown Graph

This graph displays the average download time (in seconds) for each Web page and its components.

| | |
|---|---|
| **Breakdown options** | To ascertain which components caused the delay in download time, you can break down the problematic URL by double-clicking it in the Web Page Diagnostics tree. |
| **Tips** | To isolate problematic components, it may be helpful to sort the legend according to the average number of seconds taken to download a component. To sort the legend by average, click the **Graph's Average** column. |
| **Note** | The graph can only be viewed as a pie. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

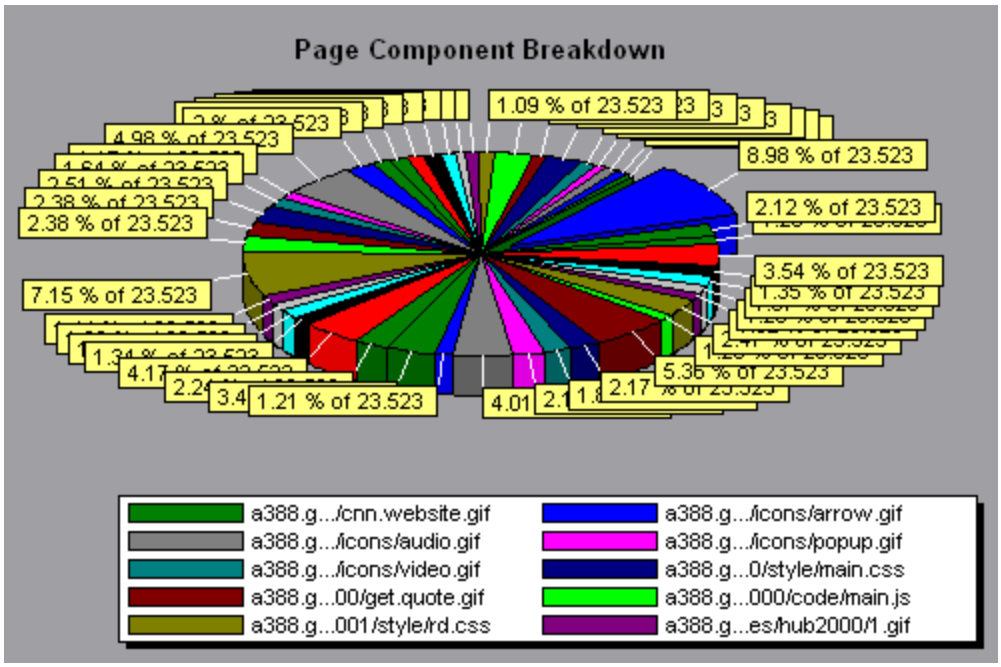
**Example**

The following graph demonstrates that the main cnn.com URL took 28.64% of the total download time, compared to 35.67% for the www.cnn.com/WEATHER component.




**Example**

The graph shows that the main cnn.com/WEATHER component took the longest time to download (8.98% of the total download time).



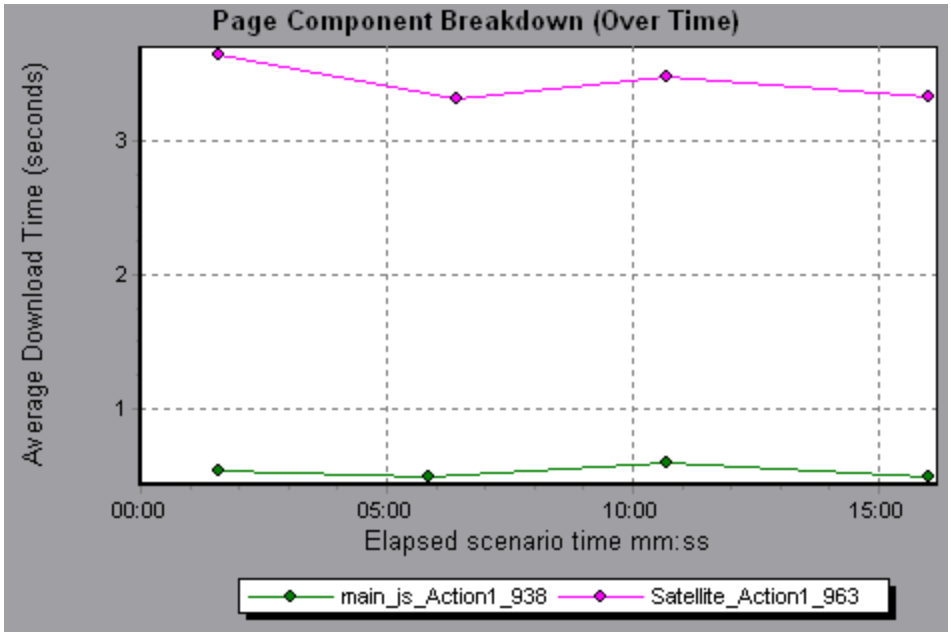

## Page Component Breakdown (Over Time) Graph

This graph displays the average response time (in seconds) for each Web page and its components during each second of the load test scenario run.

| | |
|---|---|
| **X-axis** | The elapsed time from the beginning of the scenario run. |
| **Y-axis** | The average response time (in seconds) for each component. |
| **Tips** | • To isolate the most problematic components, it may be helpful to sort the legend window according to the average number of seconds taken to download a component. To sort the legend by average, double-click the **Average** column heading.<br>• To identify a component in the graph, you can select it. The corresponding line in the legend window is selected. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

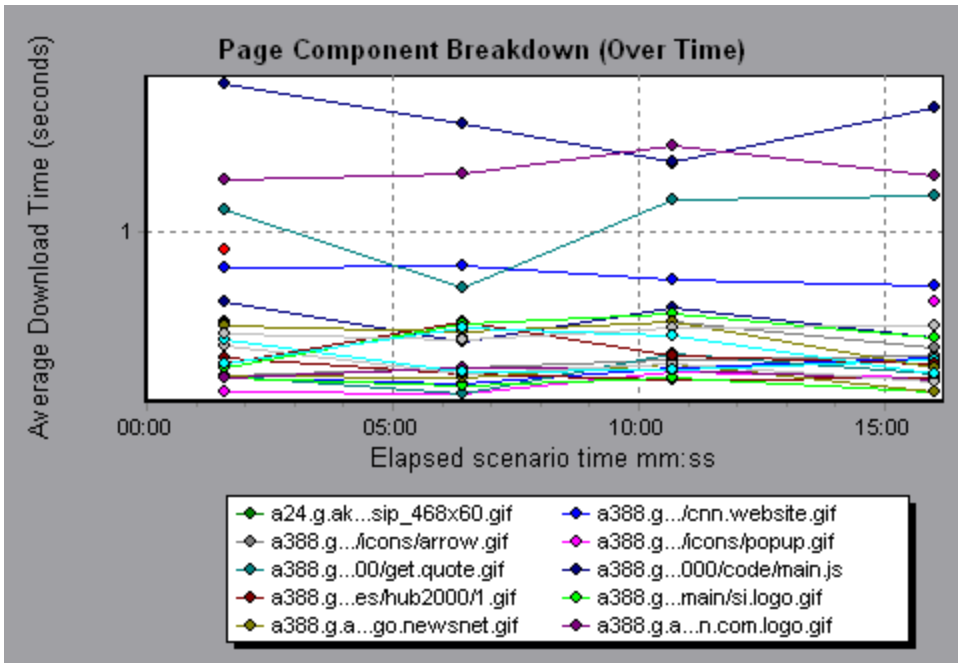
**Example**

The following graph demonstrates that the response time for Satellite_Action1_963 was significantly greater, throughout the scenario, than the response time for main_js_Action1_938.




**Example**

Using the graph, you can track which components of the main component were most problematic, and at which point(s) during the scenario the problem(s) occurred.



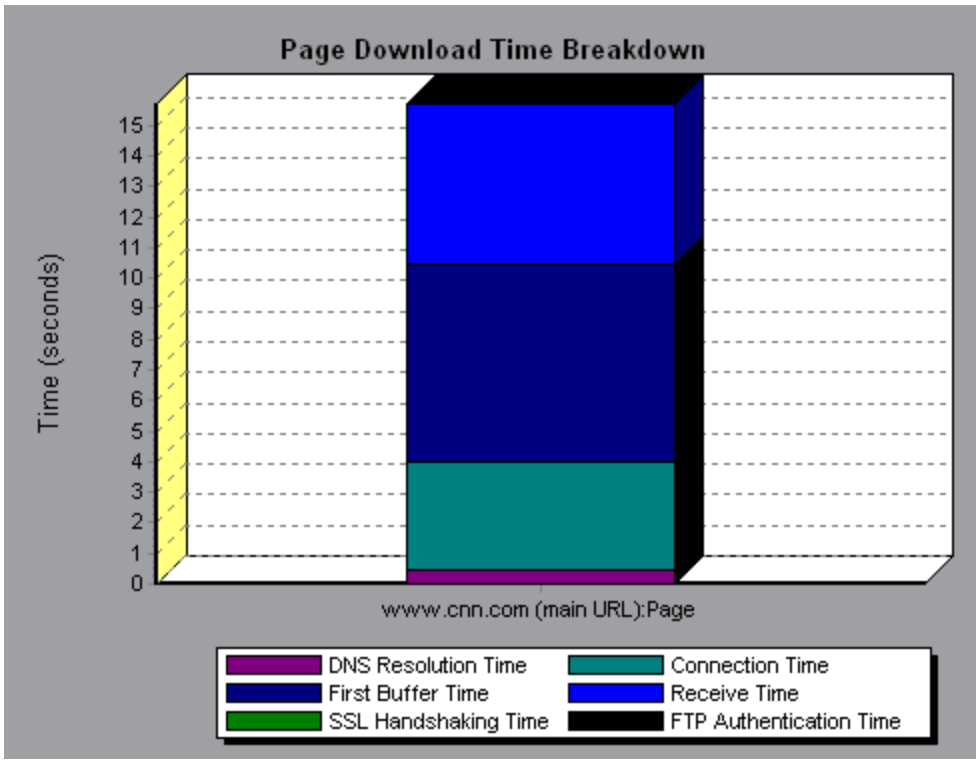

## Page Download Time Breakdown Graph

This graph displays a breakdown of each page component's download time.

| | |
|---|---|
| **Purpose** | Enables you to determine whether slow response times are being caused by network or server errors during Web page download. |
| **Breakdown options** | For breakdown options, see "Page Download Time Breakdown Graph Breakdown Options" on page 1506.<br><br>**Note:** Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the Connection Time for www.cnn.com is the sum of the Connection Time for each of the page's components. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

**Example**

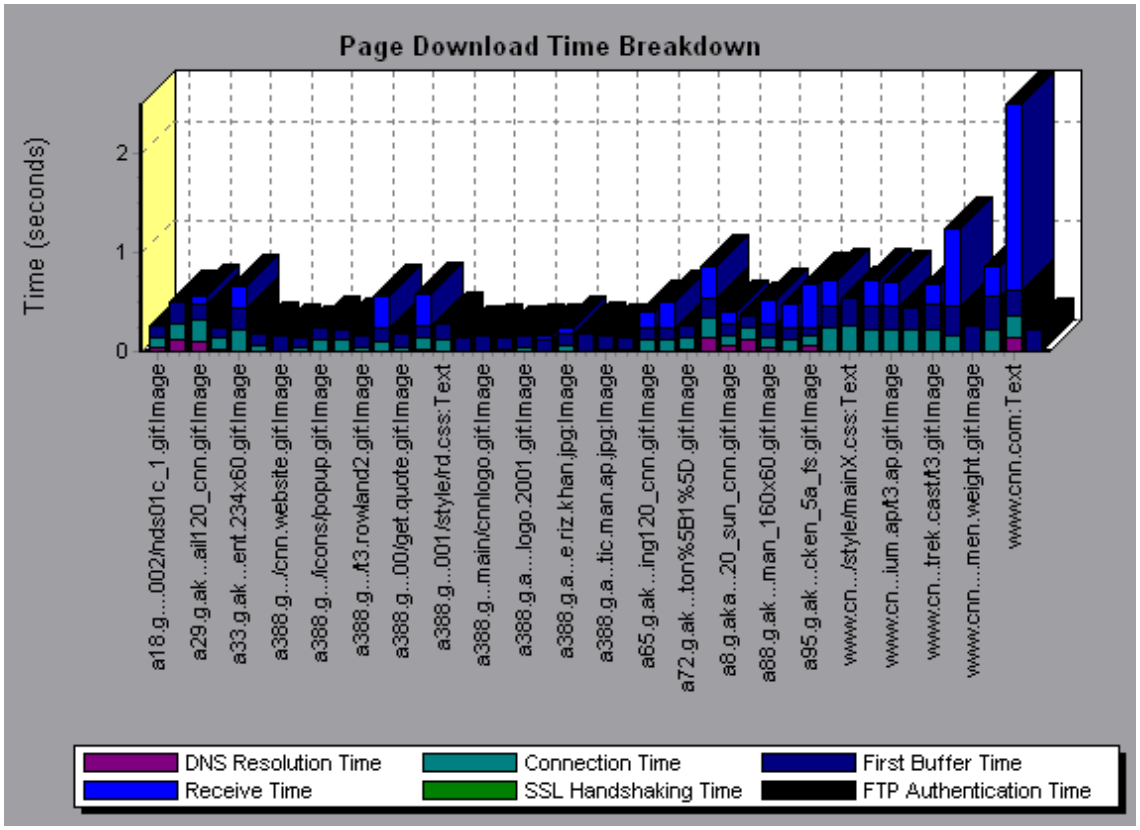
The Page Download Time Breakdown graph demonstrates that receive time, connection time, and first buffer time accounted for a large portion of the time taken to download the main cnn.com URL.




**Example**

If you break the cnn.com URL down further, you can isolate the components with the longest download time, and analyze the network or server problems that contributed to the delay in response time.

Breaking down the cnn.com URL demonstrates that for the component with the longest download time (the www.cnn.com component), the receive time accounted for a large portion of the download time.



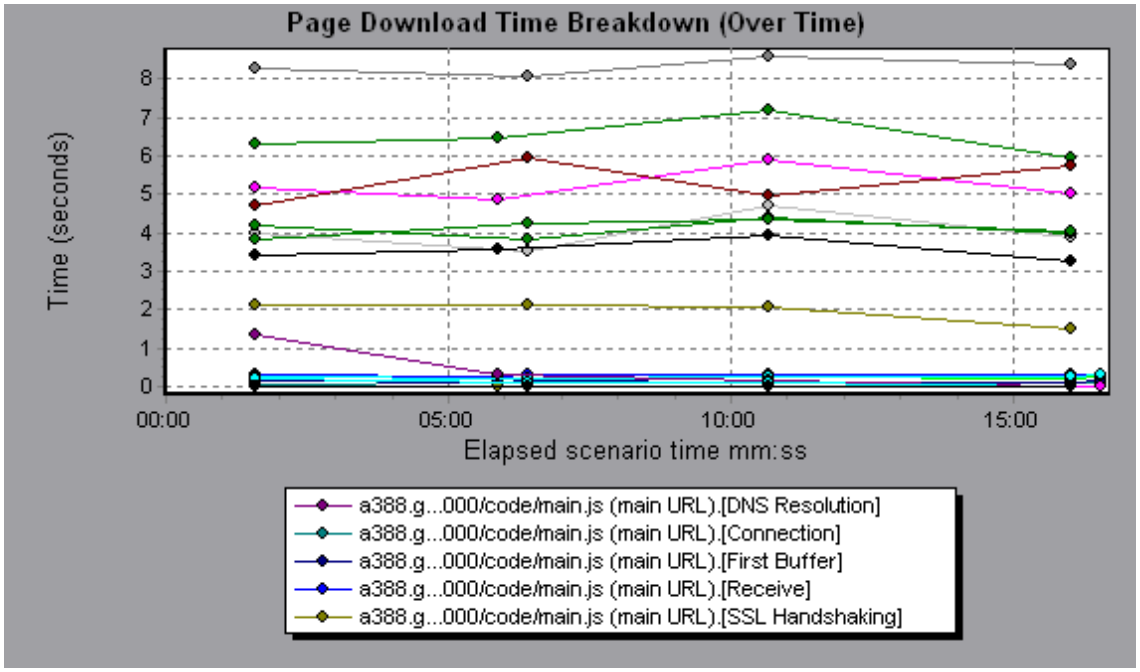

## Page Download Time Breakdown (Over Time) Graph

The graph displays a breakdown of each page component's download time during each second of the load test scenario run.

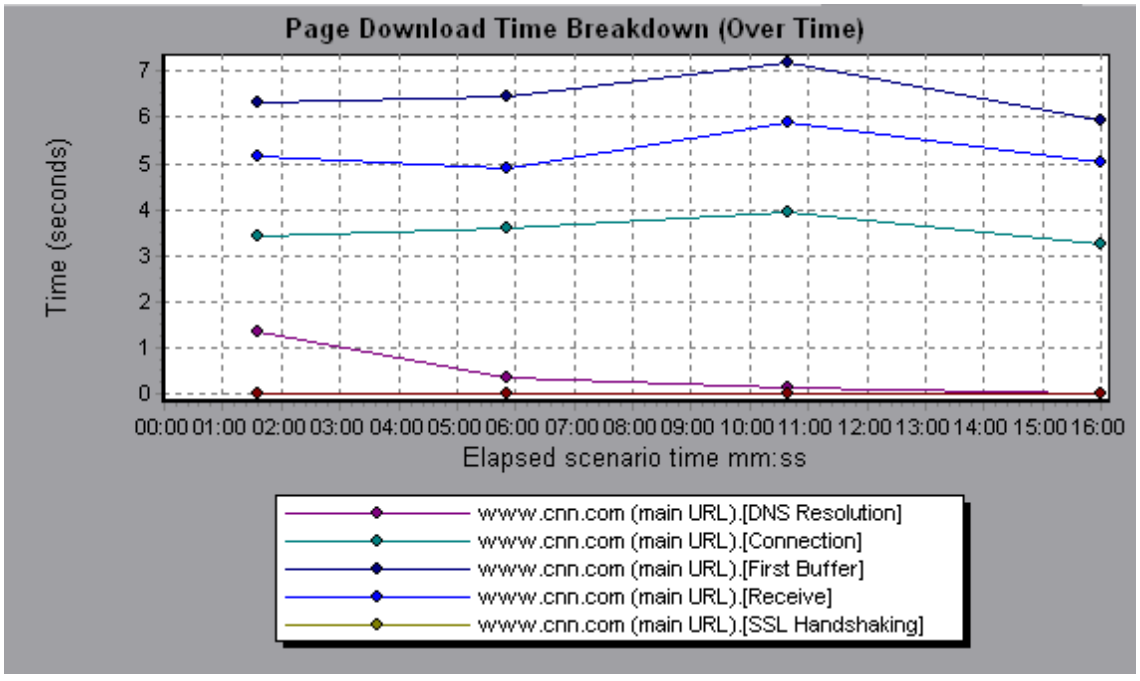
| | |
|---|---|
| **Purpose** | This graph enables you to determine at what point during scenario execution network or server problems occurred. |
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | Time (in seconds) taken for each step in the download process. |
| **Tips** | To isolate the most problematic components, you can sort the legend window according to the average number of seconds taken to download a component. To sort the legend by average, double-click the **Average** column heading. |
| **Notes** | • Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the Connection Time for www.cnn.com is the sum of the Connection Time for each of the page's components.<br>• When the Page Download Time Breakdown (Over Time) graph is selected from the Web Page Diagnostics graph, it appears as an area graph. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

**Example**

This graph enables you to determine at what point during scenario execution network or server problems occurred.




If you examine the cnn.com component, the Page Download Time Breakdown (Over Time) graph demonstrates that **First Buffer** and **Receive** time remained high throughout the scenario, and that **DNS Resolution** time decreased during the scenario.



# Page Download Time Breakdown Graph Breakdown Options

The Page Download Time Breakdown graph breaks down each component by DNS resolution time, connection time, time to first buffer, SSL handshaking time, receive time, FTP authentication time, client time, and error time.

These breakdowns are described below:

| Name | Description |
|---|---|
| DNS Resolution | Displays the amount of time needed to resolve the DNS name to an IP address, using the closest DNS server. The DNS Lookup measurement is a good indicator of problems in DNS resolution, or problems with the DNS server. |
| Connection | Displays the amount of time needed to establish an initial connection with the Web server hosting the specified URL. The connection measurement is a good indicator of problems along the network. It also indicates whether the server is responsive to requests. |
| First Buffer | Displays the amount of time that passes from the initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server. The first buffer measurement is a good indicator of Web server delay as well as network latency.<br>**Note:** Since the buffer size may be up to 8K, the first buffer might also be the time it takes to completely download the element. |
| SSL Handshaking | Displays the amount of time taken to establish an SSL connection (includes the client hello, server hello, client public key transfer, server certificate transfer, and other—partially optional—stages). After this point, all the communication between the client and server is encrypted.<br>The SSL Handshaking measurement is only applicable for HTTPS communications. |
| Receive | Displays the amount of time that passes until the last byte arrives from the server and the downloading is complete.<br>The Receive measurement is a good indicator of network quality (look at the time/size ratio to calculate receive rate). |
| FTP Authentication | Displays the time taken to authenticate the client. With FTP, a server must authenticate a client before it starts processing the client's commands.<br>The FTP Authentication measurement is only applicable for FTP protocol communications. |
| Client Time | Displays the average amount of time that passes while a request is delayed on the client machine due to browser think time or other client-related delays. |

| Name | Description |
|---|---|
| Error Time | Displays the average amount of time that passes from the moment an HTTP request is sent until the moment an error message (HTTP errors only) is returned. |

## Time to First Buffer Breakdown Graph

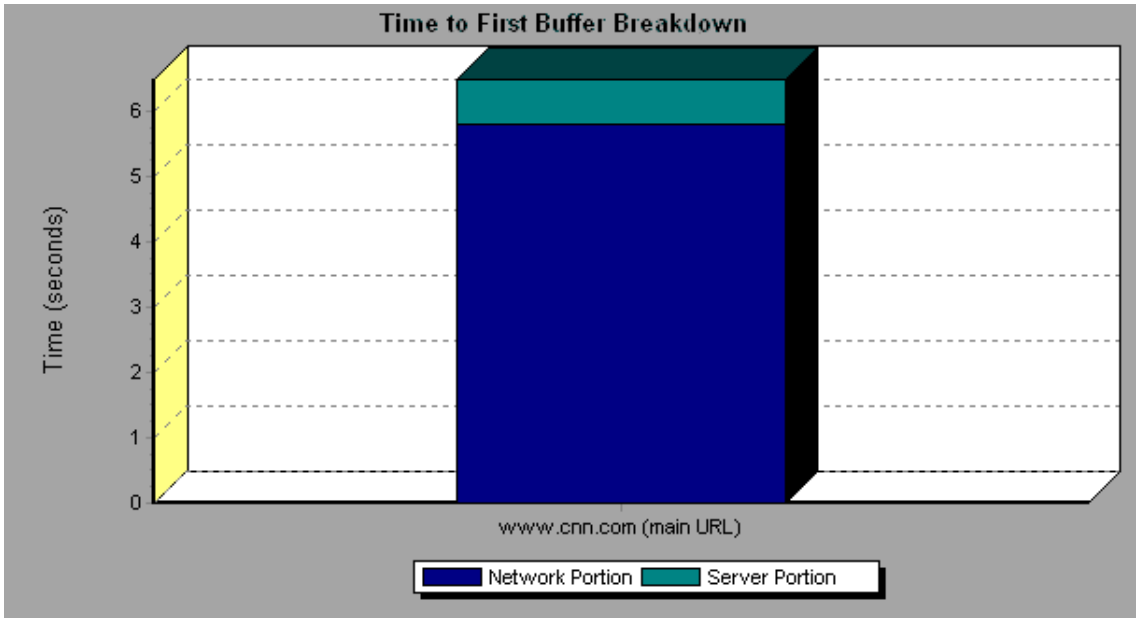
This graph displays each Web page component's relative server/network time (in seconds) for the period of time until the first buffer is successfully received back from the Web server.

> **Note:** This graph is only relevant when the load generator does not use a proxy to connect to the application under test. If the load generator is connected through a proxy, this graph will only show the proxy latency—not the application's latency.

| | |
|---|---|
| **Purpose** | If the download time for a component is high, you can use this graph to determine whether the problem is server- or network-related. |
| **X-axis** | Specifies the name of the component. |
| **Y-axis** | Shows the average network/server time (in seconds) for each component. |
| **Measurements** | • Network time is defined as the average amount of time that passes from the moment the first HTTP request is sent until receipt of ACK.<br>• Server time is defined as the average amount of time that passes from the receipt of ACK of the initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server. |
| **Note** | • Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the network time for www.cnn.com is the sum of the network time for each of the page's components.<br>• Because server time is being measured from the client, network time may influence this measurement if there is a change in network performance from the time the initial HTTP request is sent until the time the first buffer is sent. The server time displayed, therefore, is estimated server time and may be slightly inaccurate.<br>• The graph can only be viewed as a bar graph. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

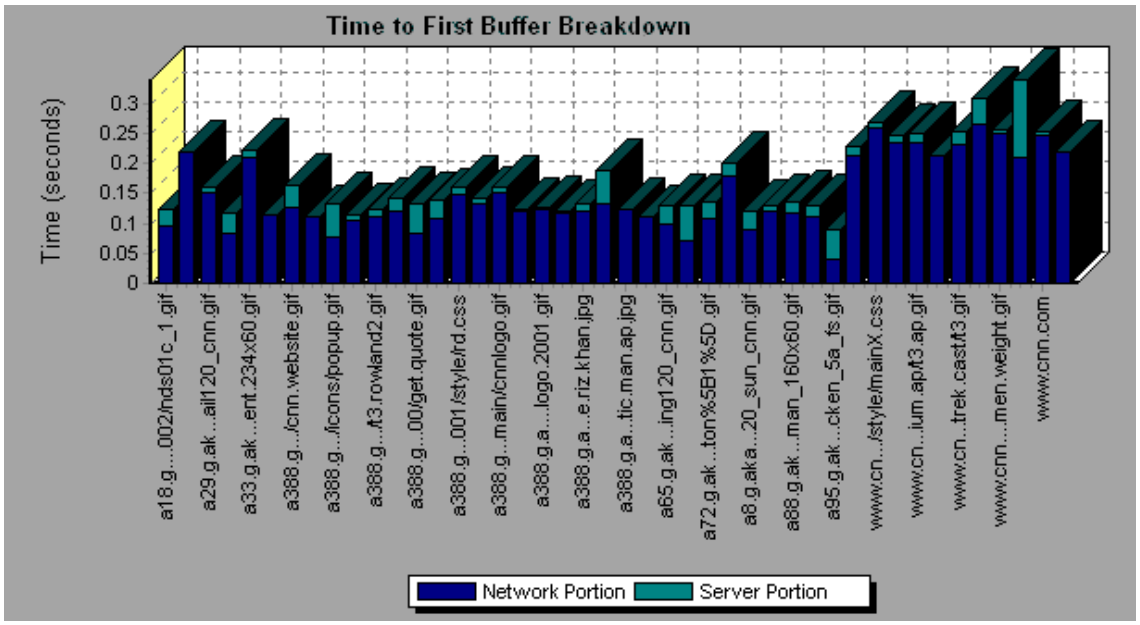
**Example**

In the following example it is apparent that network time is greater than server time.




**Example**

You can break the main cnn.com URL down further to view the time to first buffer breakdown for each of its components. It is apparent that for the main cnn.com component (the first component on the right), the time to first buffer breakdown is almost all network time.




## Time to First Buffer Breakdown (Over Time) Graph

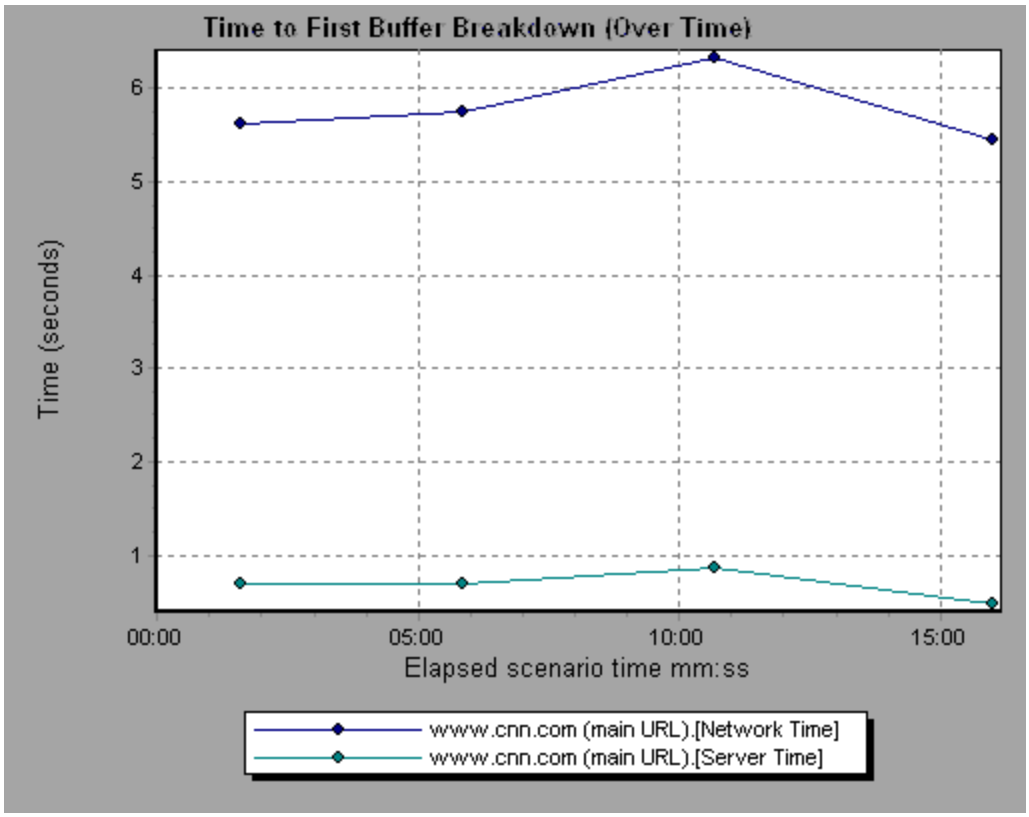
This graph displays each Web page component's server and network time (in seconds) during each second of the load test scenario run, for the period of time until the first buffer is successfully received

back from the Web server.

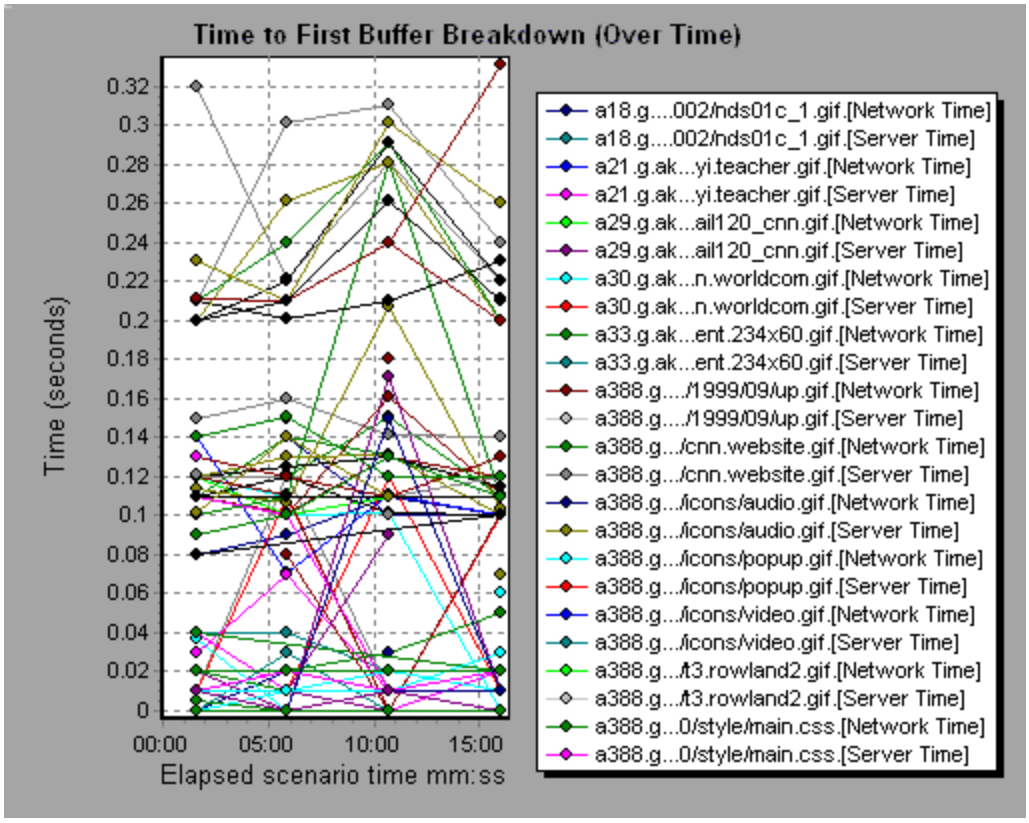
> **Note:** This graph is only relevant when the load generator does not use a proxy to connect to the application under test. If the load generator is connected through a proxy, this graph will only show the proxy latency—not the AUT latency.

| | |
|---|---|
| **Purpose** | You can use this graph to determine when during the scenario run a server- or network-related problem occurred. |
| **X-axis** | Elapsed time from the beginning of the scenario run. |
| **Y-axis** | Average network or server time (in seconds) for each component. |
| **Measurements** | • Network time is defined as the average amount of time that passes from the moment the first HTTP request is sent until receipt of ACK.<br>• Server time is defined as the average amount of time that passes from the receipt of ACK of the initial HTTP request (usually GET) until the first buffer is successfully received back from the Web server.<br>**Note:** Because server time is being measured from the client, network time may influence this measurement if there is a change in network performance from the time the initial HTTP request is sent until the time the first buffer is sent. The server time displayed, therefore, is estimated server time and may be slightly inaccurate. |
| **Note** | • Each measurement displayed on the page level is the sum of that measurement recorded for each page component. For example, the network time for **https://www.microfocus.com/** is the sum of the network time for each of the page's components.<br>• When the Time to First Buffer Breakdown (Over Time) graph is selected from the Web Page Diagnostics graph, it appears as an area graph. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

**Example**


Time to First Buffer Breakdown (Over Time)
Time (seconds)
6
5
4
3
2
1
00:00
05:00
10:00
15:00
Elapsed scenario time mm:ss
www.cnn.com (main URL).[Network Time]
www.cnn.com (main URL).[Server Time]

You can break down the main cnn.com URL to view the time to first buffer breakdown for each of its components.



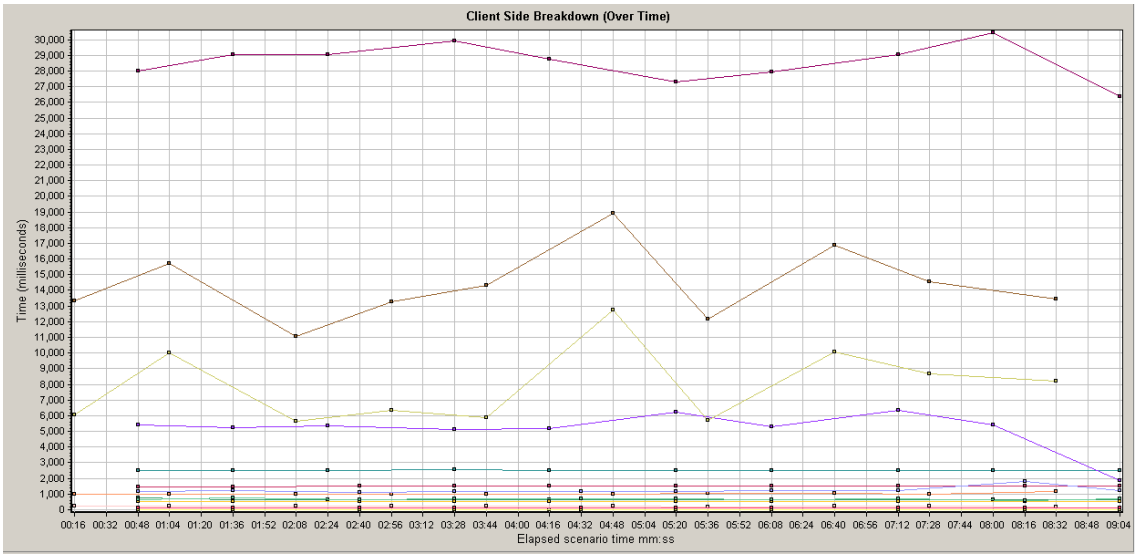

## Client Side Breakdown (Over Time) Graph

This graph displays the client side breakdown of each transaction during each second of the load test scenario run.

| | |
|---|---|
| **X-axis** | The elapsed time from the beginning of the scenario run. |
| **Y-axis** | The average response time (in seconds) for each transaction. |
| **Tips** | • To isolate the most problematic transactions, it may be helpful to sort the legend window according to the average number of seconds taken for the transaction to run. To sort the legend by average, double-click the **Average** column heading.<br>• To identify a transaction in the graph, you can select it. The corresponding line in the legend window is selected. |
| **See also** | "Web Page Diagnostics Graph" on page 1498 |

**Example**

Using the graph, you can track which transactions on the client side were most problematic, and at which point(s) during the scenario the problem(s) occurred.



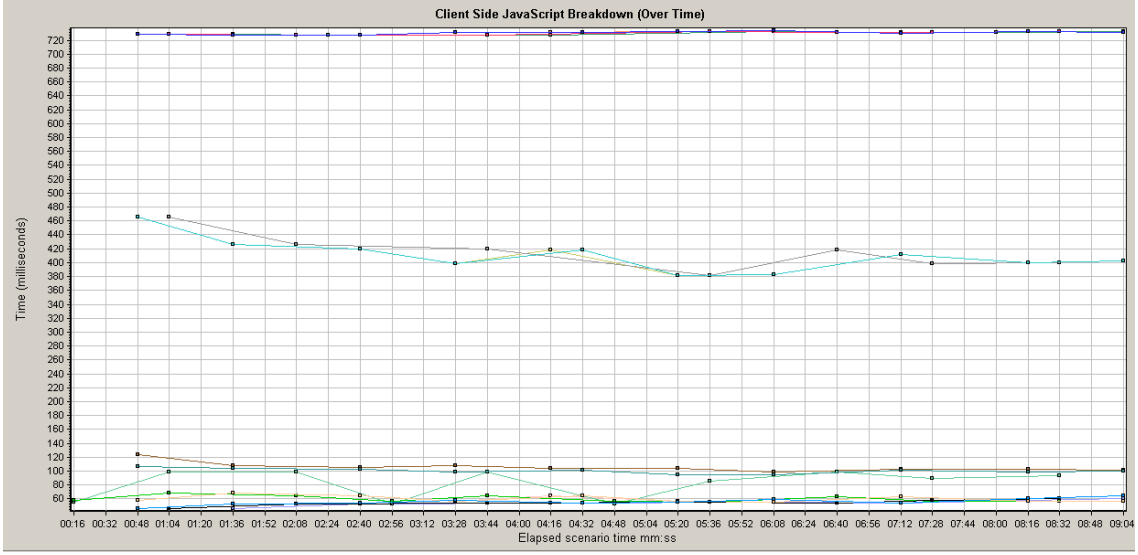

## Client Side Java Script Breakdown (Over Time) Graph

This graph displays the client side breakdown of each JavaScript transaction during each second of the load test scenario run.

| | |
|---|---|
| **X-axis** | The elapsed time from the beginning of the scenario run. |
| **Y-axis** | The average response time (in seconds) for each transaction. |
| **Tips** | • To isolate the most problematic transactions, it may be helpful to sort the legend window according to the average number of seconds taken for the transaction to run. To sort the legend by average, double-click the **Average** column heading.<br>• To identify a transaction in the graph, you can select it. The corresponding line in the legend window is selected. |
| **See also** | "Web Page Diagnostics Graph" on page 1498 |

**Example**

Using the graph, you can track which transactions on the client side were most problematic, and at which point(s) during the scenario the problem(s) occurred.



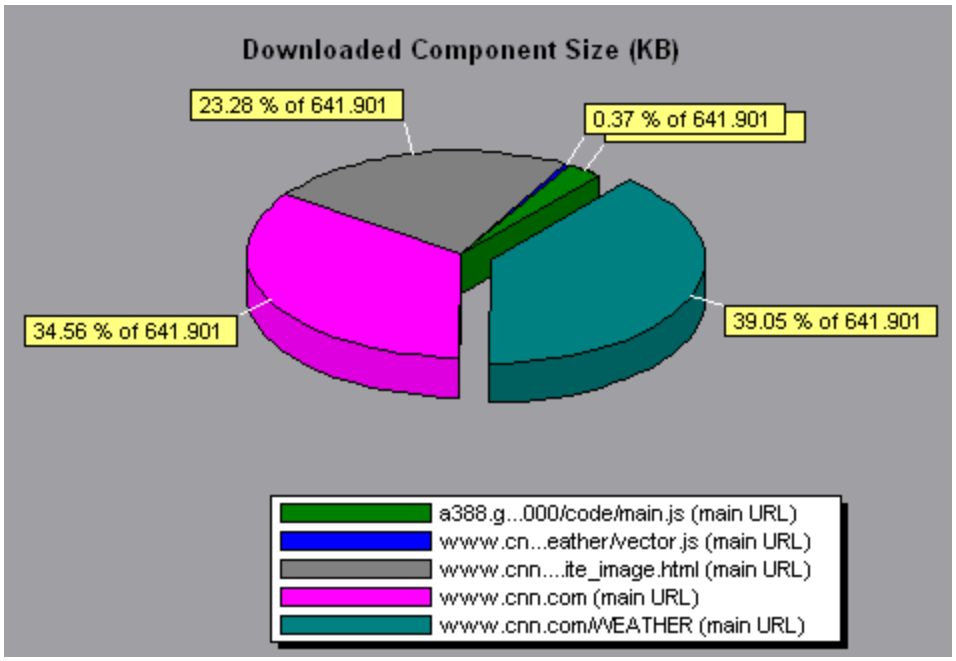

## Downloaded Component Size Graph

This graph displays the size of each Web page component.

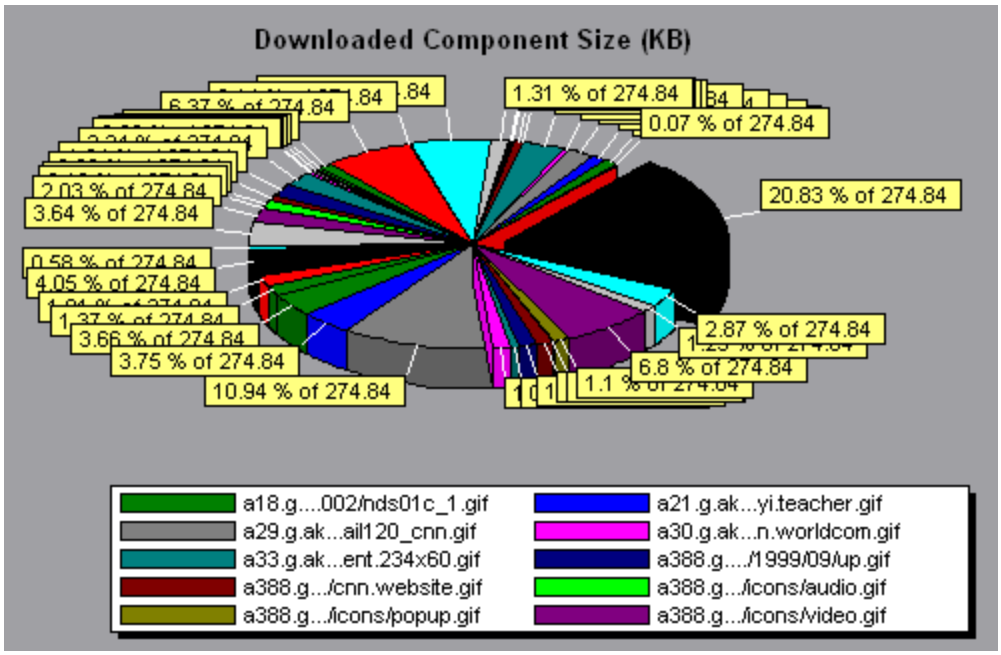
| | |
|---|---|
| **Note** | • The Web page size is a sum of the sizes of each of its components.<br>• The Downloaded Component Size graph can only be viewed as a pie graph. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

**Example**

In the following example the www.cnn.com/WEATHER component is 39.05% of the total size, whereas the main cnn.com component is 34.56% of the total size.



In the following example the cnn.com component's size (20.83% of the total size) may have contributed to the delay in its downloading. To reduce download time, it may help to reduce the size of this component.



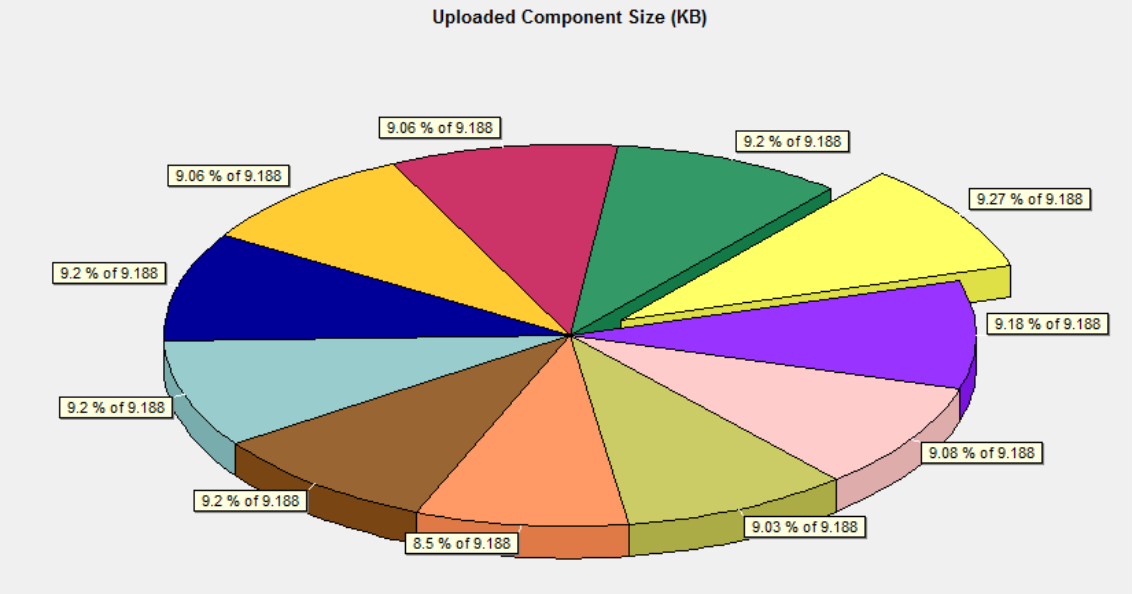

## Uploaded Component Size Graph

This graph displays the number of kilobytes that were sent for each Web page component that was uploaded.

| | |
|---|---|
| **Note** | • The Web page size is a sum of the sizes of each of its components.<br>• The Uploaded Component Size graph can be viewed only as a pie graph. |
| **See also** | "Web Page Diagnostics Graphs Overview" on page 1494 |

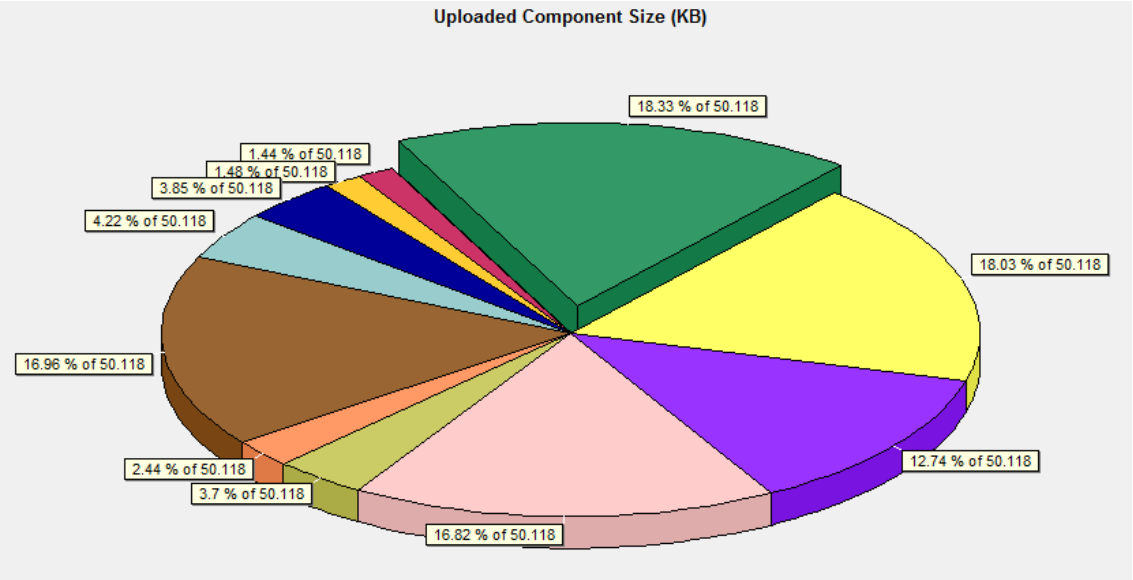
**Example 1**

In the following example, each of the components that was uploaded contained approximately 9% of the total size of all the of the uploaded components.




**Example 2**

In the following example, the size of the uploaded item that is shown in green is the biggest of all the components that were uploaded. To reduce upload time, it may help to reduce the size of this component.




# Web Resources Graphs

## Web Resources Graphs Overview

Web Resource graphs provide you with information about the performance of your Web server. You use the Web Resource graphs to analyze the following data:

- Throughput on the Web server
- The number of hits per second
- The number of HTTP responses per second
- The HTTP status codes returned from the Web server
- The number of downloaded pages per second
- The number of server retries per second
- A summary of the server retries during the load test scenario
- The number of open TCP/IP connections
- The number of TCP/IP connections per second
- The number of new and reused SSL connections opened per second

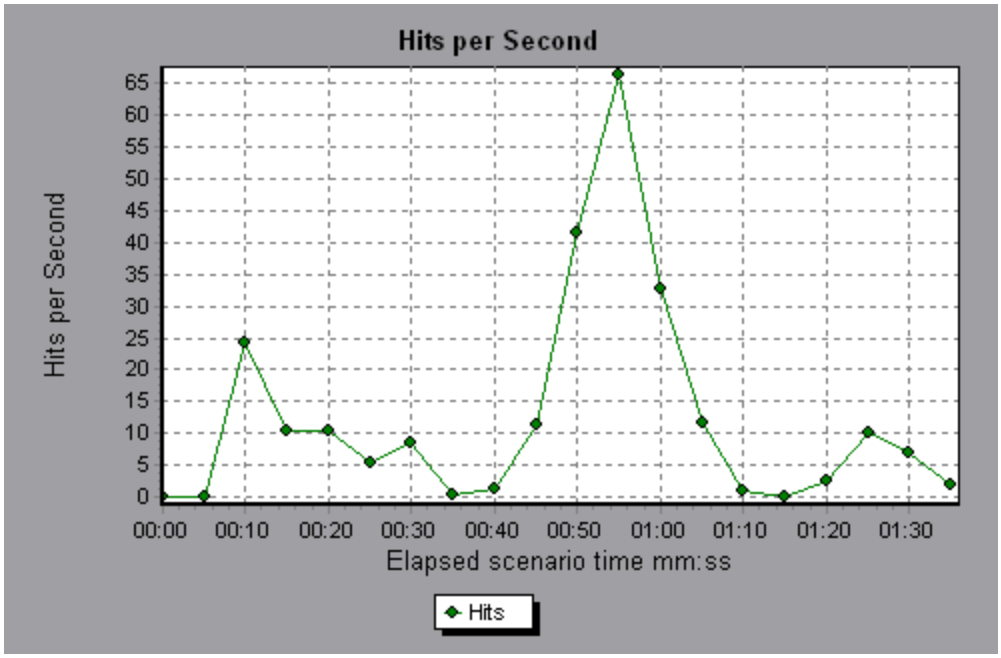
## Hits per Second Graph

This graph shows the number of HTTP requests made by Vusers to the Web server during each second of the load test scenario run.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of load Vusers generate, in terms of the number of hits. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of hits on the server. |
| **Tips** | Compare to the **Average Transaction Response Time** graph to see how the number of hits affects transaction performance. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **See also** | "Web Resources Graphs Overview" on the previous page |

### Example

In the following example, the most hits per second took place during the fifty-fifth second of the scenario.



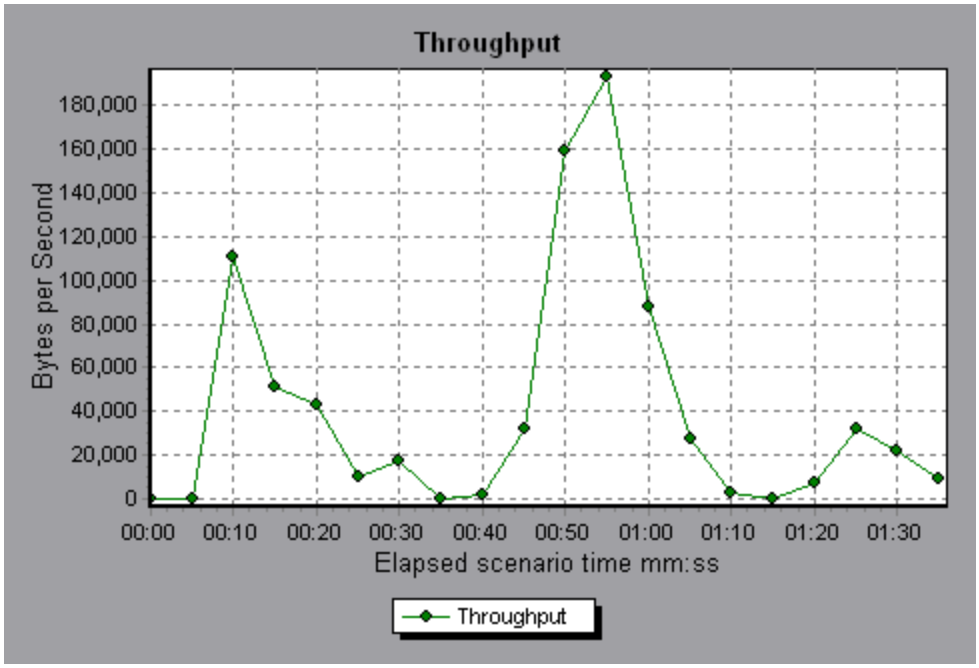

## Throughput Graph

This graph shows the amount of throughput on the server during each second of the load test scenario run. Throughput is measured in bytes or megabytes and represents the amount of data that the Vusers received from the server at any given second. To view throughput in megabytes, use the **Throughput (MB)** graph.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of load that Vusers generate, in terms of server throughput. |
| **X-axis** | Elapsed time since the start of the scenario run. |
| **Y-axis** | Throughput of the server, in bytes or megabytes. |
| **Tips** | Compare to the **Average Transaction Response Time** graph to see how the throughput affects transaction performance. |
| **Note** | You cannot change the granularity of the x-axis to a value that is less than the Web granularity you defined in the General tab of the Options dialog box. |
| **See also** | "Web Resources Graphs Overview" on page 1515 |

**Example**

In the following example, the highest throughput is 193,242 bytes during the fifty-fifth second of the scenario.



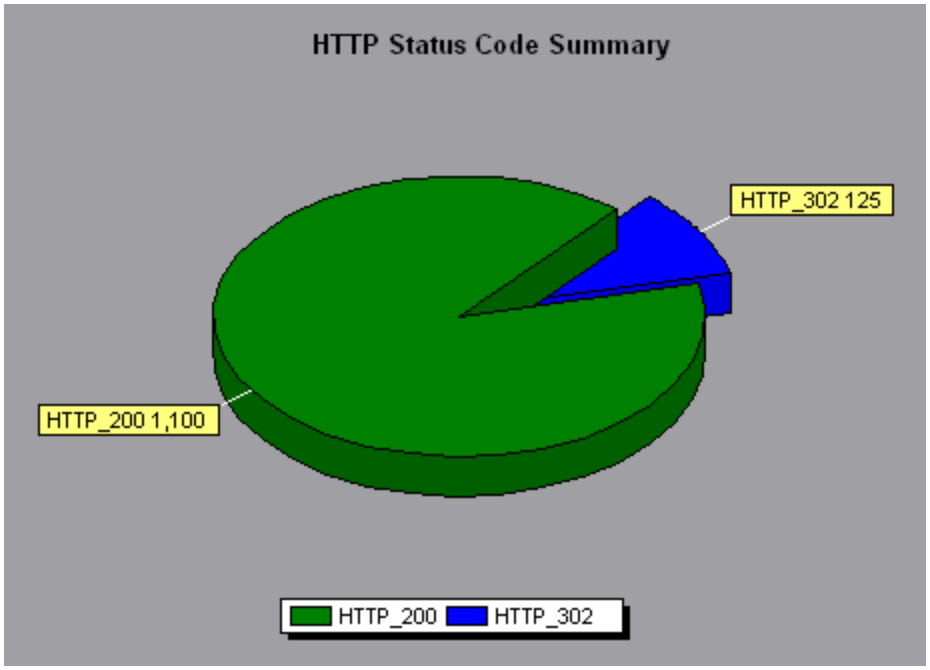

## HTTP Status Code Summary Graph

This graph shows the number of HTTP status codes returned from the Web server during the load test scenario run, grouped by status code. HTTP status codes indicate the status of HTTP requests, for example, "the request was successful","the page was not found".

| | |
|---|---|
| **Tips** | Use this graph together with the HTTP Responses per Second graph to locate scripts that generated error codes. |
| **Note** | This graph can only be viewed as a pie. |
| **See also** | • "Web Resources Graphs Overview" on page 1515<br>• "HTTP Status Codes" on the next page |

**Example**

In the following example, the graph shows that only the HTTP status codes 200 and 302 were generated. Status code 200 was generated 1,100 times, and status code 302 was generated 125 times.




## HTTP Status Codes

For information on HTTP status codes, see http://www.w3.org.

The following table displays a list of HTTP status codes often encountered in load tests:

| Code | Description |
|---|---|
| 200 | OK |
| 201 | Created |
| 202 | Accepted |
| 203 | Non-Authoritative Information |
| 204 | No Content |
| 205 | Reset Content |
| 206 | Partial Content |
| 300 | Multiple Choices |
| 301 | Moved Permanently |
| 302 | Found |
| 303 | See Other |

| Code | Description |
|---|---|
| 304 | Not Modified |
| 305 | Use Proxy |
| 307 | Temporary Redirect |
| 400 | Bad Request |
| 401 | Unauthorized |
| 402 | Payment Required |
| 403 | Forbidden |
| 404 | Not Found |
| 405 | Method Not Allowed |
| 406 | Not Acceptable |
| 407 | Proxy Authentication Required |
| 408 | Request Timeout |
| 409 | Conflict |
| 410 | Gone |
| 411 | Length Required |
| 412 | Precondition Failed |
| 413 | Request Entity Too Large |
| 414 | Request - URI Too Large |
| 415 | Unsupported Media Type |
| 416 | Requested range not satisfiable |
| 417 | Expectation Failed |
| 500 | Internal Server Error |
| 501 | Not Implemented |
| 502 | Bad Gateway |
| 503 | Service Unavailable |

| Code | Description |
|---|---|
| 504 | Gateway Timeout |
| 505 | HTTP Version not supported |

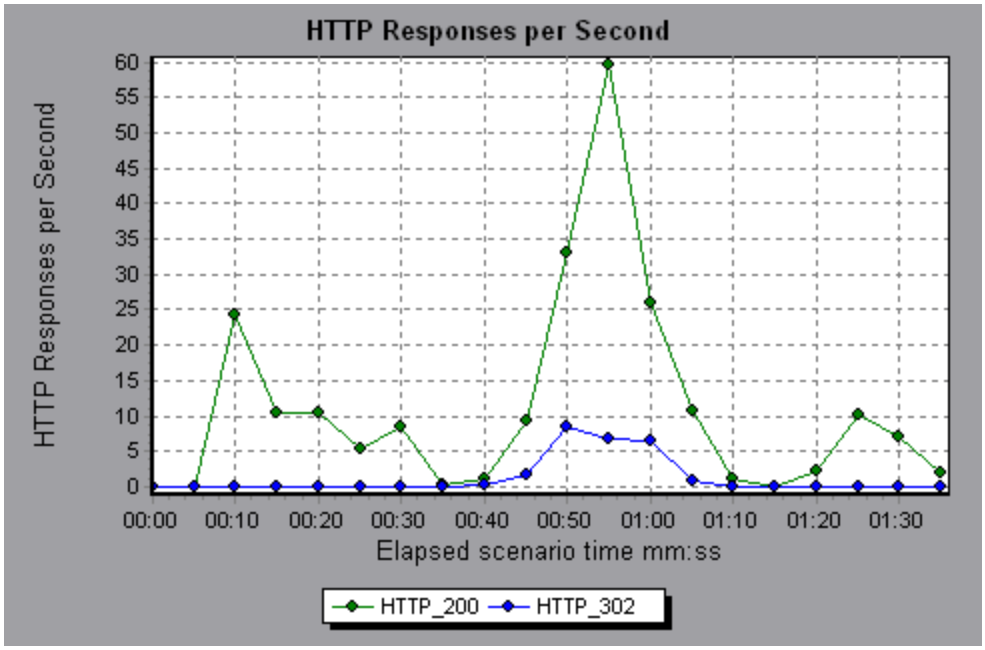
## HTTP Responses per Second Graph

This graph shows the number of HTTP status codes returned from the Web server during each second of the load test scenario run, grouped by status code. HTTP status codes indicate the status of HTTP requests, for example, "the request was successful", "the page was not found".

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of HTTP responses per second. |
| **Tip** | To locate scripts that generated error codes, you can group the results shown in this graph by script, using the "Group By" function. For more information on the "Group By" function, see "Filtering and Sorting Graph Data" on page 1342. |
| **See also** | • "Web Resources Graphs Overview" on page 1515<br>• "HTTP Status Codes" on page 1519 |

**Example**

In the following example, the greatest number of **200** status codes, 60, was generated in the fifty-fifth second of the scenario run. The greatest number of **302** codes, 8.5, was generated in the fiftieth second of the scenario run.



# Pages Downloaded per Second Graph

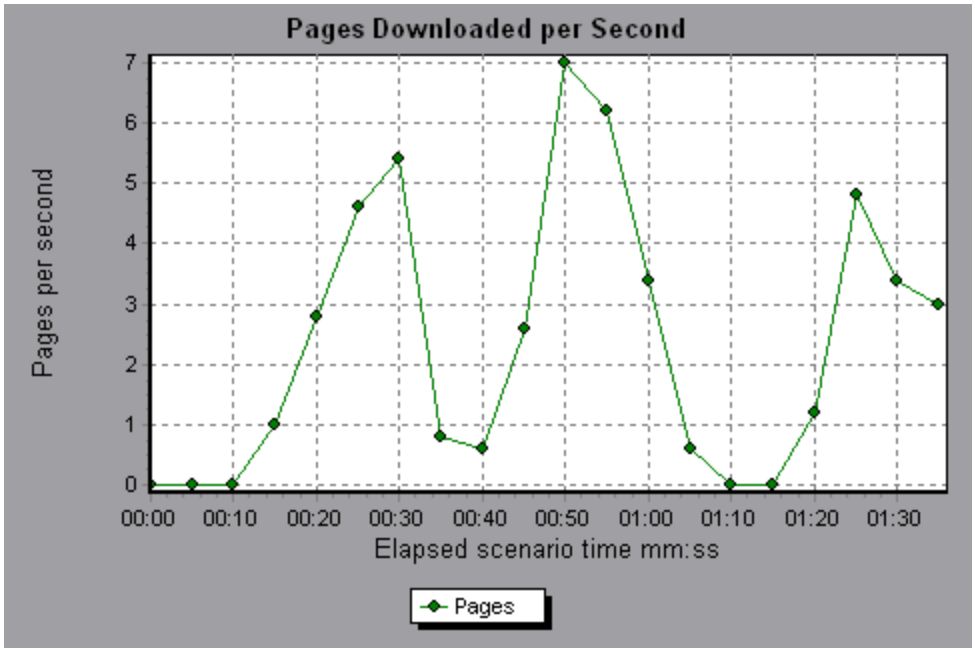
This graph shows the number of Web pages downloaded from the server during each second of the load test scenario run.

Like the Throughput graph, the Pages Downloaded per Second graph represents the amount of data that the Vusers received from the server at any given second. However, the Throughput graph takes into account each resource and its size (for example, the size of each **.gif** file, the size of each Web page). The Pages Downloaded per Second graph takes into account only the number of pages.

| | |
|---|---|
| **Purpose** | Helps you evaluate the amount of load Vusers generate, in terms of the number of pages downloaded. |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of Web pages downloaded from the server. |
| **Note** | To view the Pages Downloaded per Second graph, you must select **Pages per second (HTML Mode only)** from the runtime settings Preferences tab before running your scenario. |
| **See also** | "Web Resources Graphs Overview" on page 1515 |

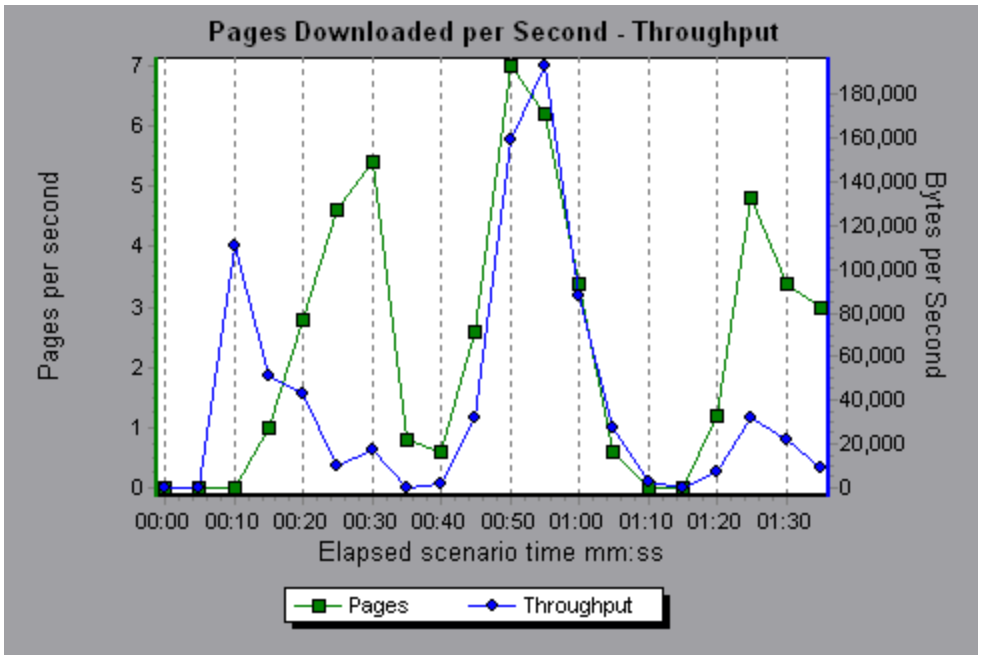
**Example 1**

In the following example, the greatest number of pages downloaded per second, about 7, occurred in the fiftieth second of the scenario run.




**Example 2**

In the following example, the Throughput graph is merged with the Pages Downloaded per Second graph. It is apparent from the graph that throughput is not completely proportional to the number of pages downloaded per second. For example, between 10 and 25 seconds into the scenario run, the number of pages downloaded per second increased while the throughput decreased.




## Retries per Second Graph

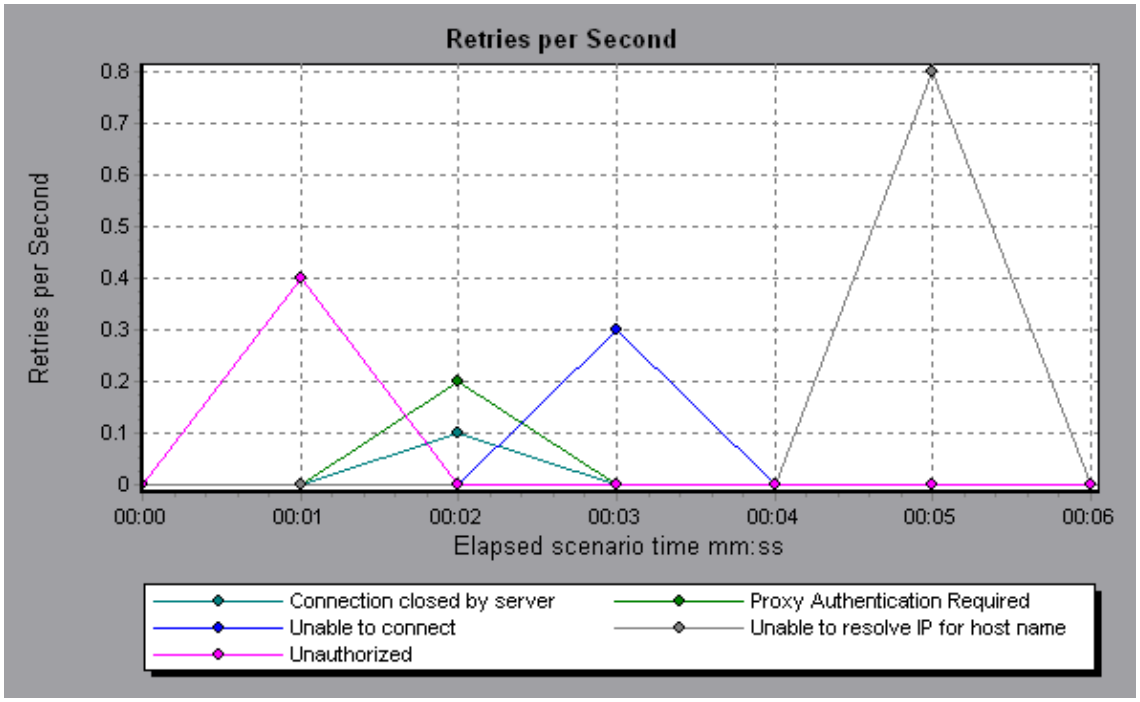
This graph displays the number of attempted server connections during each second of the load test scenario run. A server connection is retried when:

- the initial connection was unauthorized
- proxy authentication is required
- the initial connection was closed by the server
- the initial connection to the server could not be made
- when the server was initially unable to resolve the load generator's IP address

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of server retries per second. |
| **See also** | "Web Resources Graphs Overview" on page 1515 |

**Example**

In the following example, the graph shows that during the first second of the scenario, the number of retries was 0.4, whereas in the fifth second of the scenario, the number of retries per second rose to 0.8.



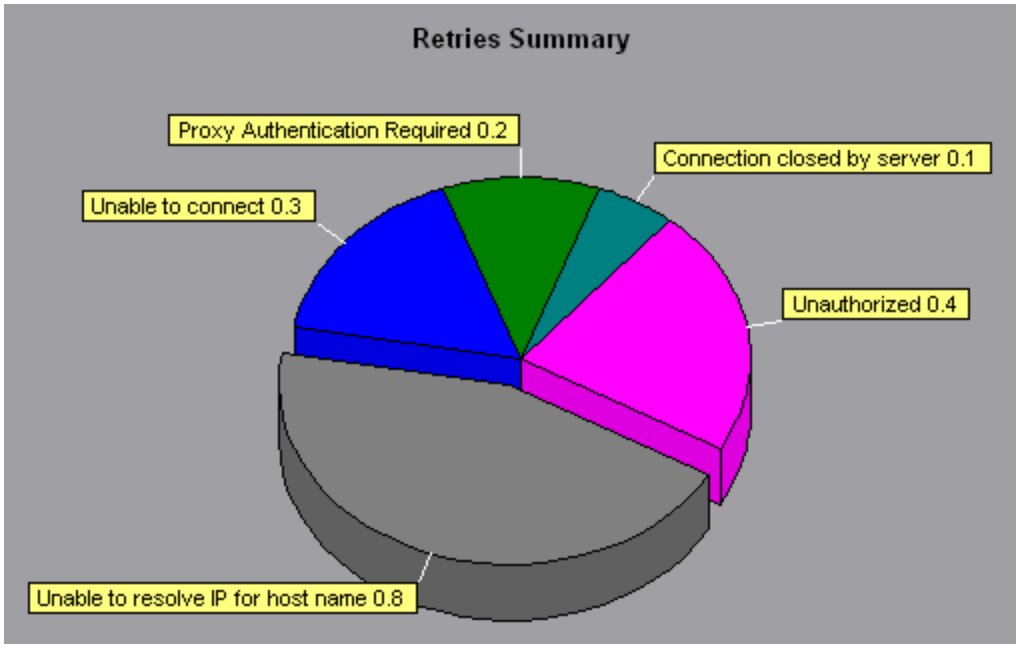

## Retries Summary Graph

This graph shows the number of attempted server connections during the load test scenario run, grouped by the cause of the retry.

| | |
|---|---|
| **Tips** | Use this graph together with the Retries per Second graph to determine when server retries were attempted. |
| **Note** | This graph can only be viewed as a pie. |
| **See also** | "Web Resources Graphs Overview" on page 1515 |

**Example**

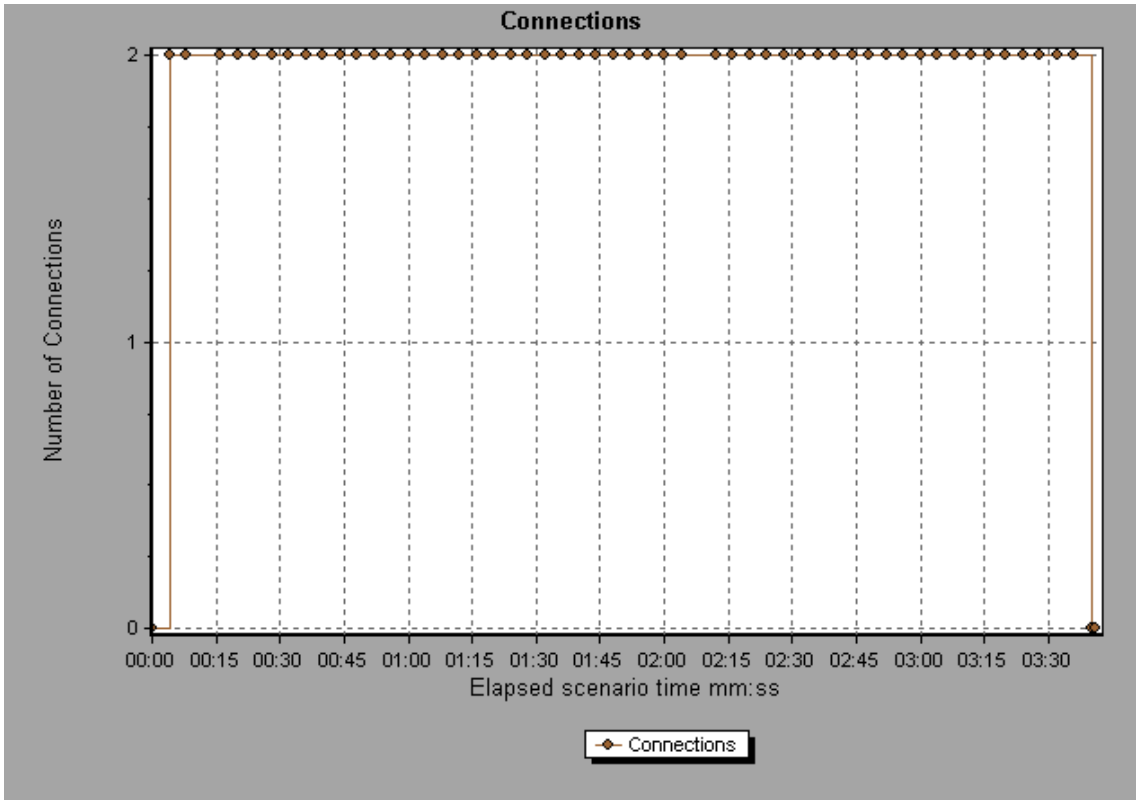
In the following example, the graph shows that the server's inability to resolve the load generator's IP address was the leading cause of server retries during the scenario run.




## Connections Graph

This graph shows the number of open TCP/IP connections (y-axis) at each point in time of the load test scenario (x-axis). Depending on the emulated browser type, each Vuser may open several simultaneous connections per Web server.

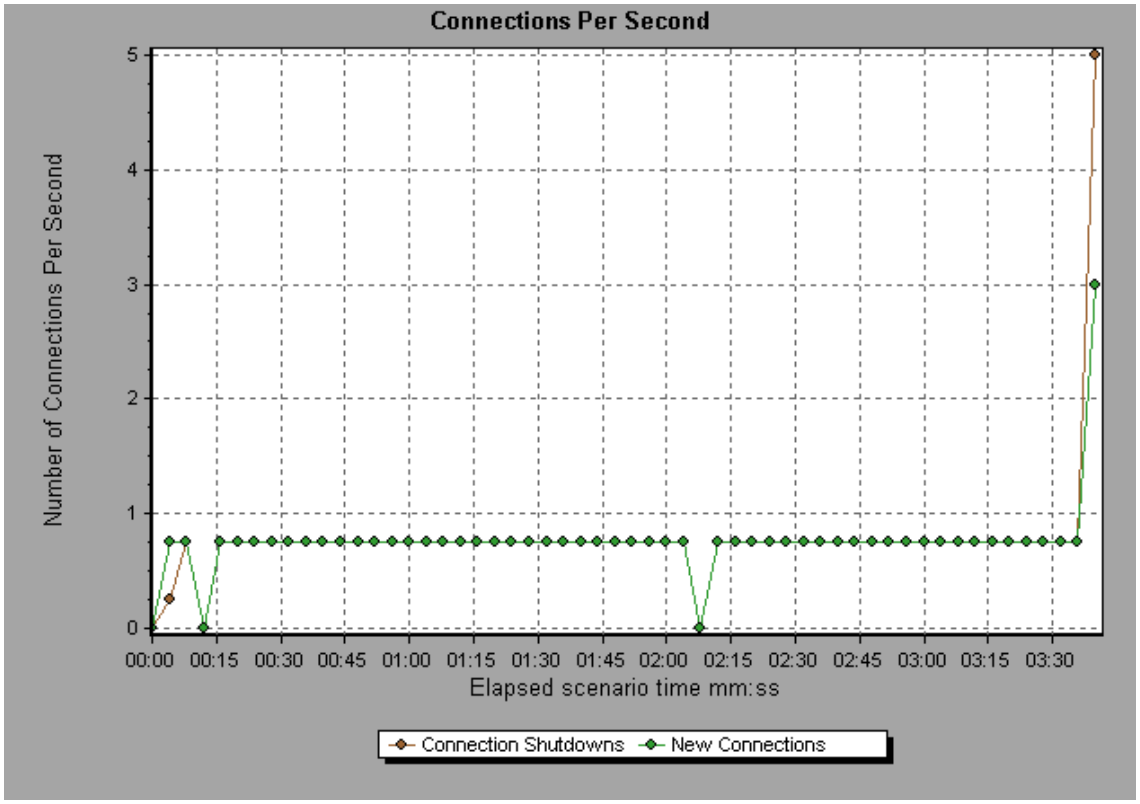
| | |
|---|---|
| **Purpose** | This graph is useful in indicating when additional connections are needed. For example, if the number of connections reaches a plateau, and the transaction response time increases sharply, adding connections would probably cause a dramatic improvement in performance (reduction in the transaction response time). |
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Open TCP/IP connections. |
| **See also** | "Web Resources Graphs Overview" on page 1515 |




## Connections per Second Graph

This graph shows the number of new TCP/IP connections (y-axis) opened and the number of connections that are shut down for each second of the load test scenario (x-axis).

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | TCP/IP connections per second. |
| **Tip** | The number of new connections should be a small fraction of the number of hits per second, because new TCP/IP connections use a lot of server, router, and network resources. Ideally, many HTTP requests should use the same connection, instead of opening a new connection for each request. |
| **See also** | "Web Resources Graphs Overview" on page 1515 |

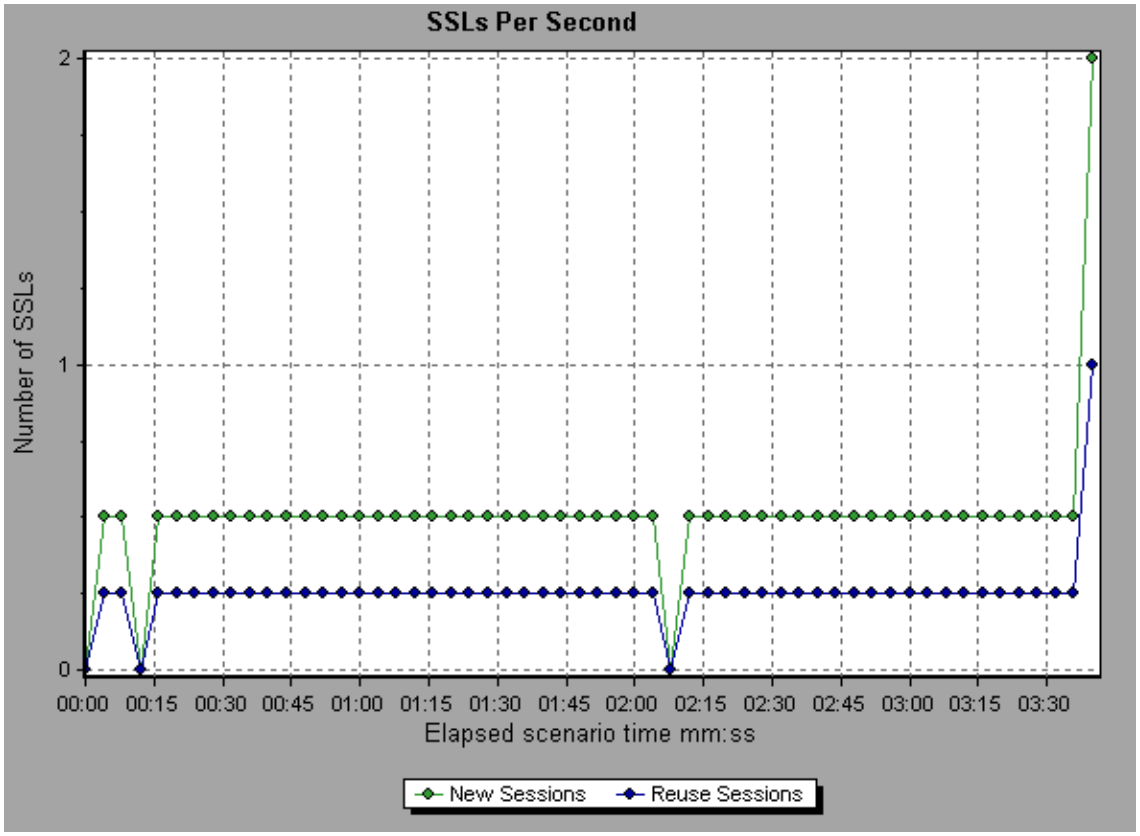



## SSLs per Second Graph

This graph shows the number of new and reused SSL Connections (y-axis) opened in each second of the load test scenario (x-axis). An SSL connection is opened by the browser after a TCP/IP connection has been opened to a secure server.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | Number of SSL Connections. |
| **Tip** | Reduce SSL connections.<br><br>Because creating a new SSL connection entails heavy resource consumption, you should reuse open SSL connections. There should be no more than one new SSL connection per Vuser.<br><br>If you reset TCP connections between iterations (**VuGen Runtime Settings > Browser Emulation node > Simulate a new user on each iteration**), you should have no more than one new SSL connection per iteration. |
| **See also** | "Web Resources Graphs Overview" on page 1515 |

**Example**




# Web Server Resource Graphs

## Web Server Resource Graphs Overview

Web Server Resource graphs provide you with information about the resource usage of the Apache and Microsoft IIS Web servers. In order to obtain data for these graphs, you need to activate the online monitor for the server and specify which resources you want to measure before running the load test scenario. For information on activating and configuring the Web Server Resource monitors, see "Web Server Resource Monitoring Overview" on page 1235.

In order to display all the measurements on a single graph, Analysis may scale them. The Legend window indicates the scale factor for each resource. To obtain the true value, multiply the scale factor by the displayed value.

## Apache Server Measurements

The following default measurements are available for the Apache server:

| Measurement | Description |
|---|---|
| **# Busy Servers** | The number of servers in the Busy state |

| Measurement | Description |
|---|---|
| **# Idle Servers** | The number of servers in the Idle state |
| **Apache CPU Usage** | The percentage of time the CPU is utilized by the Apache server |
| **Hits/sec** | The HTTP request rate |
| **KBytes Sent/sec** | The rate at which data bytes are sent from the Web server |

## IIS Server Measurements

The following default measurements are available for the IIS server:

| Object | Measurement | Description |
|---|---|---|
| **Web Service** | **Bytes Sent/sec** | The rate at which the data bytes are sent by the Web service. |
| **Web Service** | **Bytes Received/sec** | The rate at which the data bytes are received by the Web service. |
| **Web Service** | **Get Requests/sec** | The rate at which HTTP requests using the GET method are made. Get requests are generally used for basic file retrievals or image maps, though they can be used with forms. |
| **Web Service** | **Post Requests/sec** | The rate at which HTTP requests using the POST method are made. Post requests are generally used for forms or gateway requests. |
| **Web Service** | **Maximum Connections** | The maximum number of simultaneous connections established with the Web service. |
| **Web Service** | **Current Connections** | The current number of connections established with the Web service. |
| **Web Service** | **Current NonAnonymous Users** | The number of users that currently have a non-anonymous connection using the Web service. |
| **Web Service** | **Not Found Errors/sec** | The rate of errors due to requests that could not be satisfied by the server because the requested document could not be found. These are generally reported to the client as an HTTP 404 error code. |
| **Process** | **Private Bytes** | The current number of bytes that the process has allocated that cannot be shared with other processes. |

## Apache Server Graph

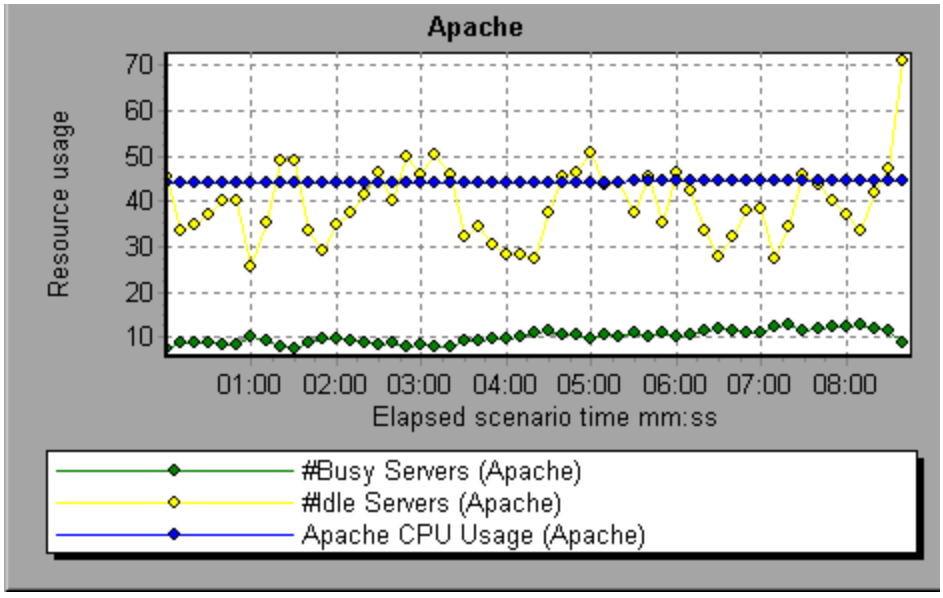
This graph shows server statistics as a function of the elapsed load test scenario time.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource usage on the Apache server during the scenario run. |
| **Note** | To obtain data for this graph, you need to enable the Apache online monitor (from Controller) and select the default measurements you want to display, before running the scenario. |
| **See also** | "Web Server Resource Graphs Overview" on page 1528<br>"Apache Server Measurements" on page 1528 |

**Example**

In the following example, the CPU usage remained steady throughout the scenario. At the end of the scenario, the number of idle servers increased. The number of busy servers remained steady at 1 throughout the scenario, implying that the Vuser only accessed one Apache server.

The scale factor for the **Busy Servers** measurement is 1/10 and the scale factor for **CPU usage** is 10.




## Microsoft Information Internet Server (IIS) Graph

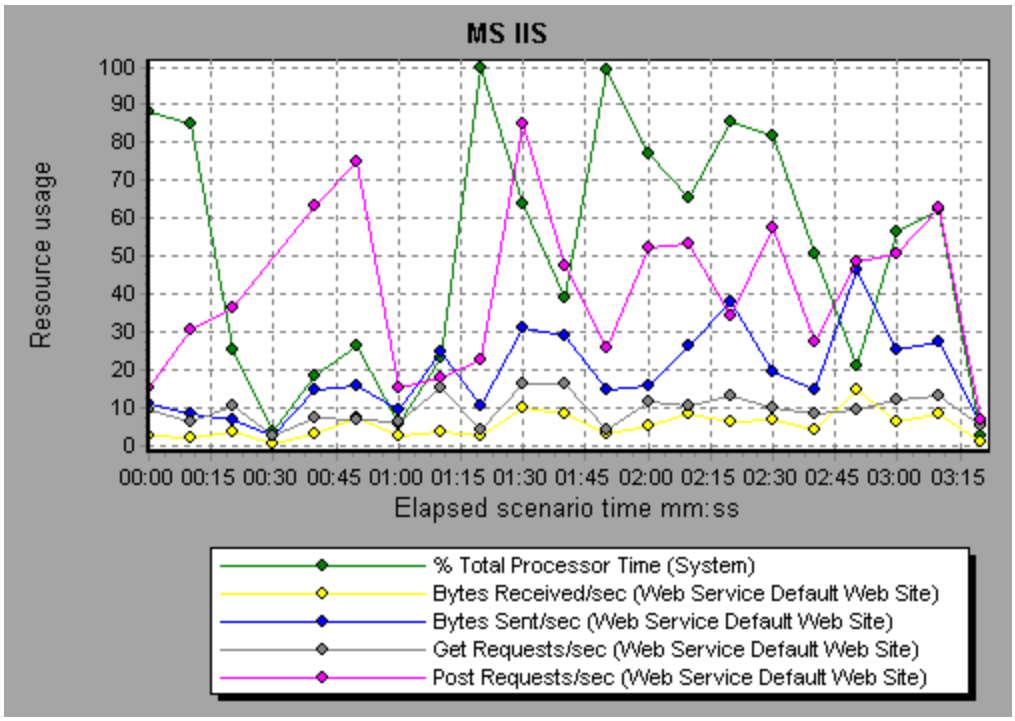
This graph shows server statistics as a function of the elapsed load test scenario time.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource usage on the MS IIS. |
| **Note** | To obtain data for this graph, you need to enable the MS IIS online monitor (from Controller) and select the default measurements you want to display, before running the scenario. |
| **See also** | "Web Server Resource Graphs Overview" on page 1528<br>"IIS Server Measurements" on page 1529 |

**Example**

In the following example the **Bytes Received/sec** and **Get Requests/sec** measurements remained fairly steady throughout the scenario, while the **% Total Processor Time**, **Bytes Sent/sec**, and **Post Requests/sec** measurements fluctuated considerably.

The scale factor for the **Bytes Sent/sec** and **Bytes Received/sec** measurements is 1/100, and the scale factor for the **Post Requests/sec** measurement is 10.




# Web Application Server Resource Graphs

Web Application Server Resource graphs provide you with resource usage information about the Microsoft ASP servers.

In order to obtain data for these graphs, you need to activate the online monitor for the application server and specify which resources you want to measure before running the load test scenario. For information on activating and configuring the Web Application Server Resource monitors, see "Web Application Server Monitoring" on page 1228. For more information, see the Monitors section the LoadRunner Controller documentation.

When you open a Web Application Server Resource graph, you can filter it to show only the relevant application. When you need to analyze other applications, you can change the filter conditions and display the desired resources.

In order to display all the measurements on a single graph, Analysis may scale them. The Legend window indicates the scale factor for each resource. To obtain the true value, multiply the scale factor by the displayed value. For more information on scaled measurements, see the example in "Web Server Resource Graphs Overview" on page 1528.

## Microsoft Active Server Pages (ASP) Graph

This graph displays statistics about the resource usage on the ASP server during the load test scenario run.

| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | The resource usage on the ASP server. |
| **Note** | To obtain data for this graph, you need to enable the Microsoft ASP online monitor (from Controller) and select the default measurements you want to display, before running the scenario. |

### Microsoft Active Server Pages (ASP) Measurements

The following default measurements are available for Microsoft Active Server Pages:

| Measurement | Description |
|---|---|
| **Errors per Second** | The number of errors per second. |
| **Requests Wait Time** | The number of milliseconds the most recent request was waiting in the queue. |
| **Requests Executing** | The number of requests currently executing. |
| **Requests Queued** | The number of requests waiting in the queue for service. |
| **Requests Rejected** | The total number of requests not executed because there were insufficient resources to process them. |
| **Requests Not Found** | The number of requests for files that were not found. |

| Measurement | Description |
|---|---|
| **Requests/sec** | The number of requests executed per second. |
| **Memory Allocated** | The total amount of memory (in bytes) currently allocated by Active Server Pages. |
| **Errors During Script Runtime** | The number of failed requests due to runtime errors. |
| **Sessions Current** | The current number of sessions being serviced. |
| **Transactions/sec** | The number of transactions started per second. |

## WebSocket Statistics Graphs

The **WebSocket Statistics** graphs provides you with statistics for the WebSocket data during the scenario run, such as byte rate, connection status, and the number of messages.

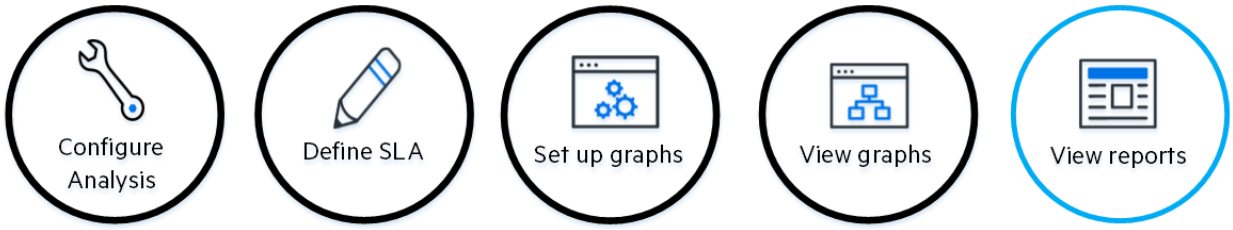
| | |
|---|---|
| **X-axis** | Elapsed time since the start of the run. |
| **Y-axis** | WebSocket per second throughout the whole scenario. |

The WebSocket Statistics graphs are:

- **WebSocket Bytes per second.** This graph shows the number of bytes that were sent and received per second.
- **WebSocket Connections per second.** This graph shows the number of new, failed, and closed connections. I
- **WebSocket Messages per second.** This graph shows the number of WebSocket messages that were sent, per second.

To gather these statistics, enable the WebSocket Statistics monitors before running your scenario. For details, see "WebSocket Statistics Monitor" on page 1233.

# Analysis Reports




## Understanding Analysis Reports

## Analysis Reports Overview

After running a load test scenario, you can view reports that summarize your system's performance. Analysis provides the following reporting tools:

- "Analysis Summary Report" on page 1547
- "SLA Reports" on page 1552
- "Transaction Analysis Report" on page 1553
- "HTML Reports" on page 1551

The Summary report provides general information about the scenario run. You can access the Summary report at any time from the Session Explorer.

The SLA report provides an overview of the defined SLAs (Service Level Agreements) with succeeded or failed status.

The Transaction Analysis report provides a detailed analysis of a specific transaction over a specific time period.

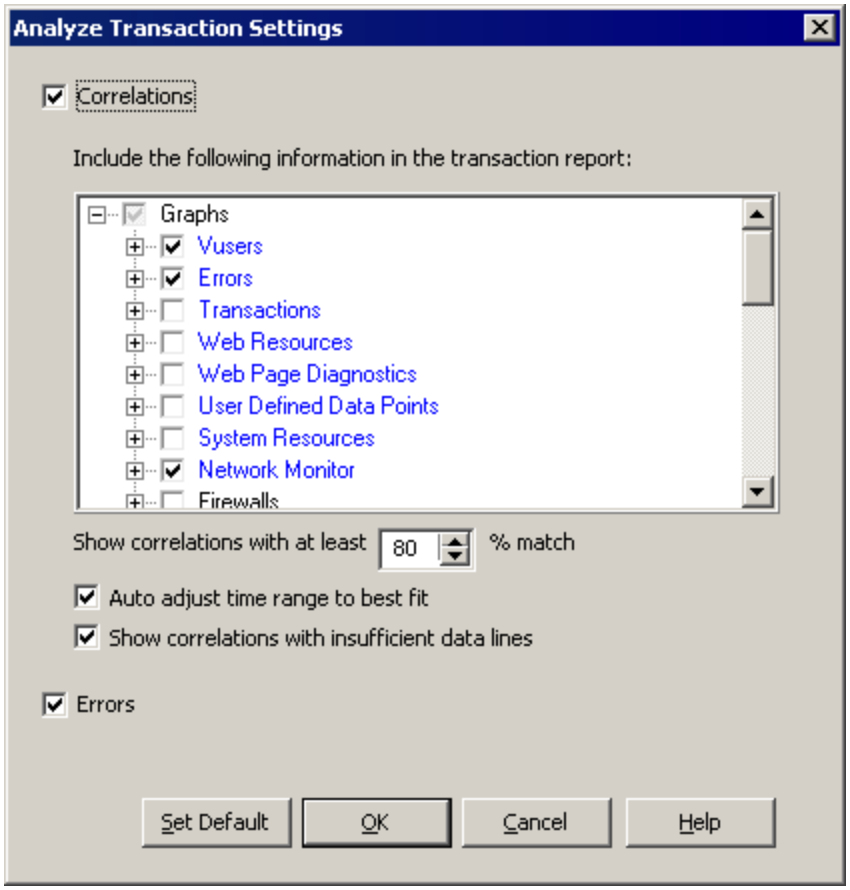
You can instruct Analysis to create an HTML report. The HTML report contains a page for each open graph, the Summary report, the SLA report, and the Transaction Analysis report.

Transaction reports provide performance information about the transactions defined within the Vuser scripts. These reports give you a statistical breakdown of your results and allow you to print and export the data.

> **Note:** SLA reports and Transaction Analysis reports are not available when generating Cross Result graphs. For more information on Cross Result graphs, see "Cross Result and Merged Graphs" on page 1357.

## Analyze Transaction Settings Dialog Box

This dialog box enables you to configure the Transaction Analysis Report to show correlations between the graph of the analyzed transaction and other graphs that you select.




| | |
|---|---|
| **To access** | Use one of the following:<br>• **Reports > Analyze Transaction > Settings**<br>• **Tools > Options > Analyze Transaction Settings tab** |
| **See also** | "Analyze Transactions Dialog Box" on the next page |

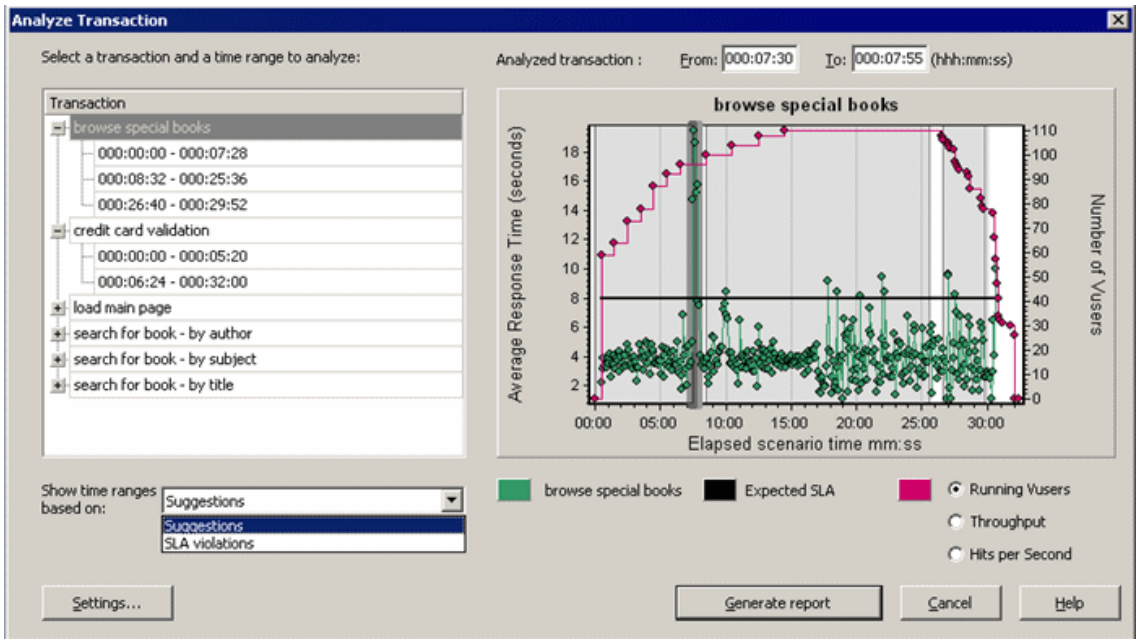
User interface elements are described below:

| UI Element | Description |
|---|---|
| **Correlations** | Defines which graphs you want Analysis to match to the graph of the transaction you selected. Graphs where data is available appear in blue. |
| **Show correlations with at least x% match** | The positive or negative percentage correlation between the graph of the analyzed transaction and the graphs selected above. You can change the percentage by entering a value in the box. The default is 20%. |

| UI Element | Description |
|---|---|
| **Auto adjust time range to best fit** | Analysis adjusts the selected time range to focus on the SLA violations within and around that time period. This option only applies when the Transaction Analysis report is generated directly from the Summary report (from the **X Worst transactions** or **Scenario behavior over time** sections). |
| **Show correlations with insufficient data lines** | Displays correlations where one of the measurements contains less than 15 units of granularity. |
| **Errors** | Displays errors in the Transaction Analysis Report if selected. |

## Analyze Transactions Dialog Box

You use the Analyze Transaction dialog box to define the criteria that will be used to analyze the selected transaction in the Transaction Analysis Report. You can analyze a transaction even if you have not defined an SLA.



| | |
|---|---|
| **To access** | **Reports > Analyze Transaction**<br>**Summary Report > right-click menu > Add New Item > Analyze Transaction**<br>**Toolbar >** <br>**Summary Report with no SLA > Statistics Summary section > Analyze Transaction tool link** |
| **Note** | Analysis data (for example, transactions) that has been excluded by the Summary Filter will not be available for analysis in the Transaction Analysis report. |
| **See also** | "Filtering and Sorting Graph Data" on page 1342 |

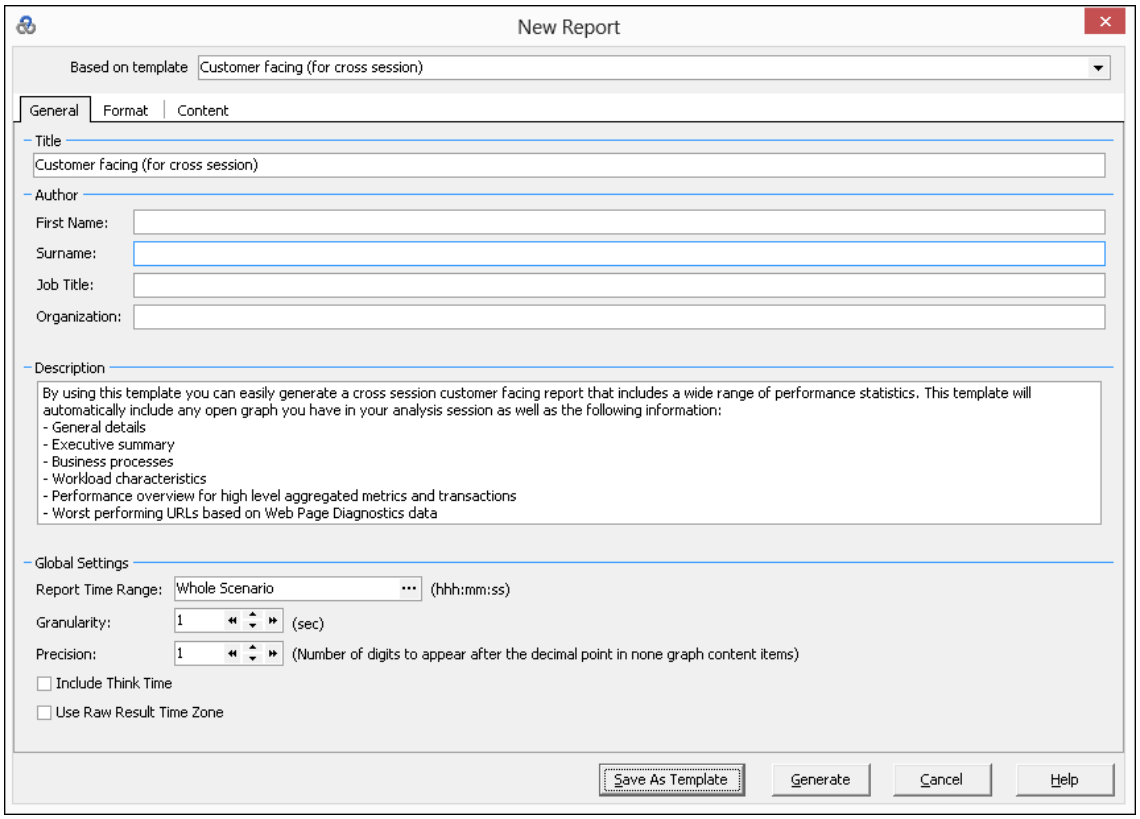
User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Show time ranges based on box** | Select one of the display options:<br>• **Suggestions.** Lists all transactions and time ranges from the scenario run.<br>• **SLA Violations.** Lists only those transactions and time ranges where the transaction exceeded the SLA. This option does not appear if there were no transactions that exceeded the SLA. |
| **Transaction** | Select the transaction to analyze from the Transaction tree. |
| **<Time Range>** | Select the time range to analyze in one of the following ways:<br>• Select the time range from the Transaction tree.<br>• Enter the time range in the **From** and **To** boxes above the graph.<br>• Select the time range by dragging the bars on the graph. |
| **<Display options>** | Select one of the following:<br>• **Running Vusers**<br>• **Throughput**<br>• **Hits per Second**<br>The option you select is displayed on the graph and will appear on the snapshot of the graph that appears on the Transaction Analysis Report. Note that your choice only affects the display of the graph and not the calculation for correlations. |

| UI Element | Description |
|---|---|
| **Settings** | Click **Settings** to define the Analyze Transaction settings in the **Analyze Transaction Settings** dialog box. For more information, see "Analyze Transaction Settings Dialog Box" on page 1534.<br><br>**Note**: You can also define the Analyze Transaction settings in the **Analyze Transaction Settings** tab of the **Options** dialog box (**Tools > Options**). |
| **Generate report** | The Transaction Analysis Report opens. Once the report has been created, you can access it at any time from the Session Explorer. |

## New Report Dialog Box

This dialog box enables you to create a report based on the report template selected. You can adjust the report template settings to generate a report that corresponds to the required report layout.




| | |
|---|---|
| **To access** | **Reports > New Report** |
| **See also** | "Report Templates Dialog Box" on the next page<br><br>**Note:** This dialog box and the Report Templates dialog box utilize the same components. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Based on Template** | The template upon which to build the report. After you select a template, the corresponding settings of the report template appear. |
| **General tab** | For user interface details, see "Report Templates - General Tab" on page 1541. |
| **Format tab** | For user interface details, see "Report Templates - Format Tab" on page 1542. |
| **Content tab** | For user interface details, see "Report Templates - Content Tab" on page 1544. |
| **Save As Template** | Prompts you for a template name that will be added to the report template list. |
| **Generate** | Generates the report according to your settings. |

# Analysis Report Templates

## Report Templates Overview

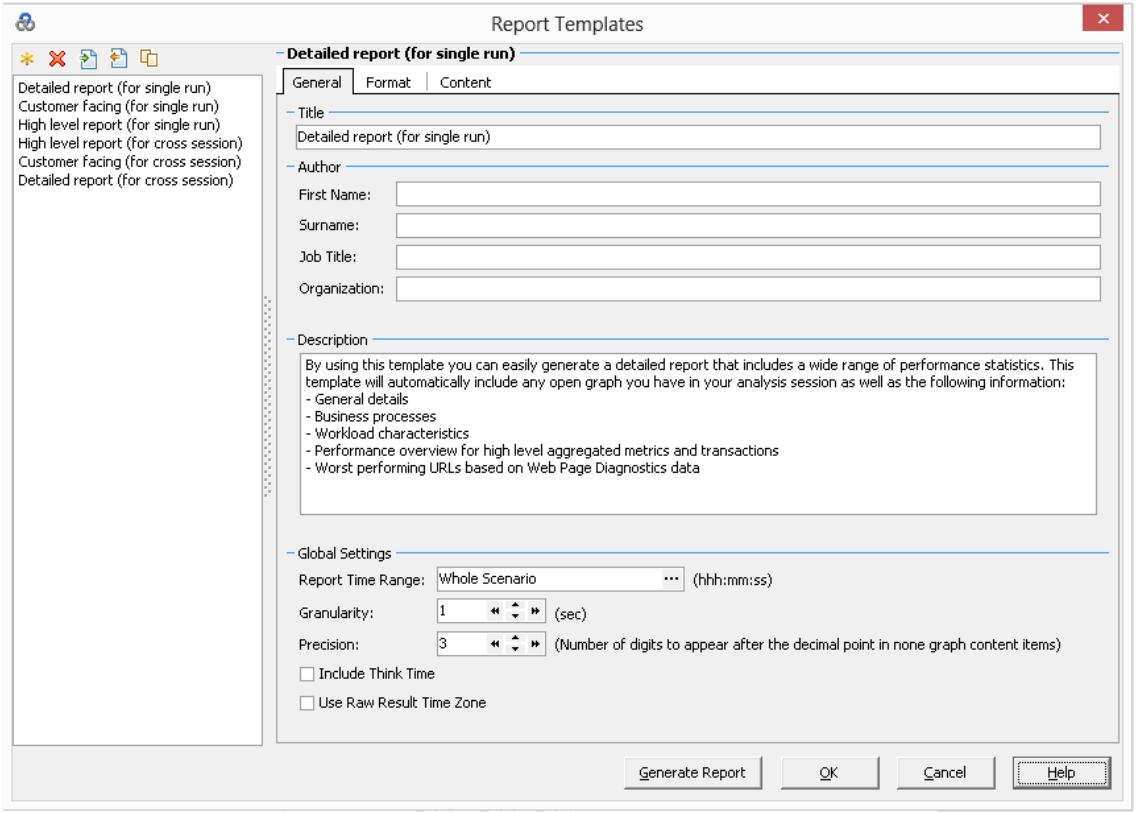
You can use Report Templates to create and customize templates which are used when generating reports. Report templates can be used across similar scenario runs and saves time and effort on recreating reports each time.

Using the Report Templates dialog box, you can record document details, define the format of the report, and select the content items to include in the report and configure each content item accordingly.

A list of report templates is displayed in the **Templates** dialog box, under **Rich Reports**. Select this option if you want to generate the report in the load run session in word, excel, HTML or PDF format. For more information on templates, see "Apply/Edit Template Dialog Box" on page 1323.

## Report Templates Dialog Box

This dialog box enables you to add, modify, import, export, or duplicate a report template.




| **To access** | **Reports > Report Templates** |
|---|---|
| **See also** | • "Report Templates Overview" on the previous page<br>• "New Report Dialog Box" on page 1538<br>**Note:** This dialog box and the New Report dialog box utilize the same components. |

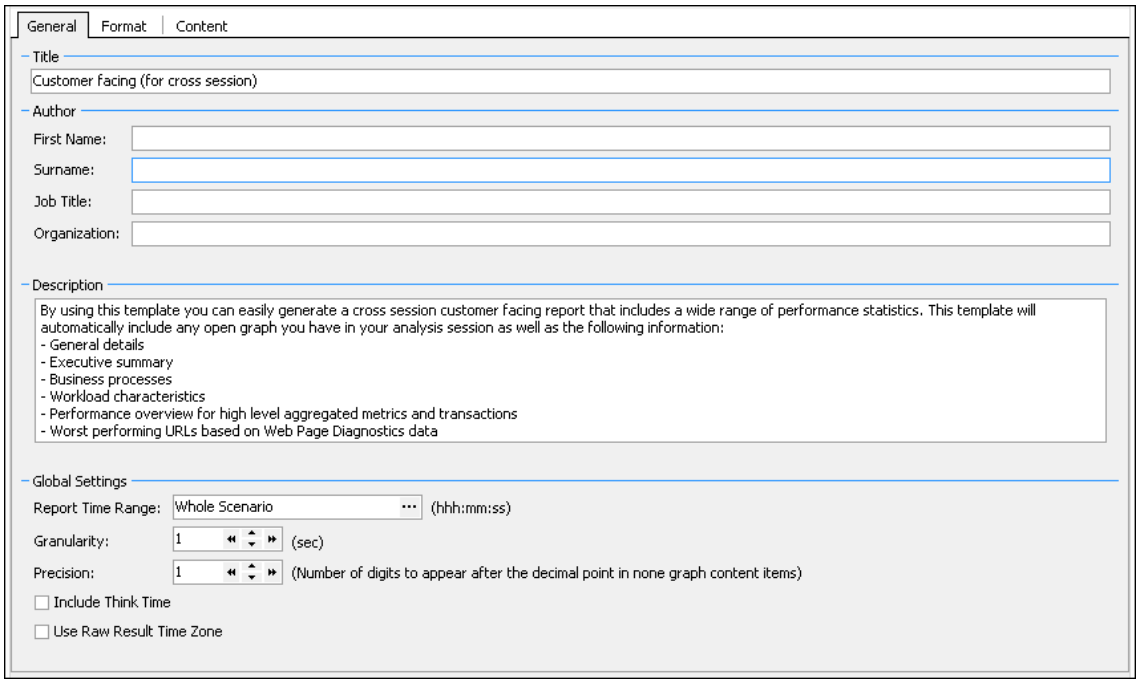
User interface elements are described below:

| UI Element | Description |
|---|---|
| | **New.** Adds a new report template. |
| | **Delete.** Removes the selected template. |
| | **Import.** Imports a report template from an XML file. |
| | **Export.** Saves the selected template as an XML file. |
| | **Duplicate.**Creates a copy of the selected template. |

| UI Element | Description |
|---|---|
| **General tab** | For user interface details, see "Report Templates - General Tab" below. |
| **Format tab** | For user interface details, see "Report Templates - Format Tab" on the next page. |
| **Content tab** | For user interface details, see "Report Templates - Content Tab" on page 1544. |
| **Generate Report button** | Generates the report according to your settings. |

## Report Templates - General Tab

This tab enables you to record document details, such as title, author name and title and set global settings, such as Report Time Range and granularity.



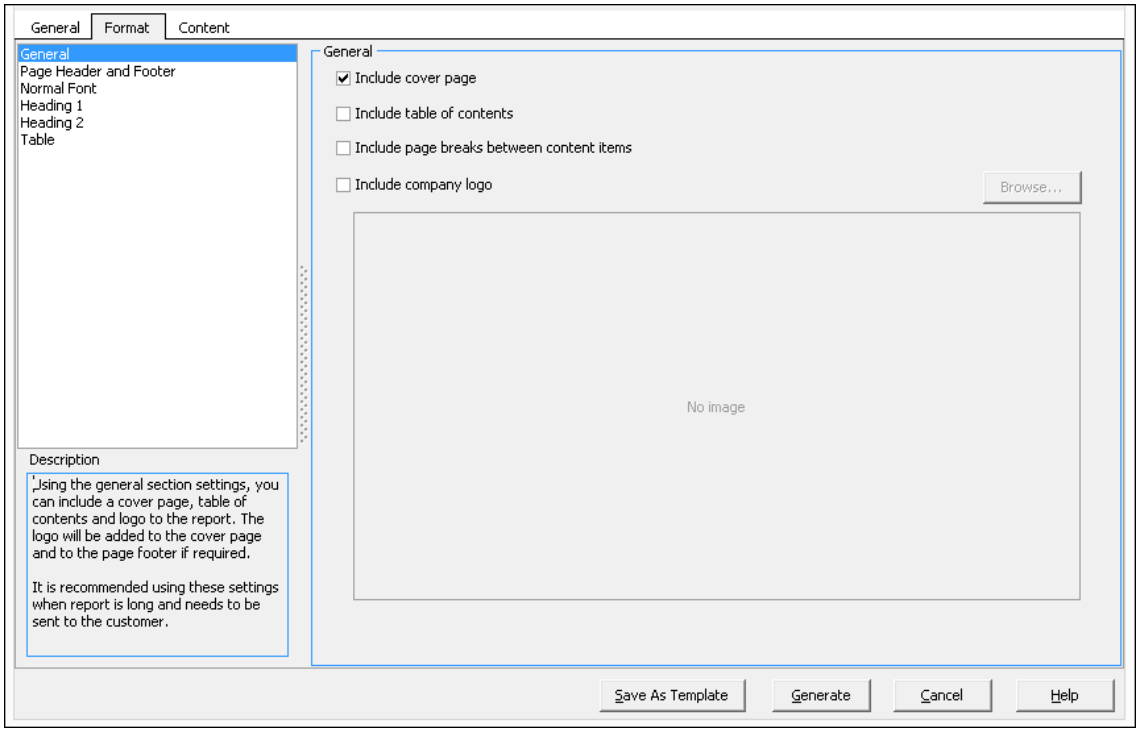

| | |
|---|---|
| **To Access** | **Reports** > **New Report...** > **General** tab<br>or<br>**Reports** > **Report Templates...** > **General** tab |
| **See also** | • "Report Templates Overview" on page 1539<br>• "New Report Dialog Box" on page 1538<br>• "Report Templates Dialog Box" on page 1539 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Title** | A description of the template. |
| **First Name** | The first name of the person to display on the report. |
| **Surname** | The last name of the person to display on the report. |
| **Job title** | The job title of the person to display on the report. |
| **Organization** | The name of the organization to display on the report. |
| **Description** | You can enter a description and include details of the report template. |
| **Report Time Range** | The default setting is Whole Scenario. Click to set the start and end time range of the scenario runtime to display on the report. |
| **Granularity** | Define granularity settings (in seconds). |
| **Precision** | The number of digits to appear after the decimal point in none graph content items. |
| **Include Think Time** | Include think time when processing the Analysis data. This data is then used when generating reports. |
| **Use Raw Result Time Zone** | When creating the report, use the time zone that was generated in the raw data results. |

## Report Templates - Format Tab

This tab enables you to define the format of report template.




| | |
|---|---|
| **To access** | **Reports** > **New Report...** > **Format** tab<br>or<br>**Reports** > **Report Templates...** > **Format** tab |
| **See also** | • "Report Templates Overview" on page 1539<br>• "New Report Dialog Box" on page 1538<br>• "Report Templates Dialog Box" on page 1539 |

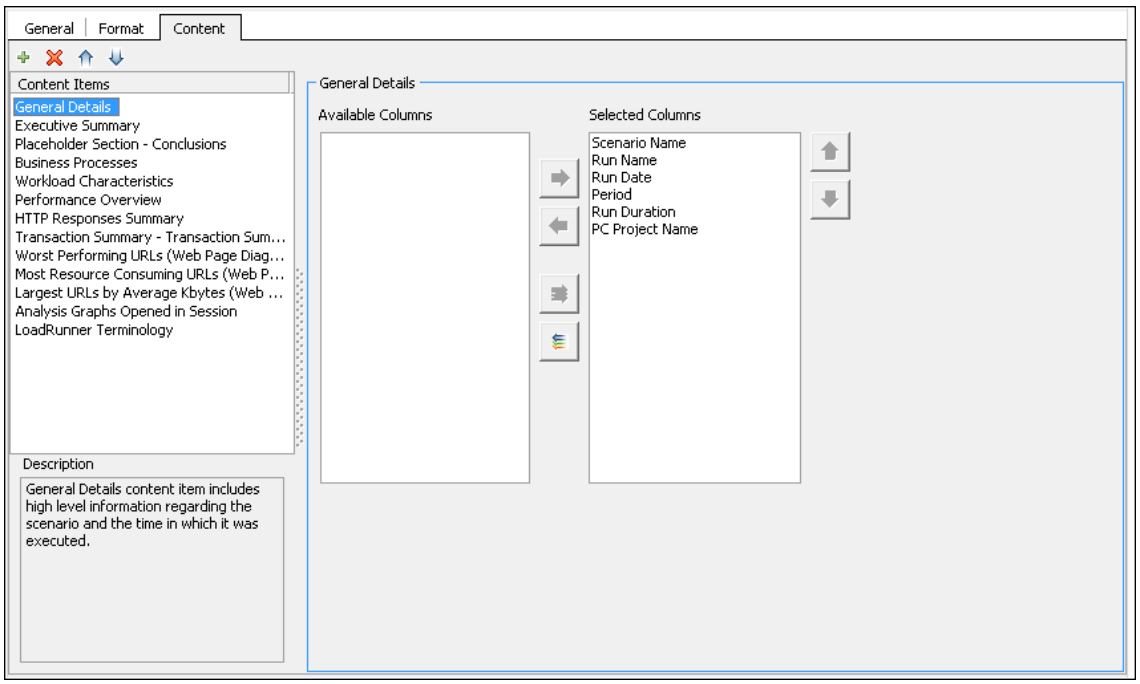
User interface elements are described below:

| UI Element | Description |
|---|---|
| **General** | General options such as:<br>• include a cover page<br>• include table of contents<br>• include company logo |

| UI Element | Description |
|---|---|
| **Page Header and Footer** | Header and footer options:<br>• Font type, size and color<br>• Bold, italicize, or underline<br>• Right, center or left align<br>• You can add tags, such as date, name or organization.<br>• You can include required details such as page count, date, name, and so forth on the left, center or right column. |
| **Normal Font** | The type of font to use in the report template. |
| **Heading 1/2** | The style for your headings. |
| **Table** | Table format options:<br>• Font type, size and color<br>• Background color<br>• Bold, italicize, or underline<br>• Right, center or left align |

## Report Templates - Content Tab

This tab enables you to select the content items to include in the report and configure each item accordingly.



| | |
|---|---|
| **To access** | **Reports** > **New Report...** > **Content** tab<br>or<br>**Reports** > **Report Templates...** > **Content** tab |
| **See also** | • "Report Templates Overview" on page 1539<br>• "New Report Dialog Box" on page 1538<br>• "Report Templates Dialog Box" on page 1539 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| | **Add Content.** Opens the **Add Content Items** pane. Select one or more items from the grid and click **OK**. |
| | **Delete Content.** Removes the selected items from the Content Items pane. |
| | **Reorder.** Reorders the content items, determining how they will be shown in the report. |
| **Contents Item pane** | A list of the content items to be included in the report.<br>• To add more items, click the Add Content button.<br>• To learn about a content item, select it and view the information in the Description pane beneath it. |

| UI Element | Description |
|---|---|
| **<Configuration pane>** | Settings for the selected content item. The components and tabs in this pane vary, based on the selected content item.<br>• **Parameters tab.** Settings such as integer values for percentiles or number of elements.<br>• **Columns tab.** Allows you to select the columns to include in the report. To include a column, make sure it appears in the **Selected Columns** pane.<br>• **Filter tab.** Allows you to enter criteria for including a specific range of a measurement.<br>• **Text area.** A rich text box for enter free text, such as in a Placeholder Section or an Executive Summary.<br>**Tip:** For the **Performance Summary** content item, you can retrieve different information about transactions such as the total number of passed or failed transactions. The item, **Weighted Average of Transaction Response time** is calculated based on the following formula: *Round (Sum of Average value in transaction response time / Sum of transactions)*. For example if you have three transactions with the response times of 0.005, 0.004, and 0.003, the weighted Average of Transaction Response Time is Round((0.005 + 0.004 + 0.003)/3) = 0.004 |
| **Generate Report** | Generates the report according to your settings. |

# Analysis Report Types

## Summary Report Overview

The Summary report provides general information about load test scenario execution. This report is always available from the Session Explorer or as a tab in the Analysis window.

The Summary report lists statistics about the scenario run and provides links to the following graphs: Running Vusers, Throughput, Hits Per Second, HTTP Responses per Second, Transaction Summary, and Average Transaction Response Time.

The appearance of the Summary report and the information displayed, will vary depending on whether an SLA (Service Level Agreement) was defined. An SLA defines goals for the scenario. LoadRunner measures these goals during the scenario run, and analyzes them in the Summary report. For more information on defining an SLA, see "SLA Reports" on page 1552.

If the scenario includes Network Virtualization and virtual locations have been defined, the Summary report includes a breakdown of data based on the virtual locations in the scenario. This enables you to analyze and compare the transaction behavior between different virtual locations.

A Summary report is also provided for Cross Result graphs. For more information about Cross Result graphs, see "Cross Result Graphs Overview" on page 1358.

> **Note:** You can save the Summary reports to an Excel file by selecting **View > Export Summary to Excel** or by clicking the **Export Summary to Excel** button on the toolbar.

## Analysis Summary Report

The Analysis Summary report provides general information about the execution of the load test scenario. It lists statistics about the scenario run and provides links to the following graphs: Running Vusers, Throughput, Hits Per Second, HTTP Responses per Second, Transaction Summary, and Average Transaction Response Time.

| | |
|---|---|
| **To access** | **Session Explorer > Reports > Summary Report** |
| **Important information** | The Summary report for Diagnostics provides a usage chart that links to and displays each individual transaction's Web, application, and database layers, and provides the total usage time for each transaction. |
| **Relevant tasks** | You can save the Summary reports to an Excel file by selecting **View > Export Summary to Excel** or by clicking on the toolbar. |

### Summary Report with No SLA

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Scenario Details** | Shows the basic details of the load test scenario being analyzed. |
| **Statistics Summary** | This section shows a breakdown of the transaction statistics and also provides links to the following:<br>• The SLA configuration wizard. For more information on defining an SLA, see "SLA Reports" on page 1552<br>• The Analyze Transaction tool. For more information on analyzing transactions, see "Analyze Transactions Dialog Box" on page 1536 |

| UI Element | Description |
|---|---|
| **Transaction Summary** | This section displays a table containing the load test scenario's diagnostics data. Included in this data is a percentile column (x Percent). This column indicates the maximum response time for that percentage of transactions performed during the run. All transaction response times are calculated based on "passed" transactions data.<br><br>**Note**: You can change the value in the percentile column in one of the following ways:<br><br>• Open the Options dialog box (**Tools > Options**). Click the **General** tab and in the Summary Report section enter the desired percentile in the Transaction Percentile box.<br><br>• Select **View > Summary** Filter or click on the toolbar. The Analysis Summary Filter dialog box opens. In the Additional Settings area enter desired percentile. |
| **HTTP Responses Summary** | This section shows the number of HTTP status codes returned from the Web server during the load test scenario, grouped by status code.<br><br>**Note**: There are additional Diagnostics sections that may appear at the end of the Summary report, depending on the configuration of your system. |

## Summary Report with SLA

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Scenario details** | This section shows the basic details of the load test scenario being analyzed. |
| **Statistics Summary** | This section shows a breakdown of the transaction statistics. |

| UI Element | Description |
|---|---|
| **X Worst Transactions** | The X Worst Transactions table shows the worst transactions in terms of how often the transactions exceeded the SLA boundary during the run, and by how much.<br><br>**Note**: You choose how many transactions are displayed in this table in the Summary Report section on the General tab of the options dialog box. Open the dialog box (**Tools** > **Options**) and enter the number of transactions to display. The default is **5**.<br><br>You expand a transaction to get more information. When expanded, the following information appears for each transaction:<br><br>• **Failure Ratio.** The percentage of time intervals where the transaction exceeded the SLA. You can see this graphically in the **Scenario Behavior Over Time** section below.<br>• **Failure Value.** The average percentage by which the transaction exceeded the SLA over the whole run.<br>• **Avg exceeding ratio.** The average percentage by which the transaction exceeded the SLA over a specific time interval.<br>For example, say in one time interval the figure is 4.25%. This means that during that time interval, the transaction may have exceeded the SLA boundary several times, each time by a different percentage margin, the average percentage being 4.25%.<br>• **Max exceeding ratio.** The highest percentage by which the transaction exceeded the SLA over a specific time interval.<br>For example, using the same time interval as above, the transaction may have exceeded the SLA several times, each time by a different percentage margin. The highest percentage being 7.39%<br>Analysis allows you to analyze a specific transaction in more detail. You open the Analyze Transaction tool from this section by clicking the **Analyze Transaction** button. For more information on Transaction Analysis Reports, see "Analyze Transactions Dialog Box" on page 1536. |

| UI Element | Description |
|---|---|
| **Scenario Behavior Over Time** | This section shows how each transaction performed in terms of the SLA over time intervals. The green squares show time intervals where the transaction performed within the SLA boundary. Red squares show time intervals where the transaction failed and gray squares show where no relevant SLA was defined.<br><br>**Note:** The time intervals displayed in the **Scenario Behavior Over Time** section may vary for each interval. The time interval set in the tracking period of the SLA is only the minimum time interval that will be displayed.<br><br>It is only the display that varies. The SLA is still determined over the time interval you choose in the **Advanced Settings** section.<br><br>Analysis allows you to analyze a specific transaction in more detail. You open the Analyze Transaction tool from the **Scenario Behavior Over Time** section in one of the following ways:<br><br>• Select the transaction to analyze from the list and enter the time interval in the **From** and **To** boxes. Then click **Analyze Transaction**.<br>• Drag the mouse over the desired transaction and time range to analyze. Then click **Analyze Transaction**.<br><br>For more information on Transaction Analysis Reports, see "Analyze Transactions Dialog Box" on page 1536. |
| **Transaction Summary** | This section displays a table containing the load test scenario's diagnostics data. Included in this data is a percentile column (x Percent). This column indicates the maximum response time for that percentage of transactions performed during the run. All transaction response times are calculated based on "passed" transactions data.<br><br>**Note:** You can change the value in the percentile column in the Summary Report section of the General tab of the Options dialog box. Open the dialog box (**Tools > Options**) and enter the desired percentage.<br><br>Alternatively, you can also change the value in the Summary Filter (**View > Summary Filter**) . |
| **HTTP Responses Summary** | This section shows the number of HTTP status codes returned from the Web server during the load test scenario, grouped by status code.<br><br>**Note:** There are additional Diagnostics sections that may appear at the end of the Summary report, depending on the configuration of your system. |

## Summary reports for Cross Result Graphs

User interface elements are described below:

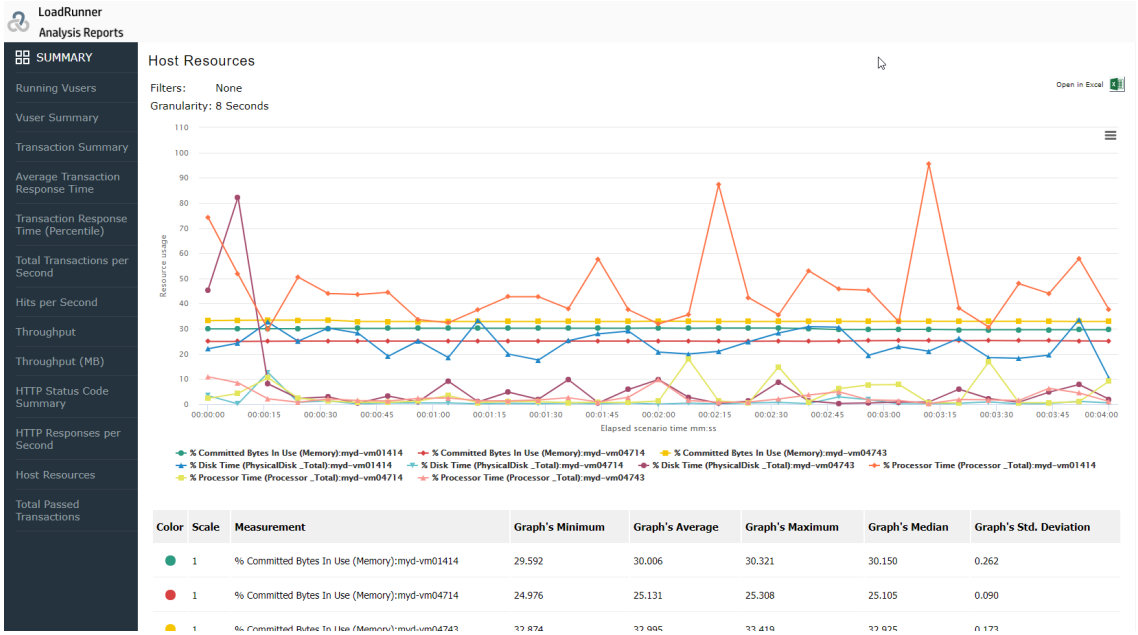
| UI Element | Description |
|---|---|
| **<graphs>** | Displays summary information for the scenarios that you are comparing. The information is displayed in a way that enables you to compare data from the different scenarios. Includes the same type of information as the regular Summary report except for the following:<br>• SLA information<br>• Diagnostics information<br>• Scenario behavior over time |

## See also:

- "J2EE and .NET Diagnostics Graphs Summary Report" on page 1432

# HTML Reports

Analysis enables you to create HTML reports for your load test scenario run. It creates a separate page for each one of the open graphs and reports.




| | |
|---|---|
| **To access** | Use one of the following:<br>• **Reports > HTML Report**<br>• **Toolbar >** |

| | |
|---|---|
| **Relevant tasks** | • Open all graphs that you want to include in the report.<br>• Specify a path and file name for the HTML report and click **Save**. Analysis saves a Summary report which has the same name as the file in the selected folder. The rest of the graphs are saved in a folder with the same name as the Summary report's file name. When you create an HTML report, Analysis opens your default browser and displays the Summary report.<br>• To copy the HTML reports to another location, be sure to copy the file name and the folder with the same name. For example, if you named your HTML report `test1`, copy `test1.html` and the folder `test1` to the desired location. |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **<Graphs> menu left frame** | Click the graph link to view an HTML report for that graph. |
| | You can view an Excel file containing the graph data, by clicking the **Open in Excel** button on the relevant graph page. |
| | **Context Menu**. Right-click to print the graph, or save it as an image (.png, .jpeg, or .svg) or PDF. |
| Graph pane | Hover over the graph to view details of the displayed measurements. |
| Legend | Click a measurement on the legend to show or hide it from the graph. |

## SLA Reports

An SLA (Service Level Agreement) defines goals for the load test scenario. LoadRunner measures these goals during the scenario run and analyzes them in the Summary report. The SLA Report shows the succeeded or failed status of all SLAs that were defined for the scenario run.

**Note:** Analysis data (for example, transactions) that has been excluded by the Summary Filter will not be available for analysis in the SLA report.

| | |
|---|---|
| **To access** | You create the SLA Report in one of the following ways**:**<br>**Reports > Analyze SLA**<br>**Right-click the Summary pane > Add New Item > Analyze SLA**<br>**Summary Report >** |
| **Relevant tasks** | "Defining Service Level Agreements" on page 1300 |

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Display of SLA statuses** | **SLA Status per goal definition**<br>• Where the SLA was defined over the whole run, the report displays a single SLA status for each goal definition.<br>**SLA status for each transaction per time interval**<br>• Where the SLA was defined per time interval within the run, the report displays the status of the SLA for each transaction per time interval. The green squares show time intervals where the transaction performed within the SLA boundary. Red squares where the transaction failed and gray squares show where no relevant SLA was defined.<br>**SLA goal definitions**<br>• Where the SLA was defined per time interval within the run, a further section appears detailing the goal definitions for the SLA. |

## Transaction Analysis Report

This report enables you to individually examine each of the transactions from the load test scenario run.

| To access | **Reports > Analyze Transaction > Generate Report button** |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Observations** | This section shows both positive and negative correlations between the graph of the transaction being analyzed, and other graphs based on the settings you chose in the Analyze Transaction Dialog Box. When two graphs are correlated, it means that their behavior matches each other by a certain percentage.<br> To view the correlating graph, select one of the results and then click the View Graph icon at the bottom of the section. The graph comparison opens.<br> You can return to the Transaction Analysis Report from the graph comparison at anytime by clicking the Back to <transaction name> icon on the toolbar.<br>**Note**: The correlations are automatically calculated according to a default ratio of 20%. You can adjust this ratio by clicking the arrows next to the percentage. Then click **Recalculate**. |

| UI Element | Description |
|---|---|
| **Errors** | This section is divided into two sub-sections.<br>• **Application Under Test errors.** Shows errors that occurred during the transaction that were direct results of Vuser activity.<br>• **All errors.** Shows Application Under Test errors as well as errors that were not related to Vuser activity, and which affected your system and not the application under test. |
| **Observation Settings** | This section displays a summary of the settings that were selected in the Advanced Settings section of the Analyze Transaction dialog box. |
| **Graph** | The Graph section displays a snapshot of selected transaction and time range for analysis merged with the display option you selected (Running Vusers, Throughput, or Hits per Second). Note that it is only a snapshot and cannot be manipulated like normal graphs. |

# Importing Data

### What do you want to do?

- Import data
- Define a custom file format

### See also:

- Supported file types
- Import Data dialog box

# Import Data Tool Overview

The LoadRunner Analysis Import Data tool enables you to import and integrate non-Micro Focus data into a LoadRunner Analysis session. After the import procedure, you can view the data files as graphs within the session, using all the capabilities of the Analysis tool.

Suppose an NT Performance Monitor runs on a server and measures its behavior. Following a LoadRunner scenario on the server, you can retrieve the results of the Performance Monitor, and integrate the data into LoadRunner's results. This enables you to correlate trends and relationships between the two sets of data: LoadRunner's and the Performance Monitor's.

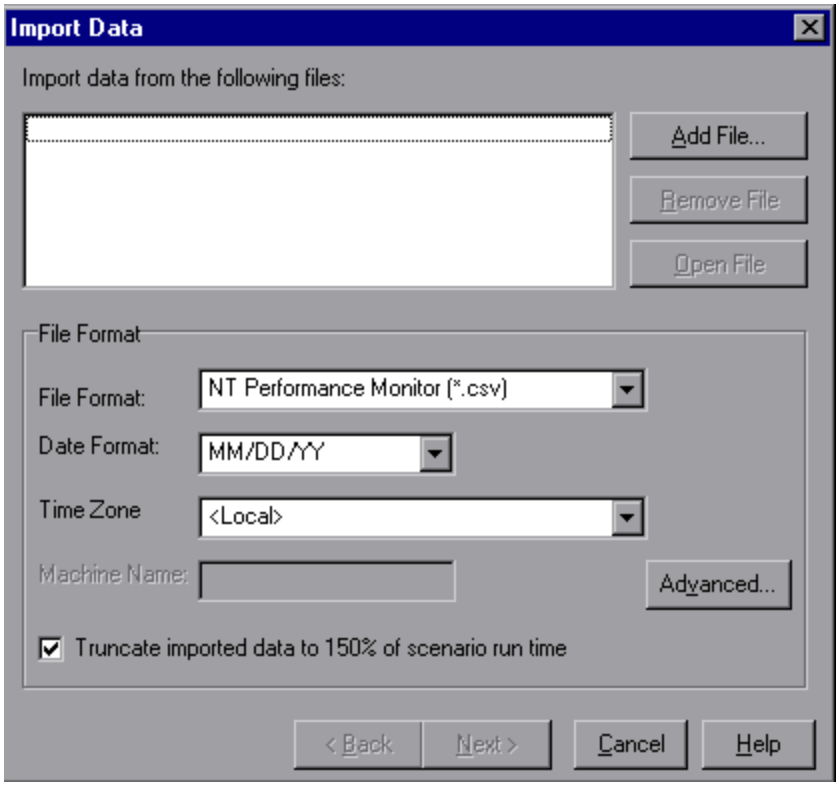
In this case, the results of the NT Performance Monitor are saved as a **.csv** file. You launch the Import Data tool, direct it to the **.csv** file, and specify its format. LoadRunner reads the file and integrates the results into its own Analysis session.

For a list of data formats that are supported, see "Supported File Types" on page 1556. To define your own custom data files, see "Define Custom File Formats" on page 1556.

# Import data into your Analysis session

This task describes how to import data files to integrate into your analysis session.

1. Choose **Tools > External Monitors > Import Data**. The Import Data dialog box opens.




2. Select the format of the external data file from the **File format** list box.
3. Click **Add File**. In the **Select File to Import** dialog box that opens, the **Files of type** list box shows the type chosen in step 2.
4. Set other file format options, as described in "Import Data Dialog Box" on page 1561. You must enter a machine name.
5. To specify character separators and symbols, click **Advanced**. For more information, see "Advanced Settings Dialog Box (Import Data Dialog Box)" on page 1558.
6. Click **Next**. The Import Data dialog box opens.
7. Select the type of monitor that generated the external data file. If your monitor type does not exist, you can add it, as described in How to Customize Monitor Types for Import.

   When opening a new graph, you will see your monitor added to the list of available graphs under this particular category. (See "Open a New Graph Dialog Box" on page 1371.)
8. Click **Finish**. LoadRunner Analysis imports the data file or files, and refreshes all graphs currently displayed in the session.

> **Note:** When importing data into a scenario with two or more cross results, the imported data

will be integrated into the last set of results listed in the **File > Cross with Result** dialog box. For more information, see "Generating Merged Graphs" on page 1361.

# Define Custom File Formats

This task describes how to define a custom format, if the file format of your import file is not supported.

If the file format of your import file is not supported, you can define a custom format.

1. Choose **Tools > External Monitors > Import Data**. The Import Data dialog box opens.
2. From the **File Format** list, select **<Custom File Format>**. The Enter New Format Name dialog box opens.
3. Enter a name for the new format (in this case, `my_monitor_format`).
4. Click **OK**. The Define External Format dialog box opens.
5. Specify the mandatory and optional data, as described in "Define External Format Dialog Box" on page 1559.
6. Click **Save**.

# Supported File Types

This section describes the supported file types.

## NT Performance Monitor (.csv)

The default file type of the NT Performance monitor, in comma separated value (CSV) format.

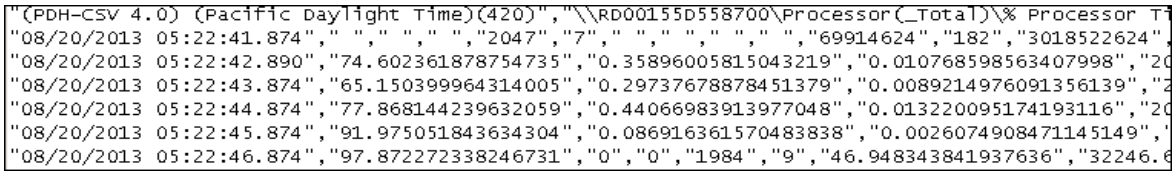
For example:

**Example:**
```
Reported on \\WINTER
Date: 03/06/15
Time: 10:06:01 AM
Data: Current Activity
Interval: 1.000 seconds
,,% Privileged Time,% Processor Time,% User Time,
,,0,0,0,
,,,,,,
,,Processor,Processor,Processor,
Date,Time,\\WINTER,\\WINTER,
03/06/15,10:06:00 AM , 0.998, 1.174, 0.000,
03/06/15,10:06:00 AM , 0.000, 0.275, 0,000,
```

## Windows Performance Monitor (.csv)

The default file type for the Windows 2000, 2008 server, Windows 7, etc. performance monitor, in CSV format.

For example:



```
"(PDH-CSV 4.0) (Pacific Daylight Time)(420)","\\RD00155D558700\Processor(_Total)\% Processor Ti
"08/20/2013 05:22:41.874"," "," "," ","2047","7"," "," "," "," ","69914624","182","3018522624",
"08/20/2013 05:22:42.890","74.602361878754735","0.35896005815043219","0.010768598563407998","20
"08/20/2013 05:22:43.874","65.150399964314005","0.29737678878451379","0.0089214976091356139","2
"08/20/2013 05:22:44.874","77.868144239632059","0.44066983913977048","0.013220095174193116","20
"08/20/2013 05:22:45.874","91.975051843634304","0.086916361570483838","0.0026074908471145149","
"08/20/2013 05:22:46.874","97.872272338246731","0","0","1984","9","46.948343841937636","32246.6
```

## Standard Comma Separated File (.csv)

This file type has the following format:

date, time, measurement_1,measurement_2, ...

where fields are comma separated and first row contains column titles.

The following example from a standard CSV file shows 3 measurements: an interrupt rate, a file IO rate and a CPU usage. The first row shows an interrupt rate of 1122.19, an IO rate of 4.18, and a CPU busy percentage of 1.59:

**Example:**
```
date, time, interrupt rate, File IO rate, CPU busy percent
03/06/15,10:06:01,1122.19,4.18,1.59
03/06/15,10:06:01,1123.7,6.43,1.42
```

## Master-Detail Comma Separated File (.csv)

This file type is identical to Standard Comma Separated Files except for an additional **Master** column which specifies that row's particular breakdown of a more general measurement. For example, a Standard CSV file may contain data points of a machine's total CPU usage at a given moment:

Date,Time,CPU_Usage

However, if the total CPU usage can be further broken up into CPU time per-process, then a Master-Detail CSV file can be created with an extra column **ProcessName**, containing the name of a process.

Each row contains the measurement of a specific process's CPU usage only. The format will be the following:

Date,Time,ProcessName,CPU_Usage

as in the following example:

**Example:**
```
date, time, process name, CPU used, elapsed time used
03/06/15,10:06:01,edaSend,0.1,47981.36
```

```
03/06/15,10:06:01,PDS,0,47981.17
```

## Microsoft Excel File (.xls)

Created by the Microsoft Excel application. The first row contains column titles. (.xlxs format is not supported.)

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | date | time | interrupt rate | File IO rate | CPU busy percent |
| 2 | 3/6/2015 | 10:06:01 | 1122.19 | 4.18 | 1.59 |
| 3 | 3/6/2015 | 10:06:01 | 1123.7 | 6.43 | 1.42 |
| 4 | 3/6/2015 | 10:12:01 | 1103.62 | 5.33 | 1.17 |
| 5 | 3/6/2015 | 10:14:01 | 1118.89 | 12.18 | 2.37 |
| 6 | 3/6/2015 | 10:15:01 | 1116.89 | 19.85 | 3.87 |
| 7 | 3/6/2015 | 10:16:01 | 1128.12 | 19.9 | 4.15 |
| 8 | 3/6/2015 | 10:06:01 | 1151.98 | 20.82 | 4.25 |

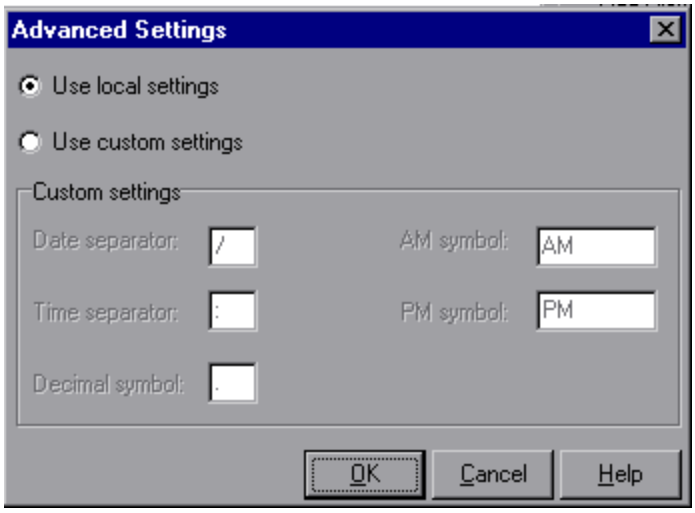
## Master-Detail Microsoft Excel file (.xls)

Created by Microsoft's Excel application. The first row contains column titles. It contains an extra **Master** column. (.xlxs format is not supported.)

| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | date | time | process name | CPU used | elapsed time used |
| 2 | 3/6/2015 | 10:06:01 | edaSend | 0.1 | 47981.36 |
| 3 | 3/7/2015 | 11:06:01 | PDS | 0 | 47981.17 |

# Advanced Settings Dialog Box (Import Data Dialog Box)

This dialog box enables you to define the data format of the imported file to settings other than of the regional configuration.




| To access | Tools > External Monitors > Import Data > Advanced |
|---|---|

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Use local settings** | Keep default settings of the regional configuration. Disables the **Custom Settings** area of the dialog box. |
| **Use custom settings** | Define your own settings. Enables the **Custom Settings** area of the dialog box.<br>• **Date Separator.** Enter a custom symbol, for example, the slash (/) character in 11/10/17<br>• **Time Separator.** Enter a custom symbol, for example, the colon (:) character in 9:54:19<br>• **Decimal symbol.** Enter a custom symbol, for example, the '.' character in the number 2.5<br>• **AM symbol.** Enter a custom symbol for the hours between midnight and noon.<br>• **PM symbol.** Enter a custom symbol for the hours between noon and midnight. |

# Define External Format Dialog Box

This dialog box enables you to define a new file format for external data files not supported by Analysis.

The Define External Format dialog box is divided into mandatory and optional information.

| To access | Tools > External Monitors > Import data > File Format > <Custom File Format> |
|---|---|
| **Relevant tasks** | "Define Custom File Formats" on page 1556 |

## Mandatory tab

User interface elements are described below:

| UI Element | Description |
|---|---|
| **Date Column Number** | Enter the column that contains the date. If there is a master column (see "Supported File Types" on page 1556), specify its number. |
| **Time Column Number** | Enter the column that contains the time. |
| **Use Master Column** | Select this if the data file contains a master column. A master column specifies the row's particular breakdown of a more general measurement. |
| **File Extension** | Enter the file suffix. |
| **Field Separator** | Enter the character that separates a field in a row from its neighbor. To select a field separator character, click **Browse** and select a character from the define Field Separator dialog box. |

## Optional tab

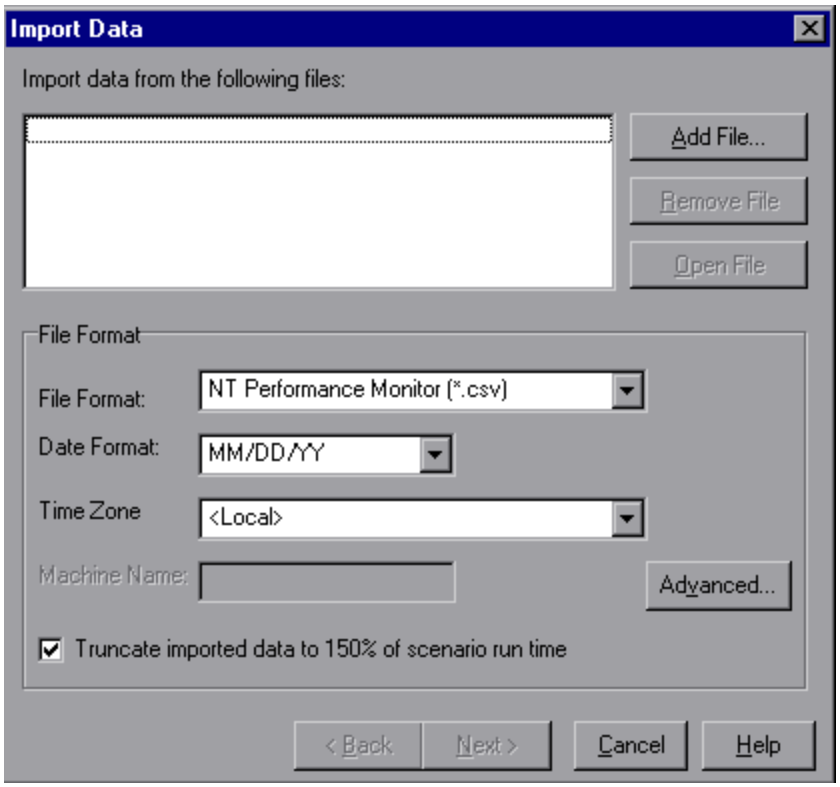
User interface elements are described below:

| UI Element | Description |
|---|---|
| **Date Format** | Specify the format of the date in the imported data file. For example, for European dates with a 4 digit year, choose **DD/MM/YYYY**. |
| **Time Zone** | Select the time zone where the external data file was recorded. LoadRunner Analysis aligns the times in the file with local time zone settings to match LoadRunner results. (LoadRunner does not alter the file itself). |
| **Machine Name** | Specify the machine name the monitor runs on. This associates the machine name with the measurement. |
| **Exclude Columns** | Indicate which columns are to be excluded from the data import, such as columns containing descriptive comments. When there is more than one column to be excluded, specify the columns in a comma-separated list. For example, 1,3,8. |

| UI Element | Description |
|---|---|
| **Convert file from UNIX to DOS format** | Monitors often run on UNIX machines. Check this option to convert data files to Windows format. A carriage return (Ascii character 13) is appended to all line feed characters (Ascii character 10) in the UNIX file. |
| **Skip the first [] lines** | Specify the number of lines at the start of the file to ignore before reading in data. Typically, the first few lines in a file contain headings and sub-headings. |

# Import Data Dialog Box

This dialog box enables you to import and integrate non-Micro Focus data files into an Analysis session.




| To access | Tools > External Monitors > Import Data |
|---|---|

User interface elements are described below (unlabeled elements are shown in angle brackets):

| UI Element | Description |
|---|---|
| **Import data from the following files** | Displays the files that you selected for import. |

| UI Element | Description |
|---|---|
| **Add file** | Select an external data file to import. A dialog box opens to enable you to select files. |
| **Remove file** | Delete an external data file from the list. |
| **Open File** | Open an external data file using the associated application. |
| **File Format** | Set the file format options.<br>• **File Format**. Choose the format of the external data file. For an explanation of available formats, see "Supported File Types" on page 1556.<br>• **Date Format.** Specify the format of the date in the imported data file. For example, for European dates with a 4 digit year, choose **DD/MM/YYYY**. |
| **Time Zone** | Select the time zone where the external data file was recorded. LoadRunner Analysis compensates for the various international time zones and aligns the times in the file with local time zone settings in order to match LoadRunner results. If the times in the imported file are erroneous by a constant offset, you can synchronize the time. |
| **<Synchronize with scenario start time>** | Time Zone also contains the option **<Synchronize with scenario start time>**. Choose this to align the earliest measurement found in the data file to the start time of the LoadRunner scenario. |
| **Machine Name** | Specify the machine name the monitor runs on. This associates the machine name with the measurement. For example, a file IO rate on the machine `fender` will be named `File IO Rate:fender`. This enables you to apply Graph settings by the machine name. For more information, see "Filtering and Sorting Graph Data" on page 1342. |
| **Advanced** | For more information, see "Advanced Settings Dialog Box (Import Data Dialog Box)" on page 1558. |
| **Truncate imported data to 150% of scenario runtime** | In certain cases, the external monitor may have collected data over a time period that was larger than the actual load test. This option deletes data that was collected while the load test was not running, limiting the data collection period to 150% of the load testing period. |

# Troubleshooting and Limitations for Analysis

This section contains troubleshooting and limitations for Analysis.

# General

- If the behavior of Analysis is unpredictable and unexpected messages appear, this might be a result of UAC Virtualization having been enabled for Analysis. You can disable UAC Virtualization on the **Analysis.exe** process in the Windows Task Manager.
- Analysis API works only on x86 platforms. If you are using Visual Studio, define the platform as x86 in the project options.
- When analyzing results from a load test in which the Web Vusers accesses the AUT through a proxy server, the **Time to First Buffer Breakdown** graph shows only zero values for Network Time and Server Time. This is because the "time to first buffer" metric is turned off when working behind a proxy, and the time values can only be calculated to the proxy server.
- Load results that contain transactions with the '@' or ',' characters may conflict with existing transactions. This is because Analysis attempts to replace those characters with the '_', and if this results in a transaction name conflict, an error will occur.

  **Workaround**: Avoid using the '@' and ',' characters in transaction names.
- The following Analysis default settings have been modified: **Include Think Time** is disabled and **Display summary while generating complete data** is enabled.
- When exporting Analysis reports to MS Word, the content load may affect the table format within the document. The recommended format is RTF.
- If the results take a long time to load, make sure that the **Use cached file to store data** option in **Tools > Options > General** tab is disabled. You should only enable this for very large result files. For details, see "General Tab (Options Dialog Box)" on page 1279.

# Graphs

- When the Analysis results consists of a large number of similar measurements, you may experience spikes in graphs, or an *Out of memory* message.

  **Workaround:** For 64-bit Windows, make sure that you have 4 GB or more memory. F or 32-bit Windows, Select **Start > Run**, and type msconfig. In the **Boot** tab, click **Advanced Options**. Select **Maximum memory** and set it to the maximum value.
- After running a Language Pack, the Analysis data generated from the sample session (in the <LoadRunner root>\tutorial folder) is displayed in English and filtering cannot be applied.

  **Workaround:** Regenerate the graphs.
- The Transaction Response Time (Percentile) graph may show inaccurate results.

  **Workaround:**

  a. Close the Analysis application.

  b. Open the **C:\Program Files (x86)\Micro Focus\LoadRunner\bin\dat\percentile.def** file

  c. In the **[Graph Definition]** section, set BasicTableName to an empty string:
  **[Graph Definitions]**

**BasicTableName=**

d. Open Analysis again and view the graph.

# ALM Integration

- When trying to save an Analysis session to the ALM repository with CAC on IIS, you may encounter an error message indicating that the session cannot be saved and that the connection is unavailable.

  **Workaround:** Increase the size of the **uploadReadAHeadSize** parameter to 16 MB or higher, and restart IIS. You can use the command line: `C:\Windows\System32\inetsrv\appcmd.exe set config "Default Web Site" -section:system.webServer/ServerRuntime /uploadReadAheadSize:16777216 /commit:apphost`
- After running a Language Pack, the Analysis data generated from the sample session (in the <LoadRunner root>\tutorial folder) is displayed in English and filtering cannot be applied.

  **Workaround:** Regenerate the graphs.

# Microsoft SQL Server

- If you are using your own policy in an MS SQL server, you may need to add your own account to the Analysis database template (in the <LoadRunner root>\bin\dat folder).
- Analysis may fail to load results created through an MS SQL database, if the decimal separator on the Analysis machine is different from the decimal separator on the MS SQL Server machine (common on non-English operating systems).

  **Workaround:** Change the decimal separator on Analysis machine to be the same as the MS SQL Server machine.
- Filtering of transactions for MS Access and SQL queries is limited to 100 transactions.
- If you are using Microsoft SQL Server 2000, you need to either migrate Analysis data, or upgrade to Microsoft SQL Server 2005. The following tasks describe how to migrate and upgrade.

**To migrate legacy Analysis data to a SQL 2005 server:**

1. From the SQL Server Management Studio, using Object Explorer, connect to an instance of SQL Server Database Engine.
2. Expand Databases, right-click Analysis database, select Tasks\Copy Database.
3. Follow the instructions in the wizard.

**To upgrade SQL 2000 to SQL 2005:**

1. Uninstall SQL 2000.
2. Install SQL 2005.
3. Restore Analysis data from backup. See http://msdn.microsoft.com/en-us/library/ms177429(SQL.90).aspx.

# Additional Components

You can install additional components that provide advanced features for working with LoadRunner. The setup files are located in the **Additional Components** folder inside the root folder of your LoadRunner installation package.

## Where to install additional components

The table below indicates which additional components are available, and where you should install each component:

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **Agent for Citrix Server** | SetupCitrixAgent.exe | Installs the Citrix Agent which enhances VuGen's capabilities in identifying Citrix client objects during Citrix protocol record and replay. For installation instructions, see "Install the LoadRunner Citrix agent on the Citrix server. (Optional)" on page 480.<br><br>The agent also enables you to use additional Citrix API functions. For details, see the Function Reference (**Help > Function Reference**). | Citrix server |
| **Agent for Microsoft Terminal Server** | SetupMSTerminalAgent.exe | Installs a utility that enhances the RDP protocol's recording mechanism in VuGen. For installation instructions, see "Installing the Microsoft Terminal Server Agent" on page 913. | RDP server |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **Assembly Crawler for Analysis API** | AssemblyCrawlerConsole.exe | Installs a command-line utility to build a .NET configuration file for a LoadRunner Analysis API application. For more information, open the **Analysis API Reference** from the **Start > Documentation** menu on the LoadRunner machine (not available with VuGen Standalone). | LoadRunner Analysis machine |
| **HostID Generator** | Host ID Generator tool, licidgenerator.exe | Opens the Host ID Generator utility that displays the computer's Host ID. This is useful when requesting a license. | LoadRunner Controller machine |
| **Network Virtualization** | • NV4ControllerSetup.exe<br>• NV4LGSetup.exe<br>• Linux setup packages: **<LoadRunner installation package>\Standalone Applications\linux_x64\NV4LG** | • NV4ControllerSetup.exe installs Network Virtualization for Controller, enabling emulation functionality.<br>• NV4LGSetup.exe installs the NV Insights Report component for VuGen, and Network Virtualization for the Windows load generator machines.<br>• The Linux packages install Network Virtualization for the Linux load generator machines.<br>For details, see "Network Virtualization Integration" on page 1237. | LoadRunner Controller, VuGen, and load generator machines |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **IDE Add-Ins** | • EclipseAddin \hp.lr.vugeneclipse42addin.jar<br>• LRVS2013IDEAddInSetup.exe<br>• LRVS2015IDEAddInSetup.exe | Installs add-ins for supported versions of Visual Studio or Eclipse enabling you to create Vuser scripts in your standard development environment using the LoadRunner API. This integration also allows you to run the test directly from Visual Studio or Eclipse, to test its functionality.<br>• Only 32-bit versions of Eclipse are supported. For more details on supported versions, see the System Requirements.<br>• To install the Visual Studio Add-in, Visual Studio must be installed in the default location.<br>• For the LRVS2015IDEAddIn for Visual Studio, Visual C ++ language must be installed to work with C++ .Net Vuser projects.<br>For details, see "Create Scripts in External IDEs" on page 877. | Visual Studio / Eclipse machine with VuGen |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **IDE Add-Ins Dev** | LREclipseIDEAddInDevSetup.exe<br>LRVS2013IDEAddInDevSetup.exe<br>LRVS2015IDEAddInDevSetup.exe | Setup files for developer add-ins for supported versions of Visual Studio and Eclipse, enabling you to create NUnit or JUnit tests in your standard development environment using the LoadRunner API.<br>• Only 32-bit versions of Eclipse are supported. For more details on supported versions, see the System Requirements.<br>• To install the Add-in, Visual Studio must be installed in the default location.<br>• For the LRVS2015IDEAddIn for Visual Studio, Visual C ++ language must be installed to work with C++ .Net Vuser projects.<br>For details, see "Create Scripts in External IDEs" on page 877. | Visual Studio or Eclipse machine with VuGen |
| **LoadRunner ProtocolSDK** | SetupLoadRunnerProtocolSDK.exe | Allows you to create and distribute custom LoadRunner protocols. For details, see "Protocol SDK" on page 912. | Any machine with Virtual Studio 2015 and WiX Toolset 3.10.3 - 3.11 |
| **mobileRemote Agent** | Select the relevant component for your operating system. | Enables you to capture a pcap file with Linux Redhat | |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **SAP Tools** | SapSpy.exe<br>VerifyScripting.exe | • **SAPGUI Spy.** Examines the hierarchy of GUI Scripting objects, on open windows of **SAPGUI Client for Windows**.<br>• **SAPGUI Verify Scripting.** Verifies that the SAPGUI Scripting API is enabled.<br>For details, see "Configure the SAP Environment" on page 677. | VuGen machine with SAPGUI client |
| **Third Parties** | Source files | The folder contains the source code of some third party software components which are being used in LoadRunner. | N/A |
| **Virtual Table Server** | SetupVTS.exe | Virtual Table Server (VTS) offers an alternative to standard LoadRunner parameterization. For details, see "Parameterization Overview" on page 357. | Any machine |

# Standalone Applications

The following LoadRunner standalone applications are available in your LoadRunner installation package's **/Standalone Applications** folder.

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **Analysis Standalone** | SetupAnalysis.exe | Installs LoadRunner Analysis as a standalone application. Install this to open LoadRunner results and create graphs and reports on a separate machine. For details, see "Introducing Analysis" on page 1269. | Any machine |
| **Load Generator** | SetupLoadGenerator.exe | Installs the LoadRunner agent on the machine in order to run load tests. After you install this software, you access this machine from the Controller. For details, see "Load Generators" on page 983. | Any machine |

| Folder | Component | Description | Install on... |
|---|---|---|---|
| **MI Listener** | SetupMIListener.exe | Installs the MI Listener, which servers as a router between the Controller and the LoadRunner agent. For details, see "Set Up Your LoadRunner System Over Firewalls" on page 1136. | Dedicated machine |
| **Monitors Over Firewall** | SetupMoFW.exe | Installs the Monitors Over Firewall component, allowing you to monitor servers located over a firewall. For details, see "Set Up Your LoadRunner System Over Firewalls" on page 1136. | Dedicated machine |
| **TruClient Standalone** | SetupTruClient.exe | Installs TruClient as a standalone application. Install this to tool to record Web applications with TruClient technology. You save the recordings to a script that can be used in a LoadRunner test run. For details, see the TruClient Help Center (select the relevant version). | Any machine |
| **VuGen Standalone** | SetupVuGen.exe | Installs LoadRunner Virtual User Generator (VuGen) as a standalone application, allowing you to create scripts for a load test. For details, see "Introducing Virtual User Generator" on page 56. | Any machine |

# Function Reference

The Function Reference describes functions that can be used in Vuser scripts in several Micro Focus products. They can be used with supported protocols in scripts maintained in Virtual User Generator for use in load testing, application management (ALM), and functional testing (Unified Functional Testing). For information about applicability, refer to the product documentation.

To open the Function Reference from a machine with LoadRunner installed, from the Windows Start menu, go to **Micro Focus > LoadRunner > Documentation > Function Reference**.

# Send Us Feedback

Let us know how we can improve your experience with the User Guide.
Send your email to: docteam@microfocus.com