



Cisco Modeling Labs (CML)を 使ってネットワークを学ぼう！ (応用編)

シスコシステムズ合同会社
システムズエンジニア 齋藤 達也
2020/07/14

Cisco Modeling Labsを使ってネットワークを学ぼう！

- 2020年7月7日（火） Cisco Modeling Labs (CML)を使ってネットワークを学ぼう！（基礎編）
 - CML とは何か、ユースケースや基本機能、ラボ構築など、デモを交えながらご紹介します
- 2020年7月14日（火） Cisco Modeling Labs (CML)を使ってネットワークを学ぼう！（応用編）
 - CMLの様々な活用方法、資格試験での活用方法をデモを交えCCIEホルダーがご紹介します
- 2020年7月28日（火） Cisco Modeling Labs (CML)を使ってネットワークを学ぼう！（DevNet編）
 - CMLのAPI・プログラマビリティを用いたネットワーク自動化の観点でCMLをご紹介します
<https://learningnetwork.cisco.com/s/article/jp-webinar-cml03>

アジェンダ

1

前回のおさらい

2

仮想ノードのご紹介
新規仮想ノードの追加例

3

ネットワークシミュレーションツール
リモートからCML内ノードへの接続方法

4

外部アプリケーションと組み合わせた
CMLの活用

5

資格試験での活用方法

自己紹介



- 氏名
 - 齋藤 達也(さいとう たつや)
- 人材育成の一環でシスコにインターンシップとして出向(2017/02～2020/06)
- 出向元
 - 富士通ネットワークソリューションズ株式会社
- ひとこと
 - インターンシップ期間中は、ACIやCisco DNA製品の技術を主に勉強しました。
 - 2019年8月にCCIE Routing & Switchingを取得しました。
 - 最近嬉しかったことは、Jリーグが再開したことです。

アジェンダ

1

前回のおさらい

2

仮想ノードのご紹介
新規仮想ノードの追加例

3

ネットワークシミュレーションツール
リモートからCML内ノードへの接続方法

4

外部アプリケーションと組み合わせた
CMLの活用

5

資格試験での活用方法

Cisco Modeling Labs (CML)とは?

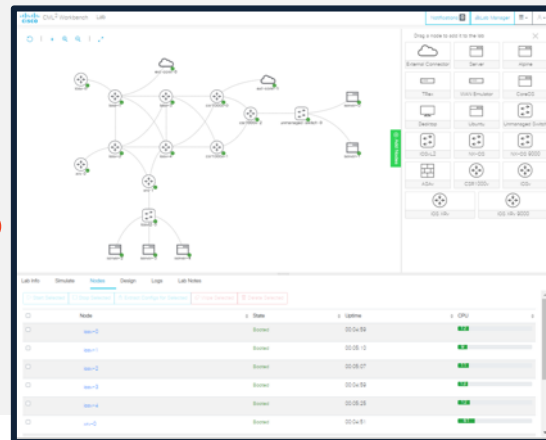
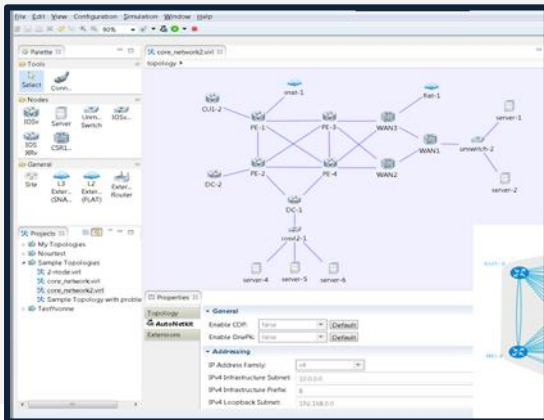
5月12日
CML2.0
アップデート!

Ciscoが提供する ネットワークシミュレーションプラットフォーム

- CML1.x
- VIRL



- CML-Enterprise 2.0
- CML-Personal 2.0



CML上で仮想化されるプラットフォーム

IOS	IOS XR	IOS XE	ASA	Nexus	Other
					<ul style="list-style-type: none">• TReX• WAN Emulator• Tiny Core Linux• Ubuntu 18.04• CoreOS• Alpine Linux
IOSv & IOSv L2	XRv & XRv 9000	CSR1000v	ASA v	NX-OS 7000v & 9000v	

※カスタムVMイメージ機能により新規イメージ、サードパーティ製イメージを追加可能

※ディスクイメージバージョン: refplat-20200409-fcs.iso

※各イメージのバージョン等の情報 <https://developer.cisco.com/docs/modeling-labs/#!reference-platforms-and-images>

ラボマネージャー

Cisco CML² Lab Manager 2.0.0-b13

List Add Lab Import Lab Tools

CPU / Load Avg

12.0%		
1 min	5 min	15 min
3.97	3.81	3.79

Memory [GB]

53.19%		
Total	Used	Free
31.25	16.62	14.23

Disk [GB]

27.79%		
Total	Used	Free
98.49	27.37	71.11

VM Stats

Allocated CPUs	Allocated Memory GB	Total VMs	Running VMs
20	22.26	97	19

Test by tatsaito 10

test by rmaruyam 8

To DNAC96 2 2

test by asai 14

メニューバー

- ラボの表示切替
- ラボ追加
- ラボのインポート
- ツールへのアクセスなど

システムステータス

- CPU使用率
- メモリ使用率
- ディスク使用率
- 合計仮想ノード数
- 実行中仮想ノード数など

ラボリスト

- ワークベンチへの移動
- ラボの実行
- ラボの停止
- ラボの初期化
- ラボの削除など

2つの基本画面

- **ラボマネージャ** : CMLサーバのステータス、ラボを管理するCMLのトップ画面
 - **ワークベンチ** : 作成したラボの編集、ネットワークシミュレーションを行うためのメイン画面
- ※**ラボ**: CMLでは1つのネットワークシミュレーションの単位をラボと呼びます

CML² Lab Manager **ラボマネージャ**

CPU / Load Avg

12.0		
1 min	5 min	15 min
3.97	3.81	3.79

Memory [GB]

53	19%	
Total	Used	Free
31.25	16.62	14.23

【ラボ】 10

Test by visit

Network diagram icon

Control icons: play, stop, edit, delete

【ラボ】 8

test by zeyuan

Network diagram icon

Control icons: play, stop, edit, delete

CML² Workbench Lab asai

Notifications

ワークベンチ

Drag a node to add it to the lab

- External Connector
- Server
- Alpine
- TRex
- WAN Emulator
- CoreOS
- Desktop
- Ubuntu
- Unmanaged Switch
- NX-OS

Network topology diagram with nodes: iosv-0, iosv-1, iosv-2, iosv-3, iosv-4, iosv2-0, csr1000-0, csr1000-1, csr1000-2, unmanaged-switch-0, server-0, server-1, server-2, server-3.

Lab Info Simulate Nodes Design Logs Lab Notes

19 Nodes 25 Links 75 Interfaces

Lab Description

ワークベンチ

The screenshot displays the Cisco CML2 Workbench interface for a lab named 'Lab asai'. The main area shows a network topology diagram with various nodes like 'iosv-0', 'iosv-1', 'iosv-2', 'iosv-3', 'iosv-4', 'iosv-5', 'iosv-6', 'iosv-7', 'iosv-8', 'iosv-9', 'iosv-10', 'iosv-11', 'iosv-12', 'iosv-13', 'iosv-14', 'iosv-15', 'iosv-16', 'iosv-17', 'iosv-18', 'iosv-19', 'iosv-20', 'iosv-21', 'iosv-22', 'iosv-23', 'iosv-24', 'iosv-25', 'iosv-26', 'iosv-27', 'iosv-28', 'iosv-29', 'iosv-30', 'iosv-31', 'iosv-32', 'iosv-33', 'iosv-34', 'iosv-35', 'iosv-36', 'iosv-37', 'iosv-38', 'iosv-39', 'iosv-40', 'iosv-41', 'iosv-42', 'iosv-43', 'iosv-44', 'iosv-45', 'iosv-46', 'iosv-47', 'iosv-48', 'iosv-49', 'iosv-50', 'iosv-51', 'iosv-52', 'iosv-53', 'iosv-54', 'iosv-55', 'iosv-56', 'iosv-57', 'iosv-58', 'iosv-59', 'iosv-60', 'iosv-61', 'iosv-62', 'iosv-63', 'iosv-64', 'iosv-65', 'iosv-66', 'iosv-67', 'iosv-68', 'iosv-69', 'iosv-70', 'iosv-71', 'iosv-72', 'iosv-73', 'iosv-74', 'iosv-75', 'iosv-76', 'iosv-77', 'iosv-78', 'iosv-79', 'iosv-80', 'iosv-81', 'iosv-82', 'iosv-83', 'iosv-84', 'iosv-85', 'iosv-86', 'iosv-87', 'iosv-88', 'iosv-89', 'iosv-90', 'iosv-91', 'iosv-92', 'iosv-93', 'iosv-94', 'iosv-95', 'iosv-96', 'iosv-97', 'iosv-98', 'iosv-99', 'iosv-100'. The diagram is overlaid with the text 'トポロジ作成 フィールド' (Topology Creation Field). To the right of the diagram is a panel titled 'Drag a node to add it to the lab' which lists various node types: External Connector, Server, Alpine, TRex, WAN Emulator, CoreOS, Desktop, Ubuntu, Unmanaged Switch, and NX-OS. Below the diagram is a 'Lab Info' panel showing '19 Nodes', '25 Links', and '75 Interfaces'. The interface also includes a 'Notifications' button, a 'Lab Manager' button, and a user profile icon.

メニューバー

- ラボのログ表示
- ラボマネージャへの移動
- ラボのエクスポート
- ラボの強制初期化など

ノード一覧

操作・情報表示パネル

- ラボ・ノードの情報
- シミュレーション・ノードの開始
- コンソールアクセス
- ラボのログ表示など

CML-Enterprise ,CML-Personal の比較

	Cisco Modeling Labs-Enterprise	Cisco Modeling Labs-Personal
対象ユーザ	企業・組織	個人
ユーザ数	マルチユーザ	シングルユーザ
ノード数(標準)	20ノード	20ノード
ノード数(最大)	300ノード	20ノード
サポート	Cisco TAC	Community Support Forum
購入方法	Cisco パートナー経由	Cisco Learning Network Store
ライセンスング	スマートライセンス	スマートライセンス
スマートアカウントの取得	必要	不要(Cisco.comアカウントがあればOK)
インターネット接続性	必須ではない ※オフラインでの認証も可能	必要 ※1度認証すれば30日間はオフラインでも使用可能
クラスタリング	ロードマップに有り	無し

※CML2.0.1 現在の仕様

アジェンダ

1

前回のおさらい

2

仮想ノードのご紹介
新規仮想ノードの追加例

3

ネットワークシミュレーションツール
リモートからCML内ノードへの接続方法

4





外部アプリケーションと組み合わせた
CMLの活用

5






資格試験での活用方法

CML仮想ノードのご紹介






CML仮想ノード スイッチ編

ノード	ノード名	説明	バージョン
 IOSvL2	IOSvL2	<ul style="list-style-type: none">エンタープライズ向けL2/L3スイッチクラシックIOS	IOS 15.2
 NX-OS	Nexus 7000v	<ul style="list-style-type: none">データセンター向けL2/L3スイッチ	NX OS 7.3.0.d1.1
 NX-OS 9000	Nexus 9000v	<ul style="list-style-type: none">データセンター向けL2/L3スイッチ	NX OS 9.2.3
 Unmanaged Switch	Unmanaged Switch	<ul style="list-style-type: none">コンソール機能非搭載のハブ	—

CML仮想ノード ルータ、ファイアウォール編

ノード	ノード名	説明	バージョン
 IOSv	IOSv	<ul style="list-style-type: none">エンタープライズ向けルータクラシックIOS	IOS 15.8(3)
 CSR1000v	CSR1000v	<ul style="list-style-type: none">クラウド向けルータ(Cloud Service Router)オープン&プログラマブルなIOS XE搭載	IOS XE 16.11.01b
 IOS XRv	IOS XRv	<ul style="list-style-type: none">サービスプロバイダー向けルータ (32bitイメージ)	IOS XR 6.3.1
 IOS XRv 9000	IOS XRv 9000	<ul style="list-style-type: none">サービスプロバイダー向けルータ (64bitイメージ)	IOS XR 6.3.1
 ASAv	ASAv	<ul style="list-style-type: none">ファイアウォールやVPN機能等を搭載した セキュリティ製品	9.12.2

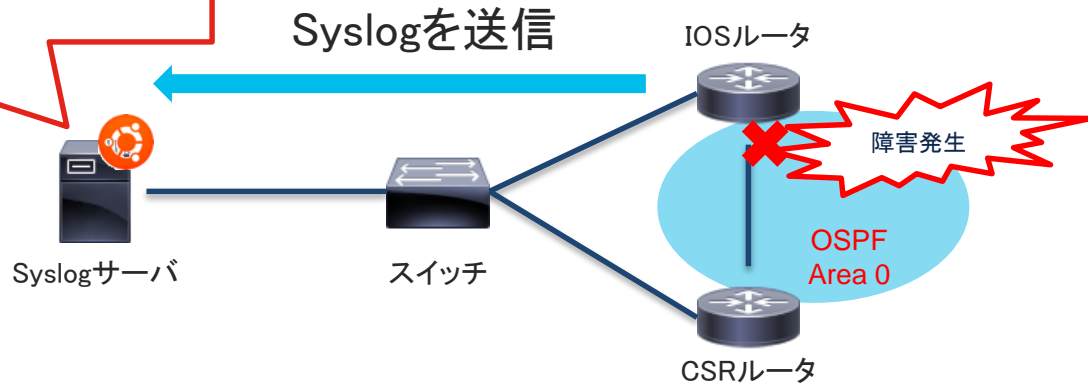
CML仮想ノード サーバ編

ノード	ノード名	説明	バージョン	クレデンシャル
 Server	Server	<ul style="list-style-type: none">• 最小限の機能のみ搭載のLinux• コンソール,VNC(CLIのみ)対応	Tiny Core Linux8.2.1	cisco/cisco
 Alpine	Alpine Linux	<ul style="list-style-type: none">• CLI機能のみ搭載の軽量なLinux• コンソール,VNC(CLIのみ)対応• パッケージマネージャは”apk”	Alpine Linux 3.10	cisco/cisco
 Desktop	Alpine Linux(GUI)	<ul style="list-style-type: none">• GUI機能を搭載したLinux• コンソール,VNC対応	Alpine Linux 3.10	cisco/cisco
 CoreOS	Core OS	<ul style="list-style-type: none">• コンテナホスト用の軽量なLinux• dockerコマンド搭載	Core OS 2135.4.0	cisco/cisco
 Ubuntu	Ubuntu 18.04	<ul style="list-style-type: none">• フル機能搭載のUbuntuサーバ• CLIのみ対応• パッケージ管理コマンド”apt”搭載	18.04.3 LTS	ubuntu/cisco

CML仮想ノード デモンストレーション

- UbuntuでSyslogサーバを構築
 - OSPFネイバーDown時にIOSルータからSyslogサーバにSyslogを送信

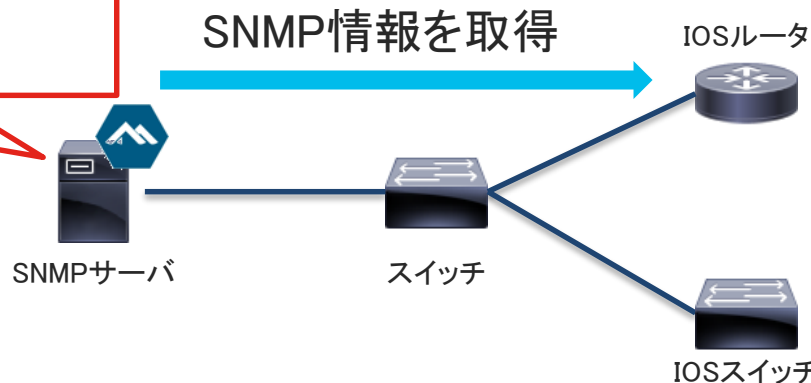
%OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on GigabitEthernet0/1 from FULL to DOWN, Neighbor Down: Dead timer expired



CML仮想ノード デモンストレーション

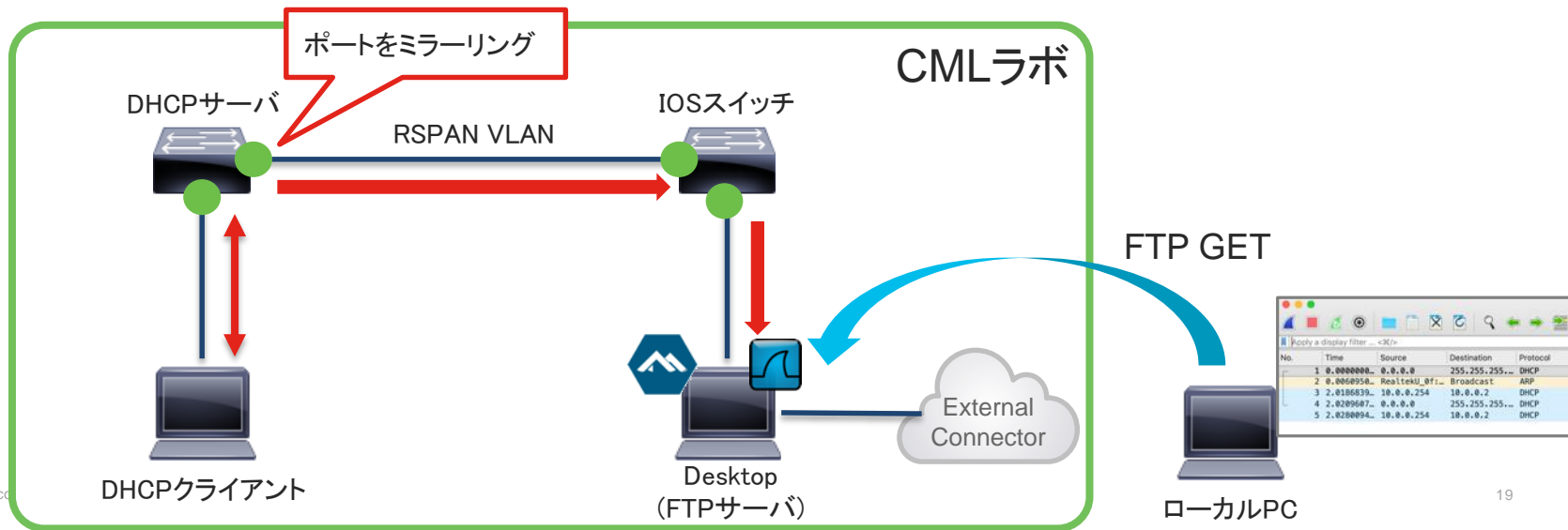
- Alpine Linuxでsnmpwalkを実行し、ルータ/スイッチのSNMP情報を取得
 - インターフェイス情報
 - ホスト名
 - SNMPエージェント起動からの経過時間など

```
alpine-base:~$ snmpwalk -v 2c -c "コミュニティ名" "デバイスIP" .1.3.6.1.2.1.2.2.1.2
IF-MIB::ifDescr.1 = STRING: GigabitEthernet0/0
IF-MIB::ifDescr.2 = STRING: GigabitEthernet0/1
IF-MIB::ifDescr.3 = STRING: GigabitEthernet0/2
IF-MIB::ifDescr.4 = STRING: GigabitEthernet0/3
```



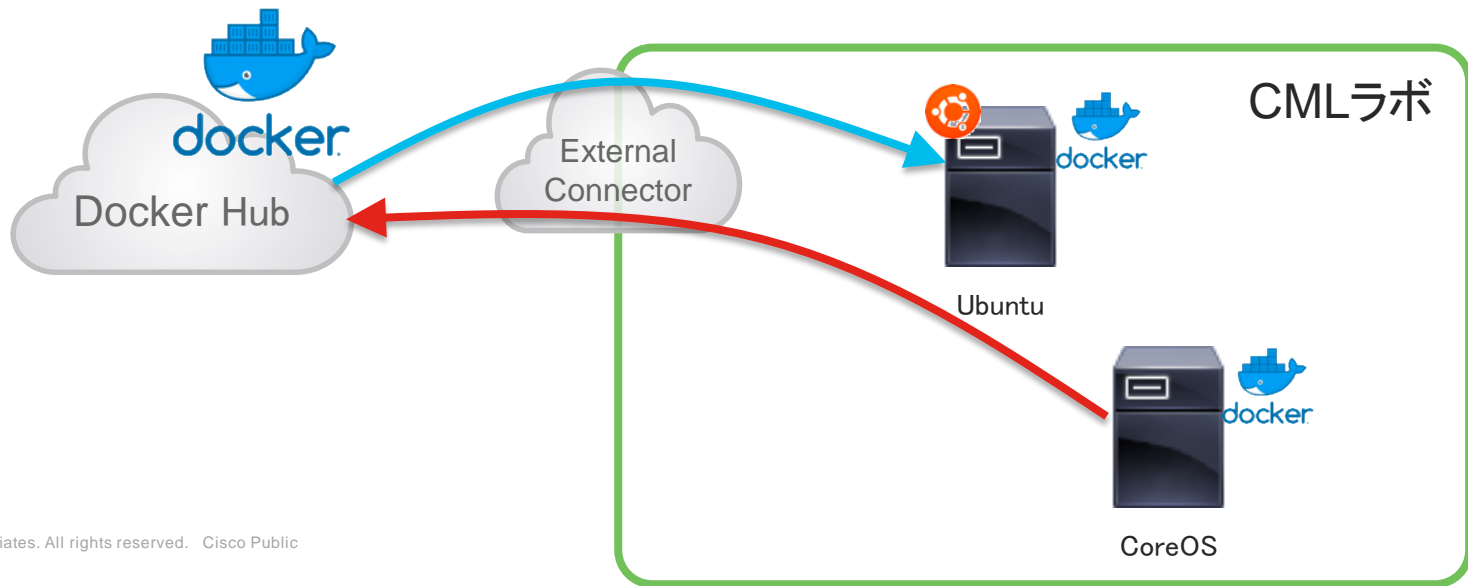
CML仮想ノード デモンストレーション

- Alpine DesktopのWireshark機能を使用したデモ
 - DHCPのやり取りをミラーリングし、DesktopのWiresharkでパケットキャプチャ
 - Desktopで取得したパケットキャプチャをローカルPCから取得






CML仮想ノード Docker Hubの活用

- Docker HubとCMLラボ内の連携
 - チームでリポジトリを管理・共有
 - Docker Hubへイメージをpush/pull



CML仮想ノード シミュレータ、その他編

ノード	ノード名	説明	バージョン
 WAN Emulator	WAN Emulator	<ul style="list-style-type: none">実際のWAN回線を擬似的に再現可能な装置	Alpine Linux 3.10
 TRex	TRex	<ul style="list-style-type: none">Ciscoが提供するオープンソースなトラフィックジェネレーター	Alpine Linux 3.10
 External Connector	External Connector	<ul style="list-style-type: none">CML内ノードを外部ネットワークに接続する際に使用	—

新規仮想ノードの追加例

新規仮想ノードの追加例

- 新規仮想ノードを追加する際、下記2つのコンポーネントが必要

YAML形式のノード構成ファイル

- CPU, RAM, ネットワークアダプターなどの情報
- GitHubのCML Communityで有志がサンプルをアップ(次ページ詳細)



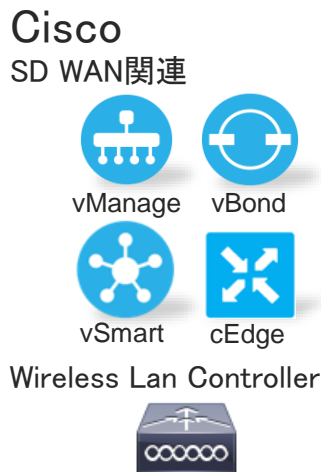
qcow2形式のディスクイメージ

- OSのイメージ
- 公式サイトよりダウンロードが必要



<参考>新規仮想ノードの追加例

- GitHubの下記参考リンクより有志がサンプルをアップ(7/14時点)
※qcow2イメージは各ベンダー公式サイトよりダウンロードが必要



Catalyst 9800

© 2020 Cisco and/or its affiliates. All rights reserved. Cisco Public

F5
アプリケーション
トラフィック管理製品



A10
アプリケーション
トラフィック管理製品



Fortinet
次世代ファイアウォール



MikroTik
クラウドルータ



<参考>新規仮想ノードの追加例 A10 vThunder編

- 手順概要(各種サイトにてダウンロード)
 1. A10の公式サイトにて**ディスクイメージ(qcow)**をダウンロード
 2. GitHubにて**ノード構成ファイル(YAML)**をダウンロード
- 手順概要(CML内の作業)
 3. CMLにてダウンロードした**ノード構成ファイル(YAML)**をインポート
 4. CMLにてダウンロードした**ディスクイメージ(qcow)**をアップロード
 5. CMLにてインポートした**定義ファイル(YAML)**と**ディスクイメージ(qcow)**を紐付け

<参考>新規仮想ノードの追加例 A10 vThunder編

1. A10の公式サイトにてディスクイメージ(qcow)をダウンロード
・ アカウントを作成し、30日間のフリートライアルライセンスを取得



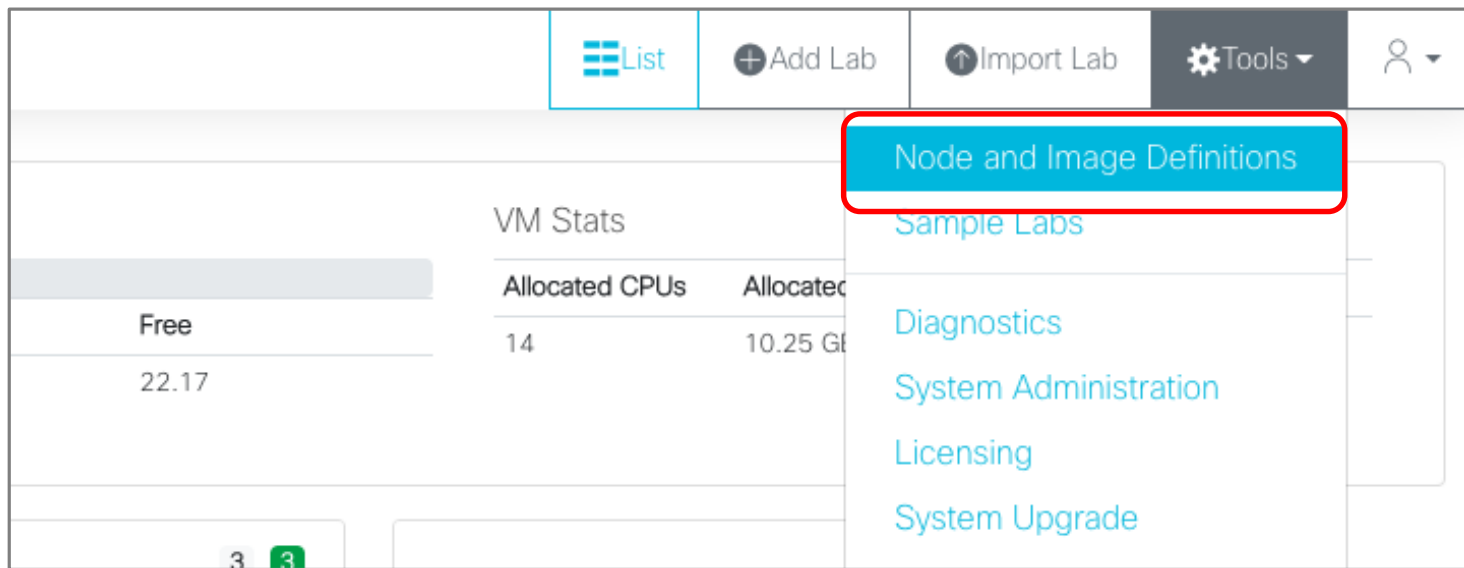
<参考>新規仮想ノードの追加例 A10 vThunder編

2. GitHubにてノード構成ファイル(YAML)をダウンロード
 - vThunderは**CPU4コア、メモリ4GB**必要

The screenshot shows a GitHub repository page for the branch 'master' in the path 'cml-community / node-definitions / a10 / vThunder /'. A commit by 'xorkkaz' is shown, with a file 'a10vthunder.yaml' highlighted by a red box. Below the commit, the README.md content is displayed, with the 'Description' section highlighted by a red box. The description states: 'This node definition provides for a three-interface node (one management interface, two ethernet interfaces) requiring 4 vCPUs and 4 GB of RAM.'

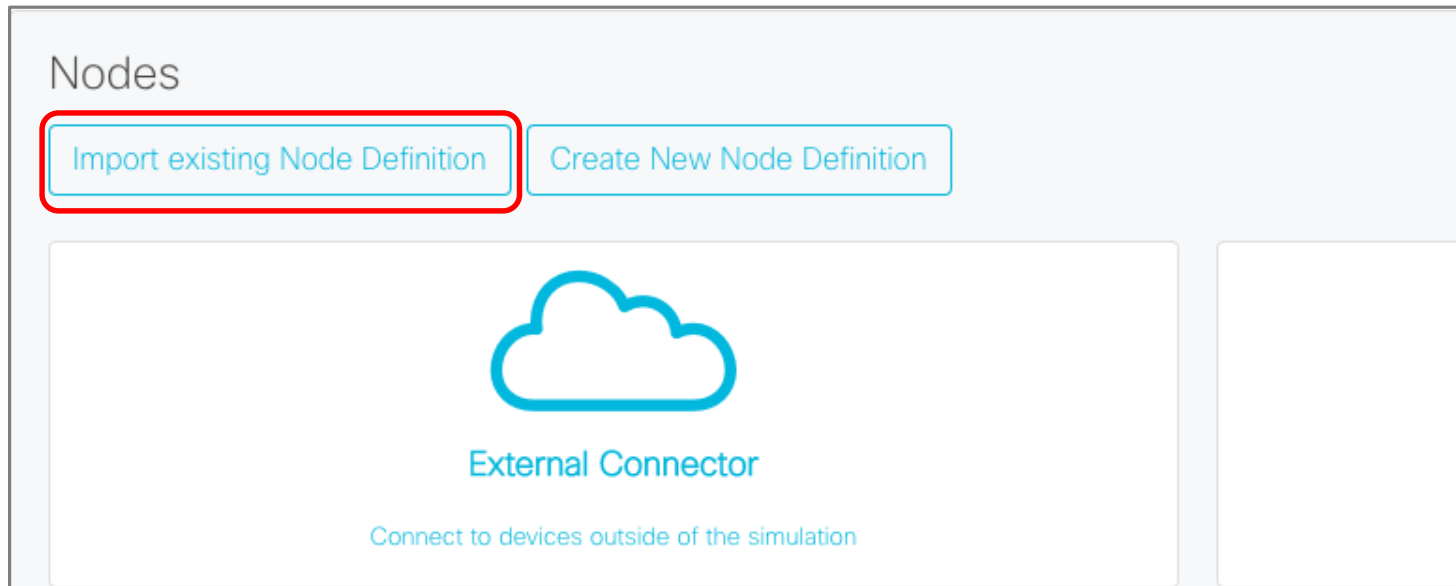
<参考>新規仮想ノードの追加例 A10 vThunder編

3. CMLにてダウンロードしたノード構成ファイル(YAML)をインポート
 - CMLの”Tools” > “Node and Image Definitions”をクリック



<参考>新規仮想ノードの追加例 A10 vThunder編

3. CMLにてダウンロードしたノード構成ファイル(YAML)をインポート
 - “Nodes”配下の”Import existing Node Definition”をクリック



<参考>新規仮想ノードの追加例 A10 vThunder編

3. CMLにてダウンロードしたノード構成ファイル(YAML)をインポート
 - GitHubにてダウンロードしたYAMLファイルをインポートし、“Browse”ボタンを押すと作成したノードが反映される

Adding a Node Definition

This feature is to add a new Node Definition, such as a new device type. If you are wanting to add a new image such as a .bin or a .qcow2, please use the add Image Definition. If in doubt, it's most likely that you wish to add a new Image Definition.

Import Definition

Imported Node Definition a10vthunder

Nodes

External Connector

Connect to devices outside of the simulation



CoreOS

CoreOS Container Linux



NX-OS

Cisco NX-OSv "Titanium" Switch



Server

Tiny Core Linux



CSR1000v

Cloud Services Router 1000V



NX-OS 9000

Cisco Nexus 9000v Switch



Unmanaged Switch

Unmanaged Switch



Desktop

Alpine Linux with X11 Desktop



Ubuntu

Ubuntu 18.04 Cloud Init Platform



Alpine

Alpine Linux



IOSv

Cisco IOSv Virtual Router Platform

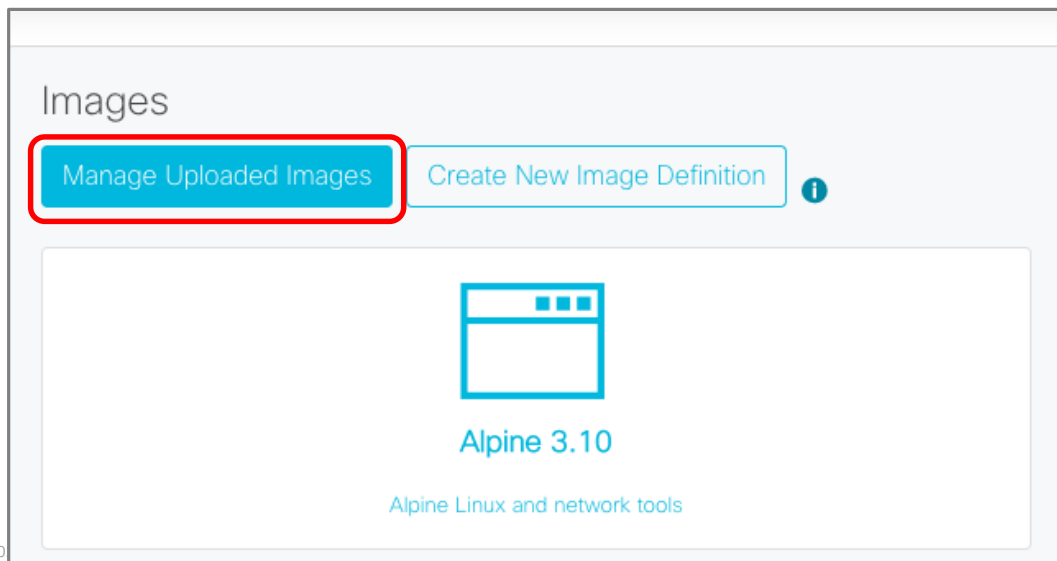


A10 vThunder

A10 vThunder Virtual Appliance

<参考>新規仮想ノードの追加例 A10 vThunder編

4. CMLにてダウンロードしたディスクイメージ(qcow)をアップロード
 - “Node and Image Definitions” > “Images” > “Manage Uploaded Images”をクリックし、qcowイメージをアップロード



<参考>新規仮想ノードの追加例 A10 vThunder編

4. CMLにてダウンロードしたディスクイメージ(qcow)をアップロード
 - qcowイメージをCMLにアップロード

Upload New Image File

Images must be uploaded in .qcow or .qcow2 format. If your image is in a different format, please convert it first before uploading.

ACOS_vThunder_4_1_4-GR1-P2_151.qcow2 Browse

Upload Image

Upload New Image File

Images must be uploaded in .qcow or .qcow2 format. If your image is in a different format, please convert it first before uploading.

Choose a file...

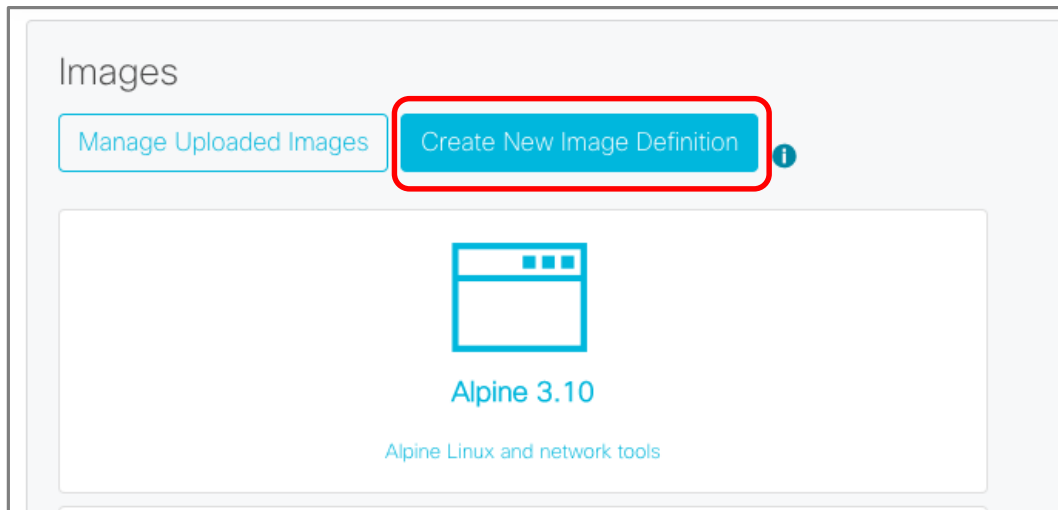
Uploaded Images

Refresh

ACOS_vThunder_4_1_4-GR1-P2_151.qcow2

<参考>新規仮想ノードの追加例 A10 vThunder編

5. CMLにてインポートした定義ファイル(YAML)とディスクイメージ(qcow)を紐付け
 - “Node and Image Definitions” > “Images” > “Create New Image Definition” をクリック



<参考>新規仮想ノードの追加例 A10 vThunder編

5. CMLにてインポートした定義ファイル(YAML)とディスクイメージ(qcow)を紐付け
 - GitHub記載の通り**CPU4コア、メモリ4GB**を入力

Image Definition

General

ID: A10-vThunder-4.1.4 ✓

Label: vThunder 4.1.4 ✓

Description:

Disk Image: ACOS_vThunder_4_1_4-GR1-P2_151.qcow2 ✓

Refresh Available Disk Images | Manage Image Uploads

Node Definition: A10 vThunder ✓

Linux Native Simulation

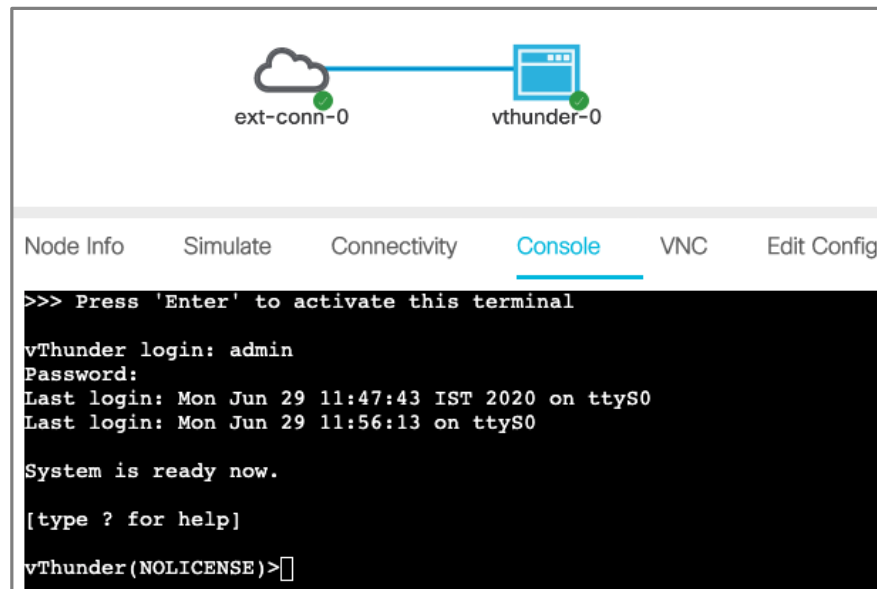
Memory (MB): 4096 ✓

CPUs: 4 ✓

Create Image Definition

<参考>新規仮想ノードの追加例 A10 vThunder編

- vThunderを起動し、コンソールログイン
(ユーザ名:admin、パスワード:a10)
※管理インターフェイスの設定、ライセンスの適用方法は今回省略



アジェンダ

1

前回のおさらい

2

仮想ノードのご紹介
新規仮想ノードの追加例

3

ネットワークシミュレーションツール
リモートからCML内ノードへの接続方法

4

外部アプリケーションと組み合わせた
CMLの活用

5

資格試験での活用方法

ネットワークシミュレーション

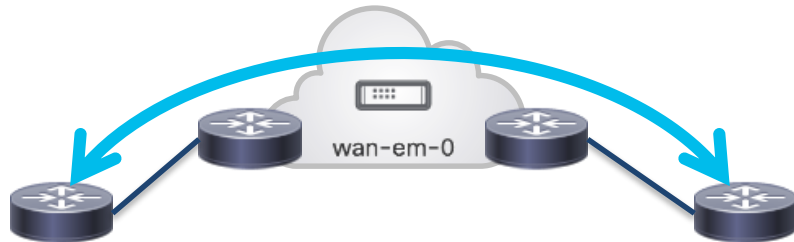
1. WANエミュレータ
2. トラフィックジェネレーター

WANエミュレータ

- **帯域や遅延、パケットロス**等のパラメータを変更し、実際のWAN回線を筐体内で再現するノード

例)

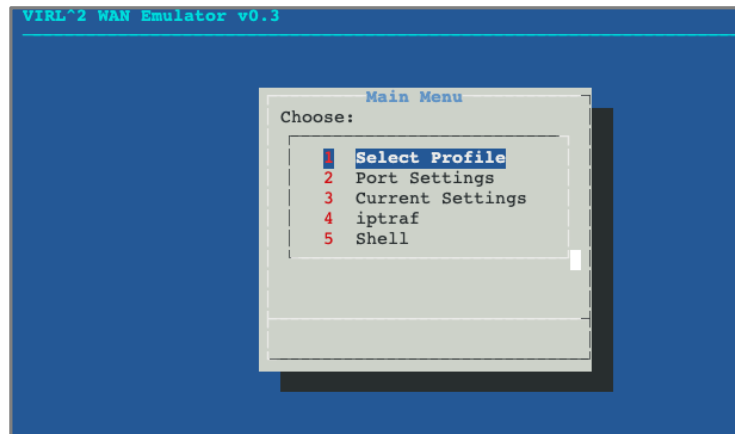
- 品質の良い/悪いWAN回線のネットワークシミュレーション
- アプリケーション動作のシミュレーション
- SD-WAN検証など



WANエミュレータ

- メインメニューのご紹介

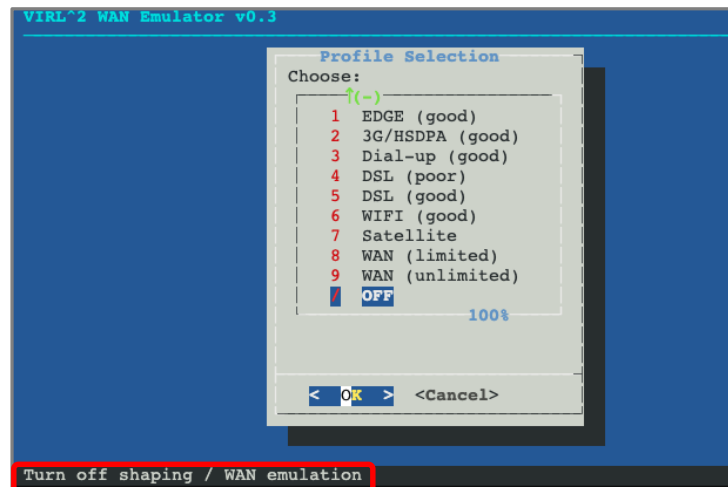
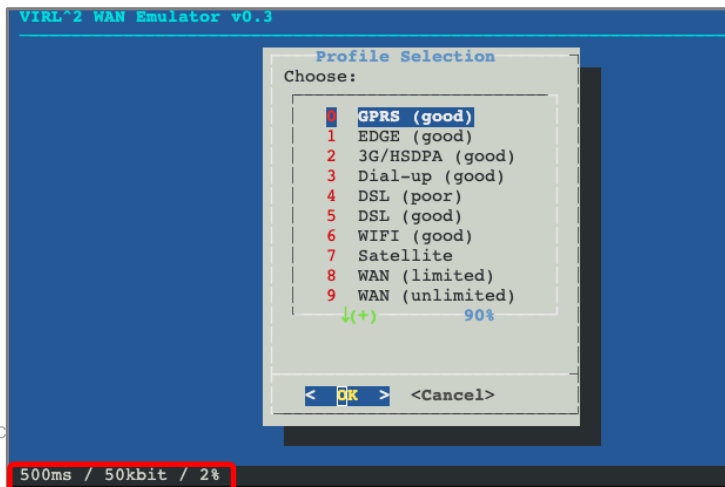
1. Select Profile: 予め10種類のプロファイルが定義
2. Port Settings: 手動で“遅延/ジッタ/パケットロス“を設定
3. Current Settings: 現在の設定の確認
4. iptraf: WANエミュレータのインターフェイスの統計が表示
5. Shell: シェル画面の表示



<参考>WANエミュレータ

1. Select Profile

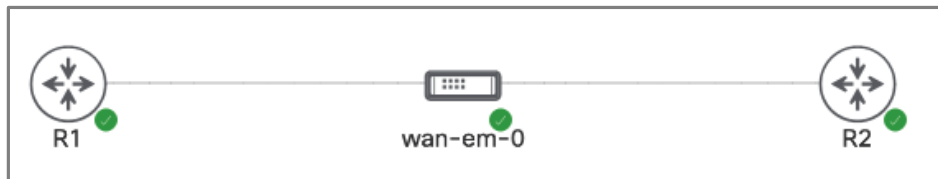
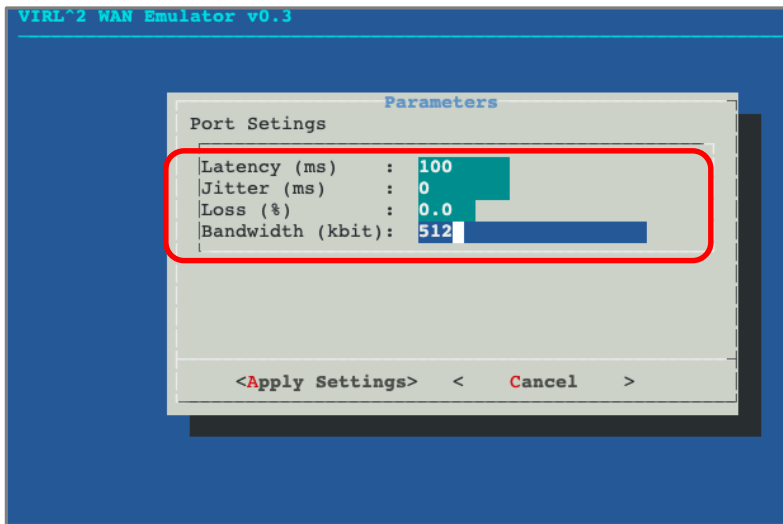
- 画面左下にそのプロファイルの**遅延/帯域/パケットロス**が表示
- 10番目の**"OFF"プロファイル**は遅延などの制御を行わない**品質のよい回線を再現**
- **ジッタを挿入できない**、プロファイルを追加、変更、削除はできない



<参考>WANエミュレータ

2. Port Settings

- 初期値は以下の仕様
 - 遅延: 100ms, ジッタ: 0ms, パケットロス: 0.0%, 帯域: 512kbit
- 初期値のまま、R1からR2へping100回実施



```
R1#ping 10.0.0.2 repeat 100
Type escape sequence to abort.
Sending 100, 100-byte ICMP Echos to 10.0.0.2, timeout is 2 seconds:
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
Success rate is 100 percent (100/100), round-trip min/avg/max = 198/203/207 ms
```

WANエミュレータでは**双方向で制御**されるため、
結果として**約200msの遅延**が発生

<参考>WANエミュレータ

3. Current Settings

- 現在の設定をポート単位で確認

```
VIRL^2 WAN Emulator v0.3

eth0 settings:
=====
qdisc htb 10: root refcnt 2 r2q 10 default 0x10 direct packets stat 0 dire
qdisc netem 100: parent 10:10 limit 1000 delay 100.0ms 50.0ms loss 2%
class htb 10:10 root leaf 100: prio 0 rate 512Kbit cell 512Kbit burst 1600
class htb 10:1 root prio 0 rate 10Gbit ceil 10Gbit burst 0b cburst 0b

eth1 settings:
=====
qdisc htb 10: root refcnt 2 r2q 10 default 0x10 direct packets stat 0 dire
qdisc netem 100: parent 10:10 limit 1000 delay 100.0ms 50.0ms loss 2%
class htb 10:10 root leaf 100: prio 0 rate 512Kbit cell 512Kbit burst 1600
class htb 10:1 root prio 0 rate 10Gbit ceil 10Gbit burst 0b cburst 0b

< EXIT >
```

4. iptraf

- インターフェイスの統計を表示

```
iptraf-ng 1.1.4
```

iface	Total	IPv4	IPv6	NonIP	BadIP	Activity
lo	0	0	0	0	0	0.00 kbps
eth0	116	116	0	0	0	6.15 kbps
eth1	116	116	0	0	0	5.99 kbps
wan0	0	0	0	0	0	0.00 kbps

5. Shell

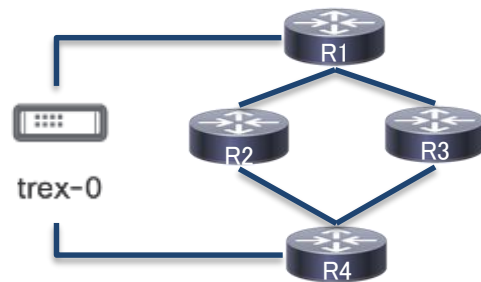
- シェル画面の起動

```
alpine-wanem:/# cat /etc/issue
Welcome to Alpine Linux 3.10
Kernel \r on an \m (\l)

alpine-wanem:/#
alpine-wanem:/#
```

トラフィックジェネレーター TRex

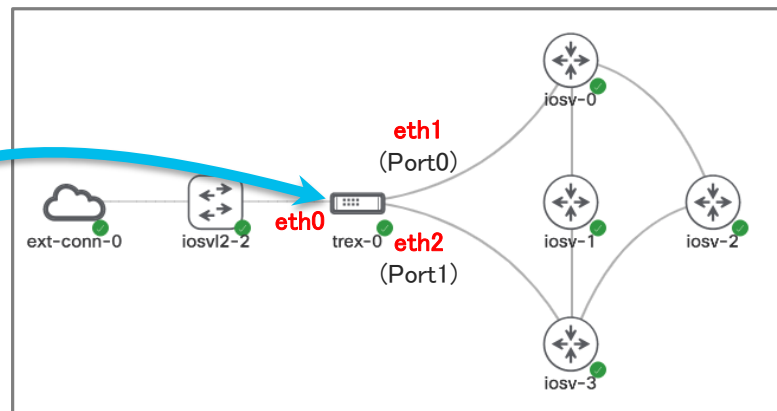
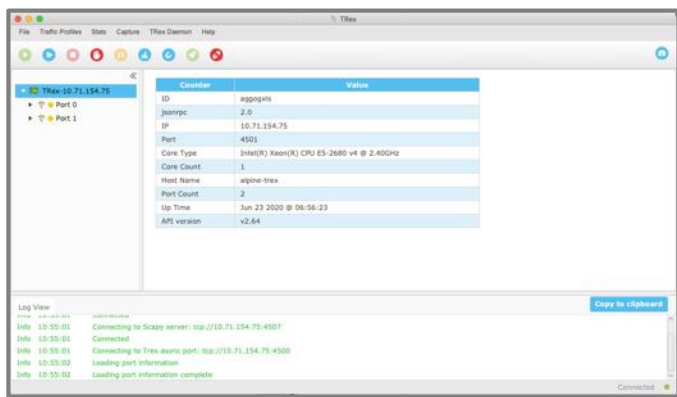
- Ciscoが提供する**オープンソース**なトラフィックジェネレーター
- CML提供のTRexはGUIアプリケーションで管理
- 大量のトラフィックを生成でき、パケットの細かいパラメータも設定可能
 - MACアドレスやIPv4のパラメータ、TCP/UDPのポート番号を設定
 - 連続的なパケットや一度に大量のパケットの生成
- ネットワークフィルタリング検証や、アプリケーション性能検証が可能
 - アクセスリストや、ルートマップのフィルタリング



トラフィックジェネレーター TRex

1. TRexの概要

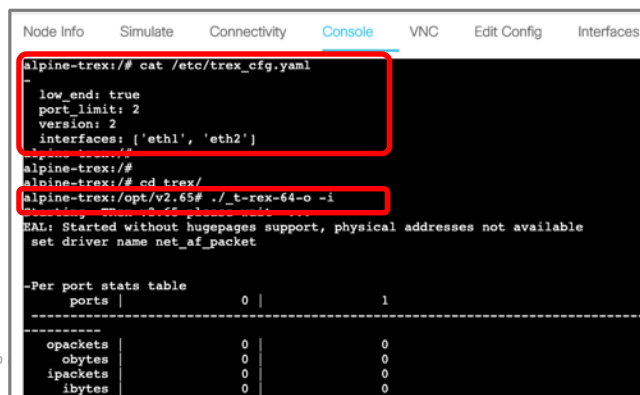
- TRexには最低2つのインターフェイスが必要でGUIアプリで管理
 - **“eth0”** → GUI管理インターフェイス
 - Windows,Mac等にクライアントソフトをインストールし&アクセスするため、外部接続が必要
 - **”eth1” or “eth2”** → テスト用インターフェイス



トラフィックジェネレーター TRex

2. TRexにアクセスするまでの手順

- TRex起動時、**設定ファイルの不備等があるため**ワークアラウンドとして下記設定が事前に必要
 - etc配下の**”trex_cfg.yaml”**ファイルの**”interfaces”**を[**’eth1’**, **’eth2’**]に編集
 - **”eth1”**,**”eth2”**はテスト用インターフェイス
 - trexディレクトリに移動し、**”./_t-rex-64-o -i”**コマンドでTRexサーバを手動起動



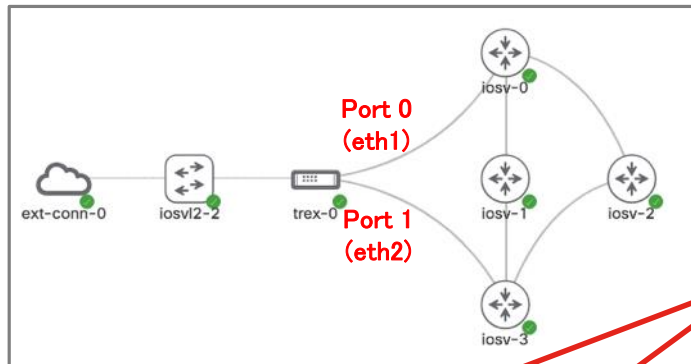
The screenshot shows a terminal window with the following content:

```
alpine-trex:/# cat /etc/trex_cfg.yaml
-
  low_end: true
  port_limit: 2
  version: 2
  interfaces: ['eth1', 'eth2']
alpine-trex:/#
alpine-trex:/# cd trex/
alpine-trex:/opt/v2.65# ./_t-rex-64-o -i
EAL: Started without hugepages support, physical addresses not available
set driver name net_af_packet

-Per port stats table
  ports |          0 |          1
-----|-----|-----
opackets |          0 |          0
obytes   |          0 |          0
ipackets |          0 |          0
ibytes   |          0 |          0
```

<参考>トラフィックジェネレーター TRex

3. TRexの主要なページのご紹介



TRexサーバの情報

Counter	Value
ID	aggoxls
jsonrpc	2.0
IP	10.71.154.75
Port	4501
Core Type	Intel(R) Xeon(R) CPU E5-2680 v4 @ 2.40GHz
Core Count	1
Host Name	alpine-trex
Port Count	2
Up Time	Jul 02 2020 @ 05:51:03
API version	v2.64

TRexサーバのログ情報

```
Log View
Info 10:36:59 Connecting to Trex server: tcp://10.71.154.75:4501
Info 10:36:59 Connected
Info 10:36:59 Connecting to Scapy server: tcp://10.71.154.75:4507
Info 10:36:59 Connected
Info 10:36:59 Connecting to Trex async port: tcp://10.71.154.75:4500
Info 10:37:00 Loading port information
Info 10:37:00 Loading port information complete
```

Copy to clipboard

ポート情報

port0 ⇒ eth1(テスト用インターフェイス)

port1 ⇒ eth2(テスト用インターフェイス)

Profile ⇒ トラフィック生成プロファイルを定義

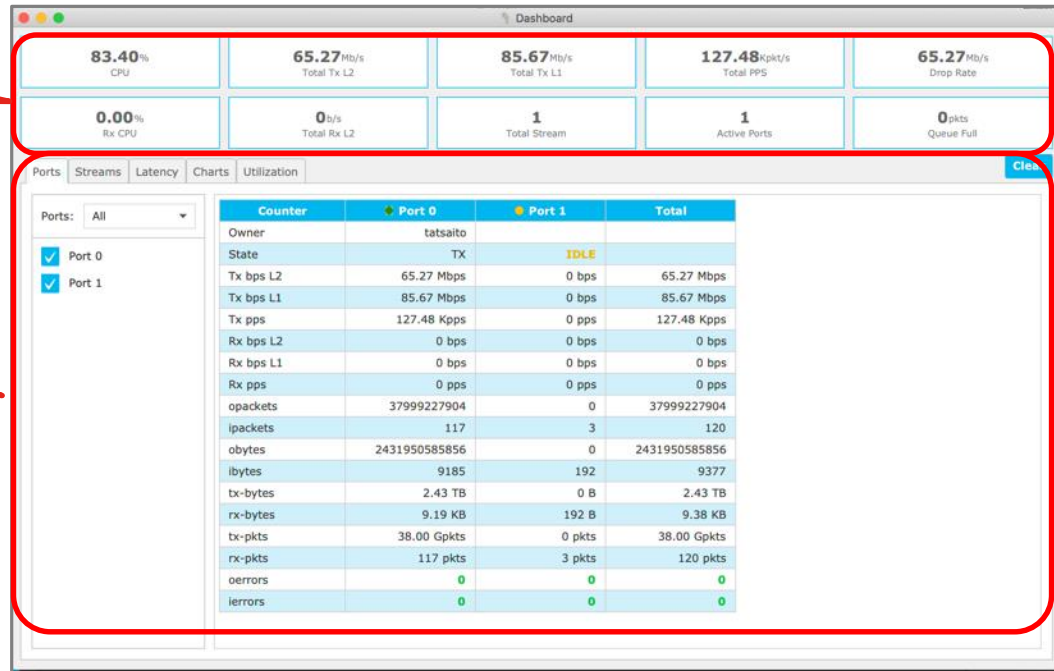
<参考>トラフィックジェネレーター TRex

3. TRexの主要なページのご紹介

- “Stats” > “Dashboard”でport単位のトラフィック情報が表示

CPU使用率や送信受信の統計等が表示

Portの詳細情報の表示
※Port0のみトラフィックを発生中



<参考>トラフィックジェネレーター TRex

3. TRexの主要なページのご紹介

• Portの設定

- Promiscuous: 自分宛て以外のパケットをすべて処理(管理・監視目的で使用)
- Service: パケットキャプチャ機能を有効化 ※次ページで解説

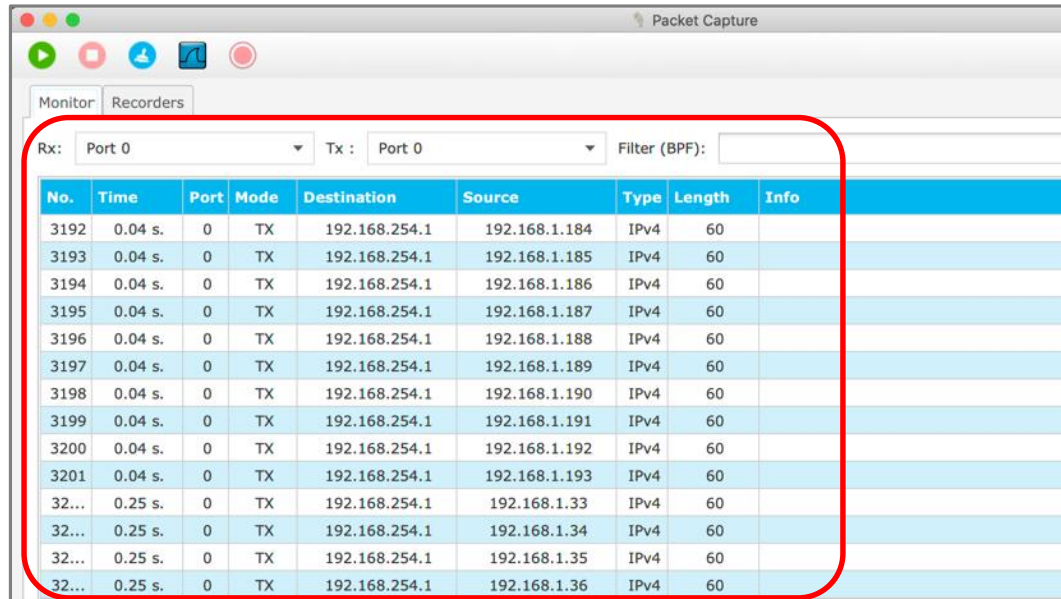
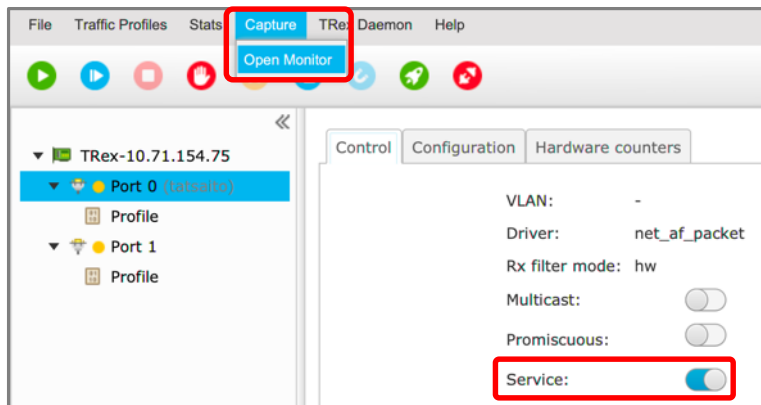
The screenshot shows the TRex web interface for a device at IP 10.71.154.75. The left sidebar shows a tree view with 'Port 0' selected. The main content area has three tabs: 'Control', 'Configuration', and 'Hardware counters'. The 'Configuration' tab is active, displaying a grid of settings for 'Port 0'. The settings include: VLAN: -, Driver: net_af_packet, Rx filter mode: hw, Multicast: (toggle off), Promiscuous: (toggle off), Service: (toggle off), Owner: -, Speed: 10 Gb/s, Status: STREAMS, Capturing: (toggle off), Link: (toggle on), LED: (toggle off), NUMA node: -1, PCI address: N/A, RX Queueing: Off, Grar ARP: 120 second(s), and Flow control: NONE. At the bottom of the configuration area, there are three buttons: 'Acquire', 'Force Acquire', and 'Reset'. A red box highlights the 'Acquire' button, and a callout box with a red border points to it, containing the text '“Acquire”を押すと編集可'.

“Acquire”を押すと編集可

<参考>トラフィックジェネレーター TRex

3. TRexの主要なページのご紹介

- Serviceにチェックを入れることで”Capture” > “Open Monitor”より、Port単位でパケットキャプチャを取得可能(図は”Port 0”のキャプチャ)



<参考>トラフィックジェネレーター TReX

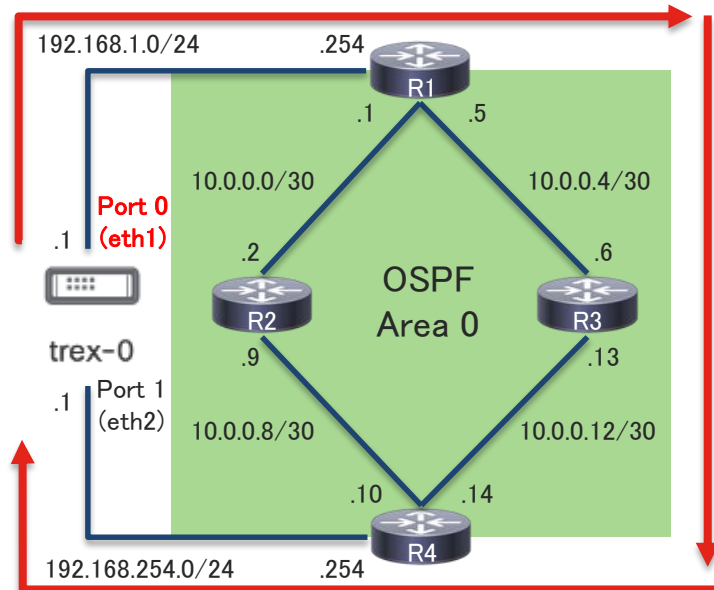
4. 実際にL3トラフィックを発生させる手順

- テスト用インターフェイスのIPアドレスの設定

SourceはTReXの”Port 0”に設定したいIP、Destinationは到達性のあるIPを指定し、”Apply”をクリック(今回DestはR1の.254) ”ARP status”が”resolved”になればOK

R1にてTReXのPort 0(192.168.1.1)のARP解決が確認

```
R1#sh ip arp
Protocol Address      Age (min)  Hardware Addr  Type         Interface
-----
Internet 10.0.0.1         -          5254.0011.565e ARPA         GigabitEthernet0/1
Internet 10.0.0.2         135       5254.0015.1ccb ARPA         GigabitEthernet0/1
Internet 10.0.0.5         -          5254.001e.cd5d ARPA         GigabitEthernet0/2
Internet 10.0.0.6         121       5254.0019.b082 ARPA         GigabitEthernet0/2
Internet 192.168.1.1     1          5254.0008.72e1 ARPA         GigabitEthernet0/0
Internet 192.168.1.254  -          5254.001e.6b99 ARPA         GigabitEthernet0/0
```



<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

- ・ プロファイルの作成、ストリームの作成

The screenshot shows the TRex GUI with two callouts. The first callout points to the '+ New Profile' button, and the second callout points to the '+ Build Stream' button and the resulting stream table.

“New Profile”をクリックし、
プロファイルを作成

“Build Stream”をクリックし、
ストリームを作成
※既に“Test”という名前で作成済み
ストリームにて細かなパラメータを設定

Index	Name	Packet Type	Length	Mode	Rate	Next Stream	
<input checked="" type="checkbox"/>	1	Test	Ethernet/IPV4	64	Multi_burst	1 pps	-

<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

- ストリーム編集モードのご紹介

Stream Properties | Protocol Selection | Protocol Data | Advanced Settings | Packet viewer

Mode

Continuous
 Burst
 Multi-Burst

Misc

Enabled
 Self start

Numbers

Number of Packets: 15
Number of Burst: 1
Packets per Burst: 10

Rate

pps | 1.0

After this stream

Stop
 Goto Stream
 Time in loop: 0

Gaps (in seconds)

ISG: 0.0 | IBG: 3 | IPG:

RX Stats

Enabled | PG ID: 0 | Latency enabled

Advanced mode

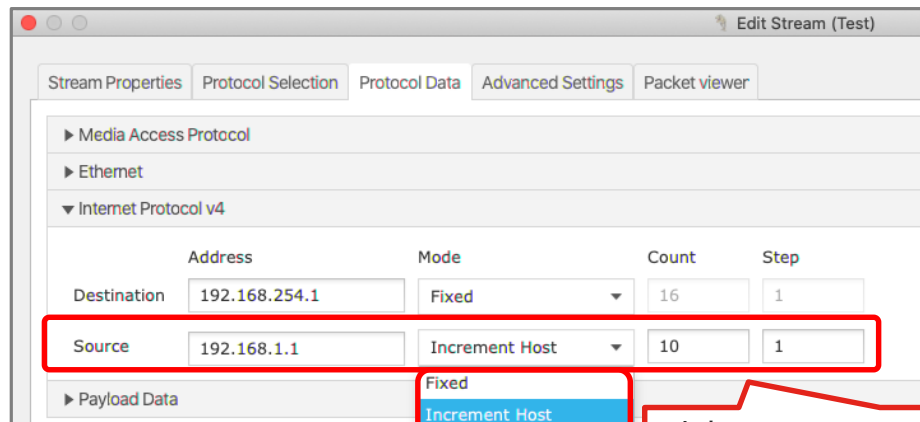
Save | Cancel

“Simple”モードと“Advanced”モードの2種類
※今回は“Simple”モードを使用

<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

• Protocol Dataの編集(IPv4)



モード

- Fixed → 固定
- Increment Host → ホストの増加
- Decrement Host → ホストの減少
- Random Host → ランダムホスト

Fixed
Increment Host
Decrement Host
Random Host

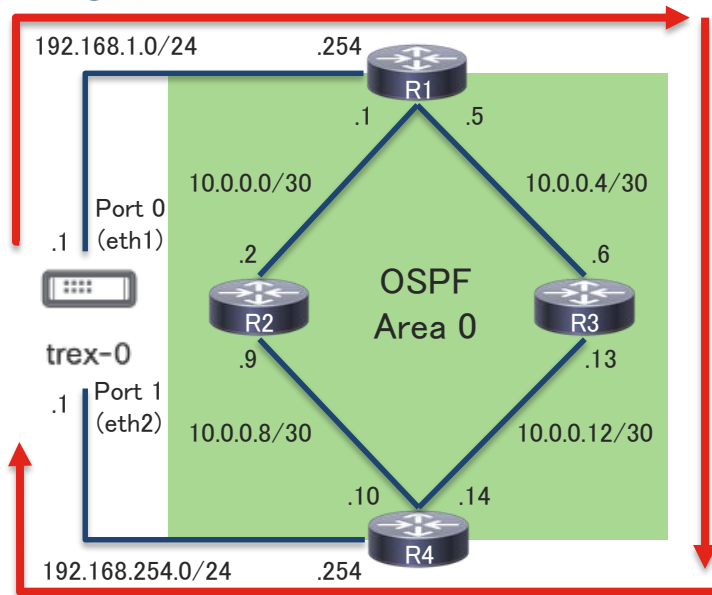
“カウント 10、ステップ 1”の場合

192.168.1.1 (カウント1)
192.168.1.2 (カウント2)
⋮
192.168.1.10 (カウント10)
192.168.1.1 (カウント1)※以後繰返
⋮



<参考>“カウント 10、ステップ 2”の場合

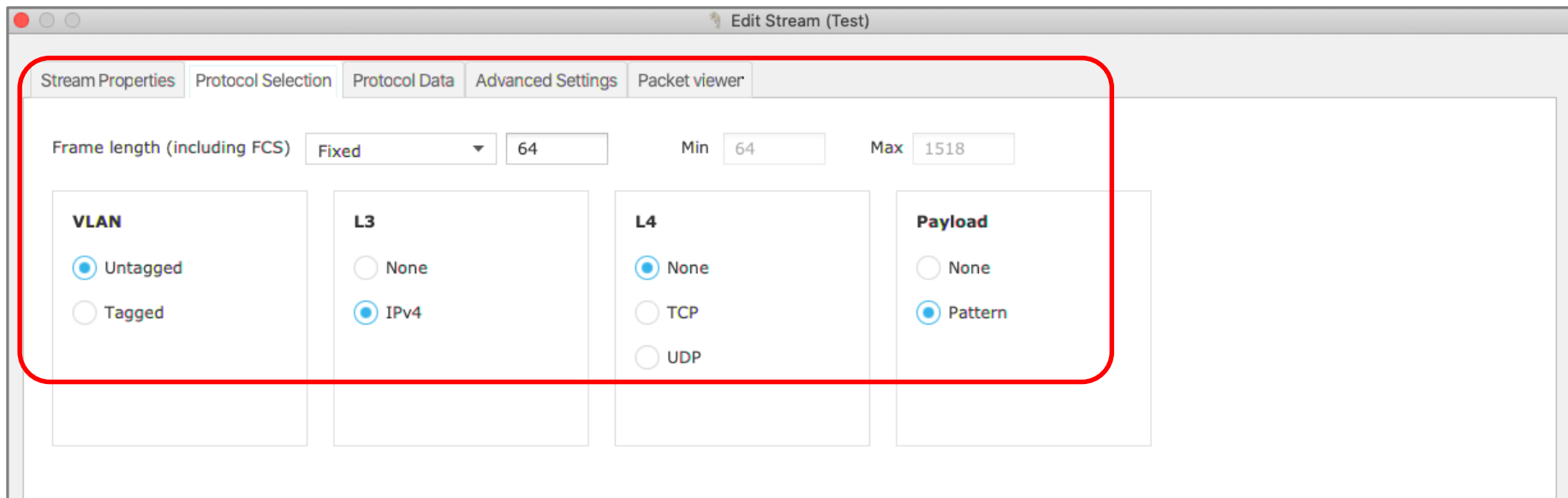
192.168.1.1 (カウント1)
192.168.1.3 (カウント2)
192.168.1.5 (カウント3)
⋮
192.168.1.19 (カウント10)



<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

- Protocol Sessionの設定
 - パケットの長さや各レイヤーの設定が可能



<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

- Stream Propertiesにて、パケットの発生条件や間隔を主に設定

The screenshot shows the TRex configuration interface with several sections highlighted by red boxes and callouts:

- Mode:** A red box highlights the 'Mode' section where 'Continuous' is selected. A callout points to this section.
- Misc:** A red box highlights the 'Misc' section where 'Enabled' and 'Self start' are checked.
- Numbers:** A red box highlights the 'Numbers' section where 'Number of Packets', 'Number of Burst', and 'Packets per Burst' are all set to 1.
- Rate:** A red box highlights the 'Rate' section where the unit is 'pps' and the value is '1.0'.
- After this stream:** A red box highlights the 'After this stream' section where 'Stop' is selected.
- Gaps(in seconds):** A red box highlights the 'Gaps(in seconds)' section. A callout explains: 'Mode、Gapsにてパケットの発生条件、間隔を設定可能 ※詳細は次ページ'. Below this, 'ISG' is set to 0.0, 'IBG' is set to 0.0, and 'IPG' is empty.
- RX Stats:** A red box highlights the 'RX Stats' section where 'Enabled' and 'Latency enabled' are checked. A callout explains: 'RX Statsを有効化することでダッシュボードのチャートビューが表示'. To the right, a smaller screenshot shows a dashboard with a line chart for 'PG ID - 0'.

<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

- Mode、Gapsの詳細 (Mode="Continuous"の場合)

シミュレーションをストップしない限り、
絶えず継続的にパケットを発生

The screenshot shows the TRex configuration interface with the following settings:

- Mode:** Continuous (selected), Burst, Multi-Burst.
- Misc:** Enabled (checked), Self start (checked).
- Numbers:** Number of Packets: 1, Number of Burst: 1, Packets per Burst: 1.
- Gaps (in seconds):** ISG: 3.0, IBG: 0.0, IPG: [empty].

A diagram below the Gaps section illustrates the timing: ISG (Inter-Segment Gap) is shown as an orange arrow, PKT1 (Packet 1) as a blue arrow, IPG (Inter-Packet Gap) as a green arrow, and PKT2 (Packet 2) as a blue arrow. The ISG value is set to 3.0 seconds.

ISG=シミュレーション開始後、
何秒後にパケットを発生させるか設定
※画像の場合3秒後にパケット発生

シミュレーションをストップしない限り、
絶えず継続的にパケットを発生

The screenshot shows the packet capture output table with the following data:

No.	Time	Port	Mode	Destination	Source	Type	Length
1	1 s.	0	TX	192.168.254.1	192.168.1.1	IPv4	60
2	1.99 s.	0	TX	192.168.254.1	192.168.1.2	IPv4	60
3	2.99 s.	0	TX	192.168.254.1	192.168.1.3	IPv4	60
4	3.99 s.	0	TX	192.168.254.1	192.168.1.4	IPv4	60
5	4.99 s.	0	TX	192.168.254.1	192.168.1.5	IPv4	60
6	5.99 s.	0	TX	192.168.254.1	192.168.1.6	IPv4	60
7	6.99 s.	0	TX	192.168.254.1	192.168.1.7	IPv4	60
8	7.99 s.	0	TX	192.168.254.1	192.168.1.8	IPv4	60
9	8.58 s.	0	TX	ff:ff:ff:ff:ff:ff	52:54:00:08:72:e1	ARP	60
10	8.99 s.	0	TX	192.168.254.1	192.168.1.9	IPv4	60
11	9.99 s.	0	TX	192.168.254.1	192.168.1.10	IPv4	60
12	10.99 s.	0	TX	192.168.254.1	192.168.1.1	IPv4	60
13	11.99 s.	0	TX	192.168.254.1	192.168.1.2	IPv4	60
14	12.99 s.	0	TX	192.168.254.1	192.168.1.3	IPv4	60
15	13.99 s.	0	TX	192.168.254.1	192.168.1.4	IPv4	60

<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

- Mode、Gapsの詳細 (Mode="Burst"の場合)

この例では1回のシミュレーションで10パケット発生

The screenshot shows the TRex configuration interface with the following settings:

- Mode:** Burst (selected)
- Misc:** Enabled and Self start (checked)
- Numbers:** Number of Packets: 10, Number of Burst: 1, Packets per Burst: 1
- Gaps (in seconds):** ISG: 3.0, IBG: 0.0, IPG: (empty)

A diagram in the Gaps section shows a sequence: ISG (orange arrow) → PKT1 (blue arrow) → IPG (green arrow) → ...

ISG=シミュレーション開始後、何秒後にパケットを発生させるか設定
※画像の場合3秒後にパケット発生

キャプチャ開始後にシミュレーションをスタートしているため0.5s誤差があるが、1パケット目は**3.0s後(ISG)**に発生

The screenshot shows the capture output table with the following data:

No.	Time	Port	Mode	Destination	Source	Type	Length
1	3.53 s.	0	TX	192.168.254.1	192.168.1.1	IPv4	60
2	3.93 s.	0	TX	192.168.254.1	192.168.1.2	IPv4	60
3	4.33 s.	0	TX	192.168.254.1	192.168.1.3	IPv4	60
4	4.73 s.	0	TX	192.168.254.1	192.168.1.4	IPv4	60
5	5.13 s.	0	TX	192.168.254.1	192.168.1.5	IPv4	60
6	5.53 s.	0	TX	192.168.254.1	192.168.1.6	IPv4	60
7	5.93 s.	0	TX	192.168.254.1	192.168.1.7	IPv4	60
8	6.33 s.	0	TX	192.168.254.1	192.168.1.8	IPv4	60
9	6.73 s.	0	TX	192.168.254.1	192.168.1.9	IPv4	60
10	7.13 s.	0	TX	192.168.254.1	192.168.1.10	IPv4	60

10パケット発生したためシミュレーション終了

<参考>トラフィックジェネレーター TRex

4. 実際にL3トラフィックを発生させる手順

- Mode、Gapsの詳細 (Mode="Multi-Burst" の場合)

この例では5パケットを発生×3セット

The screenshot shows the TRex configuration interface with the following settings:

- Mode:** Multi-Burst (selected)
- Misc:** Enabled, Self start
- Numbers:** Number of Packets: 10, Number of Burst: 3, Packets per Burst: 5
- Gaps (in seconds):** ISG: 3.0, IBG: 2.0

Diagram illustrating the gap settings:

ISG = 開始後3秒後にパケット発生
IBG = 2セット目は2.0秒後発生

The screenshot shows the traffic capture interface with the following table:

No.	Time	Type	Length
1	3.47 s.	Pv4	60
2	3.87 s.	Pv4	60
3	4.27 s.	0	60
4	4.67 s.	TX	60
5	5.07 s.	0	60
6	7.07 s.	TX	60
7	7.47 s.	0	60
8	7.87 s.	TX	60
9	8.27 s.	0	60
10	8.67 s.	TX	60
11	10.67 s.	TX	60
12	11.07 s.	0	60
13	11.47 s.	Pv4	60
14	11.87 s.	Pv4	60
15	12.27 s.	Pv4	60

6パケット目は2.0s (IBG) の遅延がある

11パケット目も2.0s (IBG) の遅延がある

リモートからCML内ノードへの 接続方法

1. コンソールサーバ
2. ブレイクアウトツール

コンソールサーバ

- **CML自体がコンソールサーバ**として機能しており、SSH経由で接続することでCML内のノードにアクセスが可能

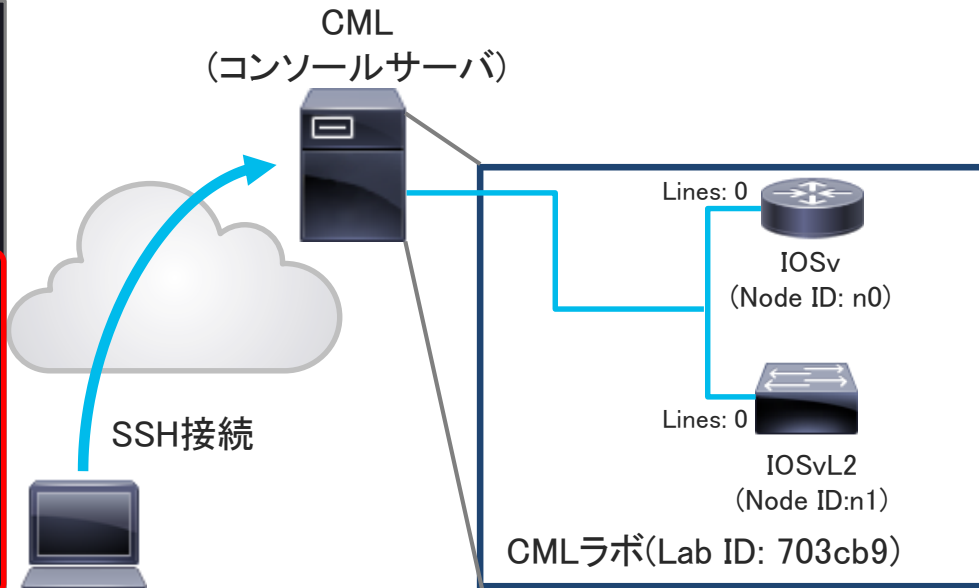
```
TATSAITO-M-80UF:~ tatsaito$ ssh -l admin 10.71.154.106
admin@10.71.154.106's password:

****
CML^2 Console Server
****

tab completion works
list available nodes and node labels / IDs with "list"
it's also possible to do a "open /lab_1/n0/0" command

consoles> list
Lab ID   Node ID   Lines      Node Label      Lab Title
-----
703cb9   n0        0,1        iosv-0          Basic Lab
703cb9   n1        0,1        iosvl2-0        Basic Lab
703cb9   n2        0,1        iosv-1          Basic Lab
consoles>
consoles> open /703cb9/n1/0
Connecting to console for iosvl2-0
Connected to terminalserver.
Escape character is '^]'.

iosvl2-0>
iosvl2-0>
```

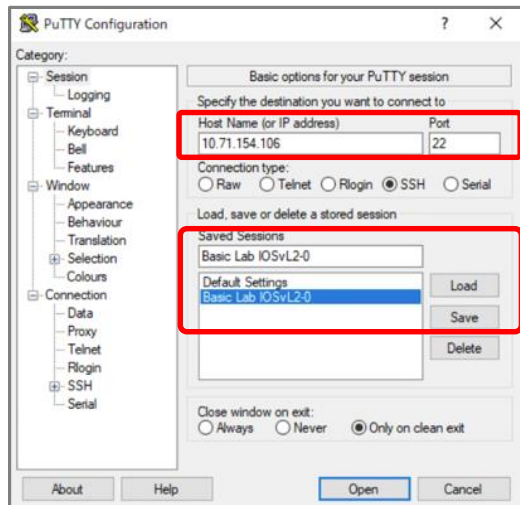
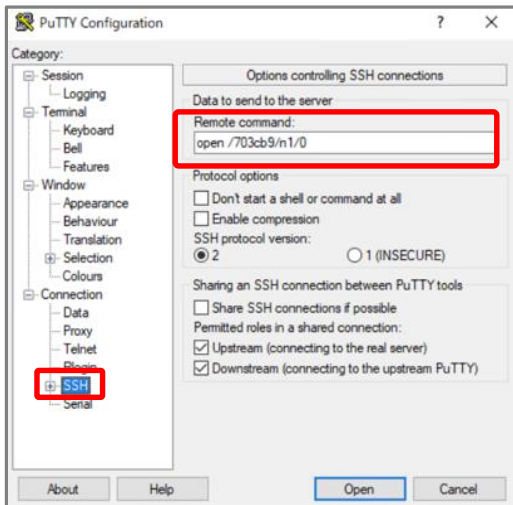


自身の端末

<参考>接続の簡素化 Windows編

- Windowsの場合、Puttyを使用することでSSHリモートコマンドを入力したセッションの保存が可能になり、2回目以降の接続が簡素化

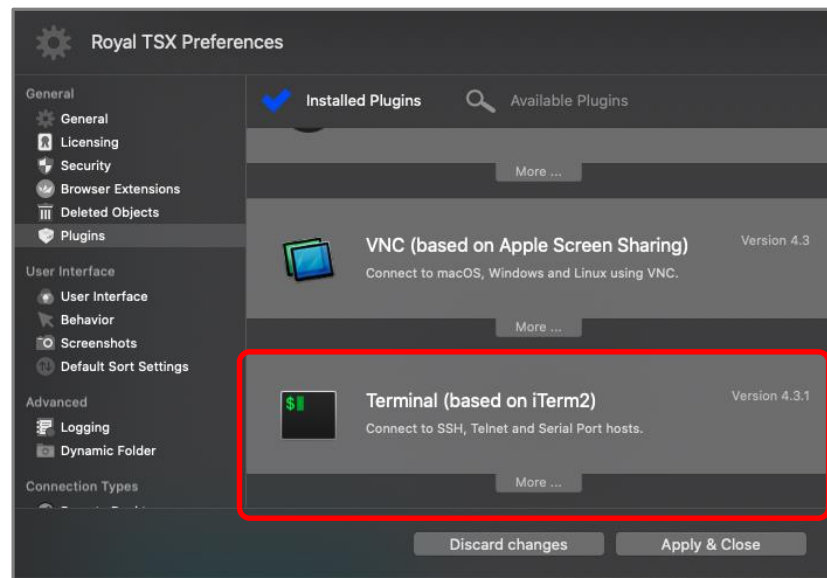
- SSHコマンド open /”ラボID”/”ノードID”/”ライン番号”を入力
- Host Name、Saved Sessionsにタイトルを入れ保存



```
10.71.154.106 - PuTTY
login as: admin
admin@10.71.154.106's password:
Connecting to console for iosv12-0
Connected to terminalserver.
Escape character is '^]'.
*****
* IOSv is strictly limited to use for evaluation, demons
* education. IOSv is provided as-is and is not supported
* Technical Advisory Center. Any use or disclosure, in w
* of the IOSv Software or Documentation to any third par
* purposes is expressly prohibited except as otherwise a
* Cisco in writing.
*****
iosv12-0>
```

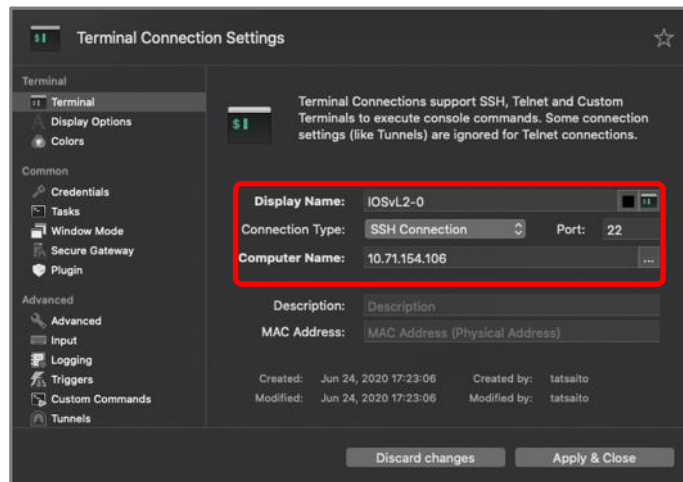
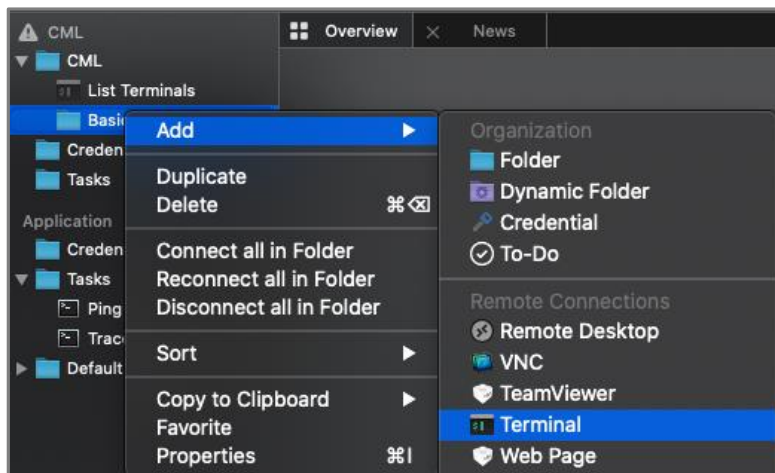
<参考>接続の簡素化 Mac編

- Macの場合、iTerm2とRoyal TSXを使用することでSSHリモートコマンドを入力したセッションの保存が可能になり、2回目以降の接続が簡素化
- Royal TSXをインストール後、Preferences > General > PluginsでiTerm2のプラグインをインストール



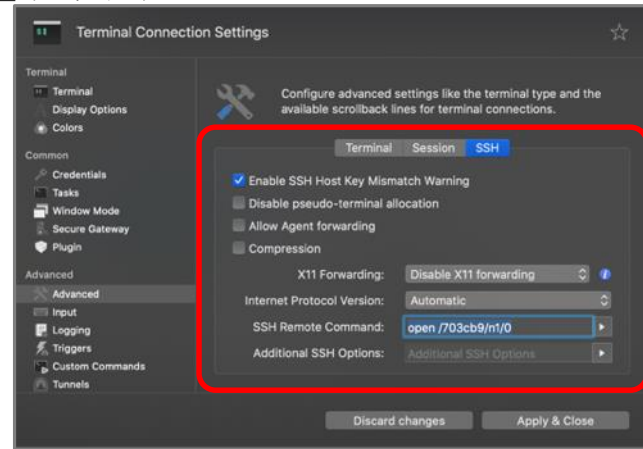
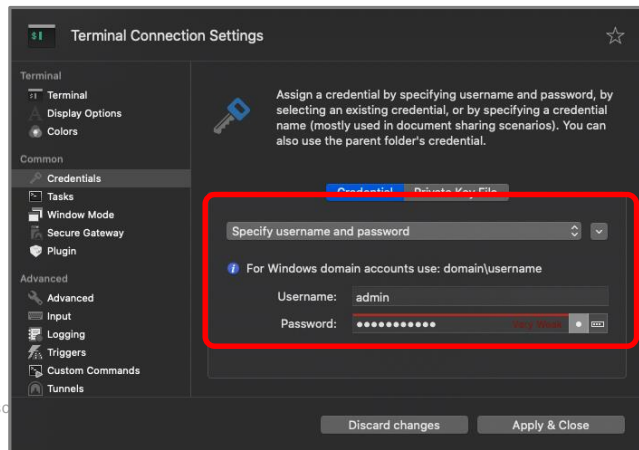
<参考>接続の簡素化 Mac編

- Macの場合、iTerm2とRoyal TSXを使用することでSSHリモートコマンドを入力したセッションの保存が可能になり、2回目以降の接続が簡素化
2. 任意のフォルダを作成 > “Add”> “Remote Connections”> “Terminal”をクリック
 3. Terminalの”Display Name”, “Computer Name(CMLのアドレス)”を入力



<参考>接続の簡素化 Mac編

- Macの場合、iTerm2とRoyal TSXを使用することでSSHリモートコマンドを入力したセッションの保存が可能になり、2回目以降の接続が簡素化
4. Commonの”Credentials”でCMLのログイン情報を入力
 5. Advancedの”SSH”の”SSH Remote Command”欄にopen /”ラボID”/”ノードID”/”ライン番号を入力し”Apply & Close”をクリック

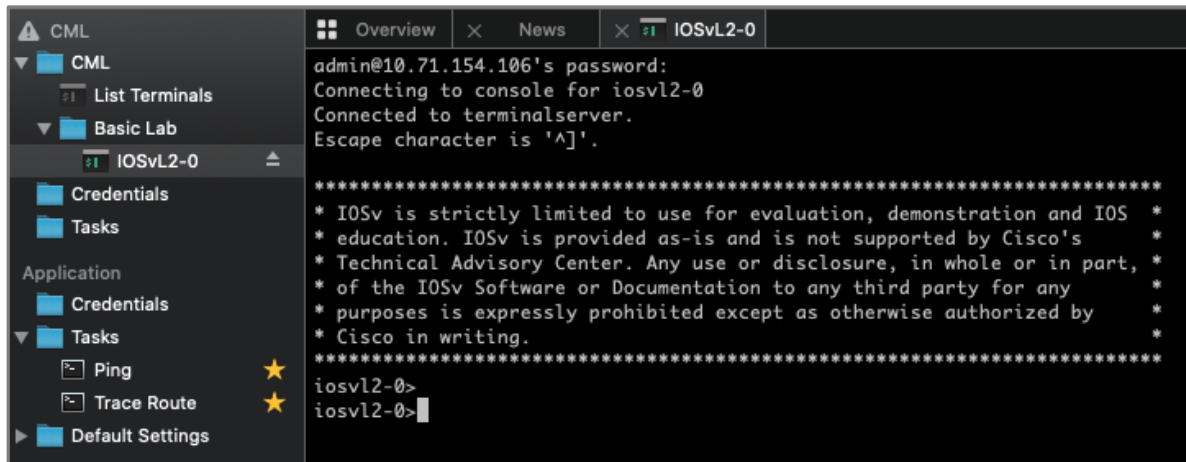


<参考>接続の簡素化 Mac編

- Macの場合、iTerm2とRoyal TSXを使用することでSSHリモートコマンドを入力したセッションの保存が可能になり、2回目以降の接続が簡素化

6. 作成したTerminalをダブルクリックすると接続が可能

※Royal TSXは**10コネクション以上はライセンスが必要なため注意**



The screenshot shows the Royal TSX application interface. On the left, a sidebar displays a tree view with folders for 'CML', 'Basic Lab', 'Credentials', 'Tasks', and 'Application'. The 'Basic Lab' folder is expanded, showing a terminal session named 'IOSvL2-0'. The main window displays the terminal output for this session, showing a successful connection to a console for 'iosvl2-0' and a prompt for the password. The terminal output includes a warning message about the limited use of IOSv software for evaluation and demonstration purposes.

```
CML
├── CML
│   └── List Terminals
├── Basic Lab
│   └── IOSvL2-0
├── Credentials
├── Tasks
├── Application
│   ├── Credentials
│   └── Tasks
│       ├── Ping
│       └── Trace Route
└── Default Settings
```

```
Overview | News | IOSvL2-0
admin@10.71.154.106's password:
Connecting to console for iosvl2-0
Connected to terminalserver.
Escape character is '^]'.

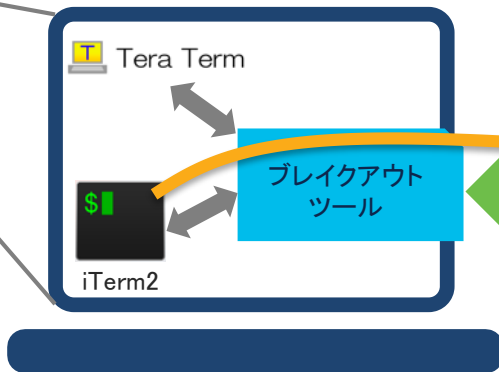
*****
* IOSv is strictly limited to use for evaluation, demonstration and IOS *
* education. IOSv is provided as-is and is not supported by Cisco's *
* Technical Advisory Center. Any use or disclosure, in whole or in part, *
* of the IOSv Software or Documentation to any third party for any *
* purposes is expressly prohibited except as otherwise authorized by *
* Cisco in writing.
*****
iosvl2-0>
iosvl2-0>
```

ブレイクアウトツール

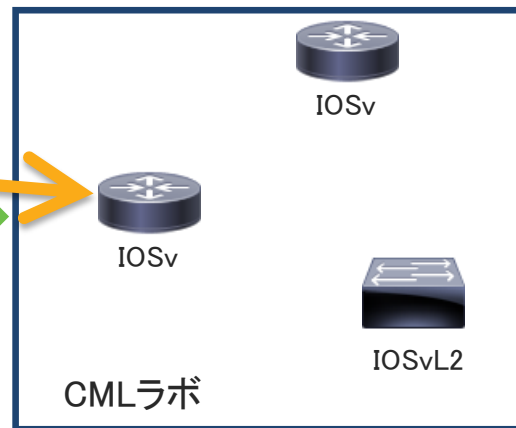
- クライアントにブレイクアウトツールをインストールすることで**クライアントをプロキシ**としてCMLラボのノードにコンソール接続が可能になる機能
- CML内のノードへのアクセスが**すべてTLS通信**
- 例) localhost:9000にTelnet接続が可能

```
TATSAITO-M-80UF:~ tatsaito$ telnet localhost 9000
Trying ::1...
Connected to localhost.
Escape character is '^]'.

iosv-0>en
Password:
iosv-0#
```



TLS通信



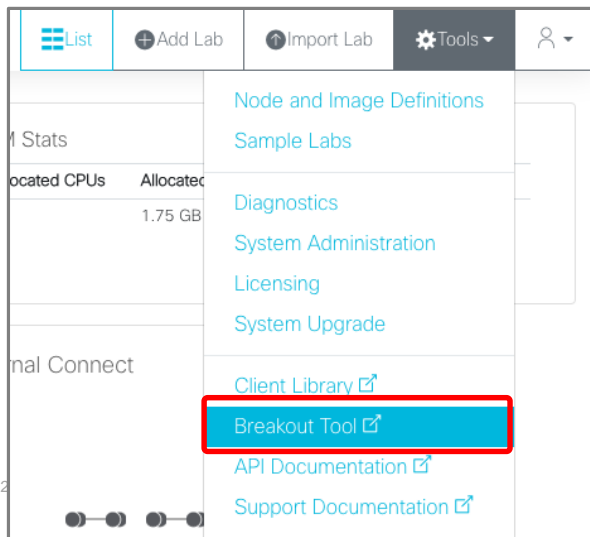
ブレイクアウトツール

- 設定手順

1. CMLの”Breakout Tool”ページよりブレイクアウトツールのファイルをダウンロード
2. ブレイクアウトツールファイルを実行可能な権限に変更(Mac,Linuxのみ)
3. ブレイクアウトツールのセキュリティのポップアップの確認(Macのみ)
4. ターミナルにてブレイクアウトツールの起動
5. ブラウザにてブレイクアウトツールの各種設定
 - CMLサーバのアドレスの設定
 - CMLサーバログイン時のクレデンシャルの設定
 - ブレイクアウトツールで接続するラボの有効化
6. ブレイクアウトツールの終了

ブレイクアウトツール Macの例

1. CMLの”Breakout Tool”ページよりブレイクアウトツールをダウンロード
 - CML右上の”Tools” > “Breakout Tool”をクリック
 - CML2 Breakoutページ下の”breakout-macos-x86_amd64”をクリックしブレイクアウトツールをダウンロード



Index of /breakout/

breakout-linux-x86_amd64	08-Apr-2020 19:39	10100816
breakout-macos-x86_amd64	08-Apr-2020 19:39	13390936
breakout-windows-x86_amd64.exe	08-Apr-2020 19:39	13134848

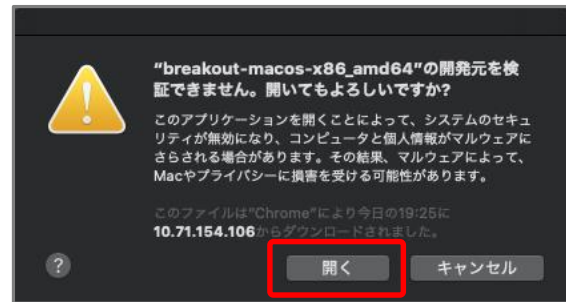
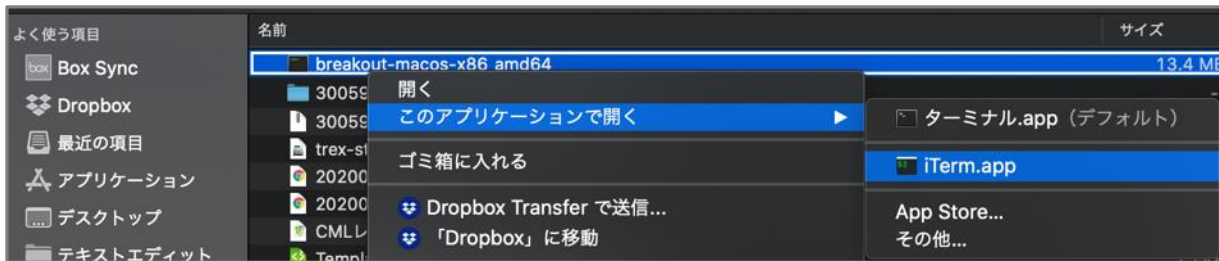
ブレイクアウトツール Macの例

- ブレイクアウトツールのファイルを実行可能な権限に変更(Mac,Linuxのみ)
 - MacやLinuxの場合、ダウンロードしたバイナリファイルを実行する前に実行可能な権限に変更

```
TATSAITO-M-80UF:~ tatsaito$ cd Downloads/
TATSAITO-M-80UF:Downloads tatsaito$ ls -alh | grep breakout-macos-x86_amd64
-rw-r--r--@  1 tatsaito  staff   13M  6 24 19:25 breakout-macos-x86_amd64
TATSAITO-M-80UF:Downloads tatsaito$
TATSAITO-M-80UF:Downloads tatsaito$ chmod u+x breakout-macos-x86_amd64
TATSAITO-M-80UF:Downloads tatsaito$
TATSAITO-M-80UF:Downloads tatsaito$ ls -alh | grep breakout-macos-x86_amd64
-rwxr--r--@  1 tatsaito  staff   13M  6 24 19:25 breakout-macos-x86_amd64
TATSAITO-M-80UF:Downloads tatsaito$ █
```

ブレイクアウトツール Macの例

- ブレイクアウトツールのセキュリティポップアップの確認(Macのみ)
 - ブレイクアウトツールをiTerm等のターミナルで開くとセキュリティのポップアップが表示されるが問題ないので開くをクリック
 - 一度ターミナル画面が開き、しばらくすると閉じてしまうが問題なし



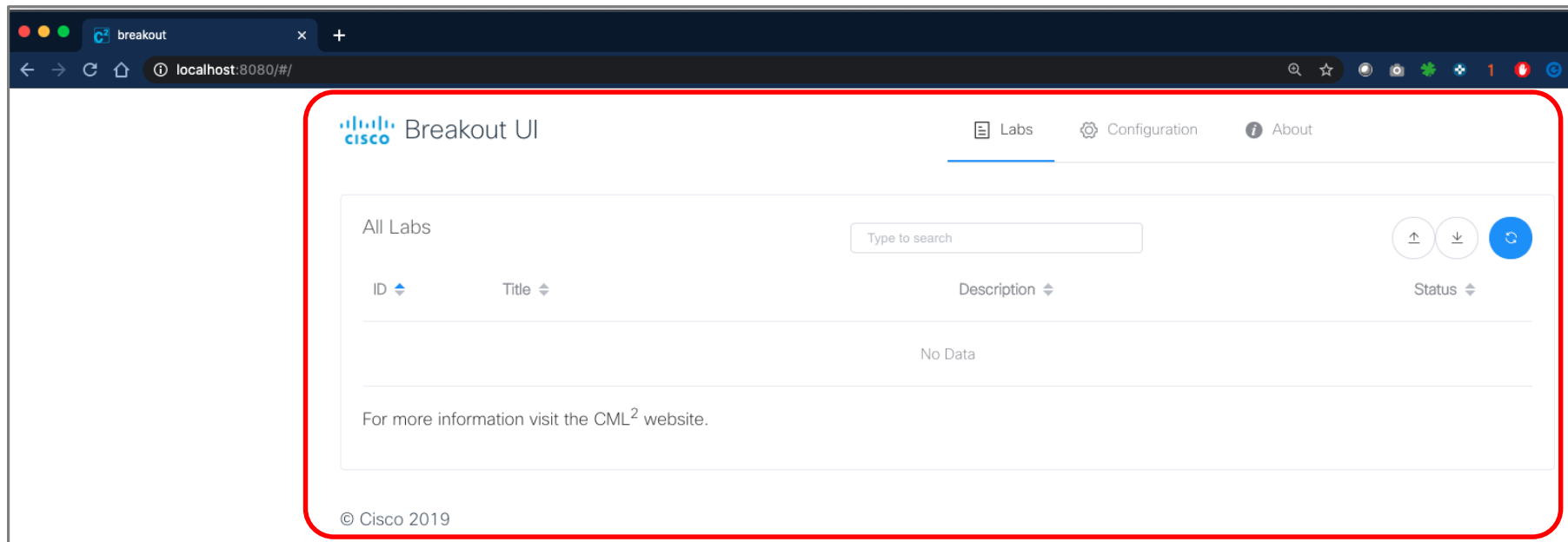
ブレイクアウトツール Macの例

4. ターミナルにてブレイクアウトツールの起動
 - ターミナルにてブレイクアウトツールのフォルダに移動し、
” ./breakout-macos-x86_amd64 ui”を実行
[http://\[::1\]:8080](http://[::1]:8080)にアクセスするよう表示されればOK

```
TATSAITO-M-80UF:Downloads tatsaito$ ./breakout-macos-x86_amd64 ui
Starting up...
W0624 19:38:42.453251 89269 run.go:238] open labs.yaml: no such file or directory
Running... Serving UI/API on http://[::1]:8080, Ctrl-C to stop.
```

ブレイクアウトツール Macの例

5. ブラウザにてブレイクアウトツールの各種設定
 - ブラウザにて<http://localhost:8080>にアクセスするとブレイクアウトUI画面が表示



ブレイクアウトツール Macの例

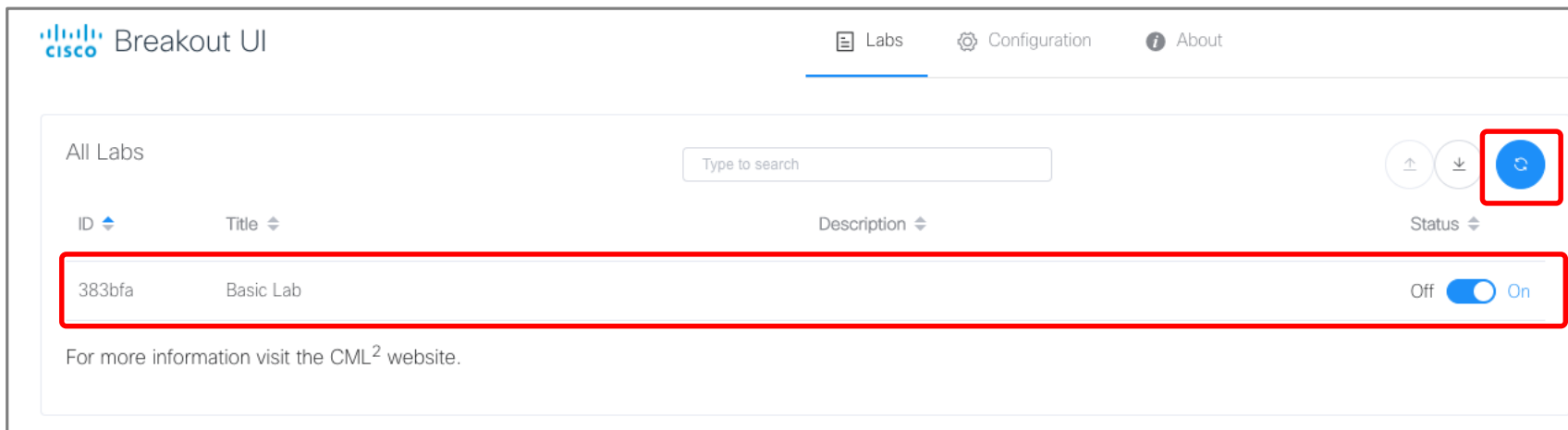
5. ブラウザにてブレイクアウトツールの各種設定

- “Configuration”タブをクリックし下記項目を入力し、保存
 - Controller Address: CMLのIPアドレス
 - Username/Password: CMLブラウザログイン時のユーザ名/パスワード

The screenshot shows the Cisco Breakout UI Configuration page. The 'Controller address' field is set to 'https://10.71.154.106' and the 'Username' field is set to 'admin'. Both fields are highlighted with red boxes. Other fields include 'Verify TLS certificate', 'Populate all nodes', 'Console Start Port', 'WVC Start Port', 'Listen Address', 'UI Server Port', and 'Labs Filename'.

ブレイクアウトツール Macの例

5. ブラウザにてブレイクアウトツールの各種設定
 - “Labs”タブに戻り右上の更新マークを押すと現在稼働しているラボ一覧が表示
ステータスが”OFF”になっているため、”ON”にする



The screenshot shows the Cisco Breakout UI interface. At the top, there are navigation tabs for "Labs", "Configuration", and "About". Below the navigation, there is a search bar labeled "Type to search". To the right of the search bar are three circular icons: an up arrow, a down arrow, and a refresh icon. The refresh icon is highlighted with a red square. Below the search bar is a table with columns for "ID", "Title", "Description", and "Status". The first row of the table is highlighted with a red rectangle and contains the following data: ID: 383bfa, Title: Basic Lab, and Status: Off. A red toggle switch is next to the "Off" status, and it is currently in the "On" position. Below the table, there is a link to the CML² website.

ID	Title	Description	Status
383bfa	Basic Lab		Off <input checked="" type="checkbox"/> On

ブレイクアウトツール Macの例

5. ブラウザにてブレイクアウトツールの各種設定
 - 接続したいラボをクリックすると、ノード一覧が表示
 - ノードのリンクをクリックするとコンソール接続が可能

ID	Node Label	Links	Status	Port Name	Port Number	Enable
n0	iosv-0	telnet://::1:9000	serial0 serial1	serial0	9000	<input checked="" type="checkbox"/>
		telnet://::1:9001		serial1	9001	<input type="checkbox"/>
n1	iosv2-0	telnet://::1:9002 telnet://::1:9003	serial0 serial1	serial0	9002	<input checked="" type="checkbox"/>
				serial1	9003	<input type="checkbox"/>
n2	iosv-1	telnet://::1:9004 telnet://::1:9005	serial0 serial1	serial0	9004	<input checked="" type="checkbox"/>
				serial1	9005	<input type="checkbox"/>



```
Trying ::1...
Connected to localhost.
Escape character is '^]'.

iosv-0>
iosv-0>en
Password:
iosv-0#
iosv-0#
iosv-0#
```

ブレイクアウトツール Macの例

6. ブレイクアウトツールの終了

- ブレイクアウトツール終了の際は、ターミナルにて“ctrl + c”を入力

```
2020/06/24 19:50:03 [TATSAITO-M-80UF/Ff5JbfZYaA-000021] "POST http://localhost:8080/api/co
nfig HTTP/1.1" from [::1]:58545 - 200 2B in 1.560794ms
get active VNC keys from controller...
get simplified node definitions from controller...
get active console keys from controller...
get all the labs from controller...
get all the nodes for the labs from controller...
get nodes for lab 383bfa from controller...
2020/06/24 19:50:47 [TATSAITO-M-80UF/Ff5JbfZYaA-000022] "GET http://localhost:8080/api/ref
resh HTTP/1.1" from [::1]:58545 - 200 2B in 795.151436ms
2020/06/24 19:50:47 [TATSAITO-M-80UF/Ff5JbfZYaA-000023] "GET http://localhost:8080/api/lis
t HTTP/1.1" from [::1]:58545 - 200 682B in 93.242µs
2020/06/24 19:52:02 [TATSAITO-M-80UF/Ff5JbfZYaA-000024] "POST http://localhost:8080/api/li
st HTTP/1.1" from [::1]:58545 - 200 76B in 283.345µs
2020/06/24 19:52:02 [TATSAITO-M-80UF/Ff5JbfZYaA-000025] "PUT http://localhost:8080/api/sta
rt/383bfa HTTP/1.1" from [::1]:58545 - 200 76B in 96.908236ms
2020/06/24 19:52:02 [TATSAITO-M-80UF/Ff5JbfZYaA-000026] "GET http://localhost:8080/api/lis
t HTTP/1.1" from [::1]:58545 - 200 702B in 60.393µs
2020/06/24 19:53:49 [TATSAITO-M-80UF/Ff5JbfZYaA-000027] "GET http://localhost:8080/api/lis
t HTTP/1.1" from [::1]:58545 - 200 702B in 63.449µs
^C
TATSAITO-M-80UF:Downloads tatsaito$
TATSAITO-M-80UF:Downloads tatsaito$
```

アジェンダ

1

前回のおさらい

2

仮想ノードのご紹介
新規仮想ノードの追加例

3

ネットワークシミュレーションツール
リモートからCML内ノードへの接続方法

4

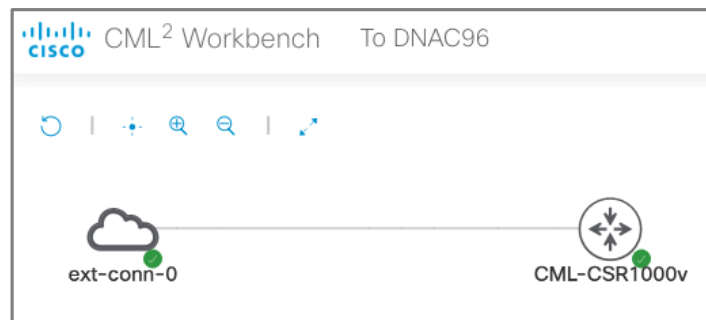
外部アプリケーションと組み合わせた
CMLの活用

5

資格試験での活用方法

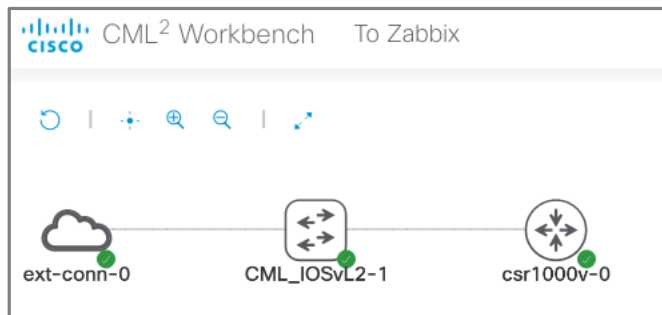
Cisco DNA Centerでの活用

- CMLのラボごとにCisco DNA Centerで自動化や分析の検証が可能



Zabbixでの活用

- CMLラボを使用しSNMPの監視やトラップの検証が可能



名前	アプリケーション	アイテム	トリガー	グラフ	ディスカバ	Web	インターフェース	テンプレート	ステータス	エージェントの状態	エージェント番号	情報	
Zabbix server	アプリケーション 11	アイテム 83	トリガー 52	グラフ 15	ディスカバ 12	Web 127.0.0.1:10050		Template App Zabbix Server, Template OS Linux (Template App Zabbix Agent)	有効	OK	SNMP	OK	なし
CML_IOSvL2-1	アプリケーション 9	アイテム 81	トリガー 22	グラフ 4	ディスカバ 18	Web 10.71.154.111:161		SNMP Trap, Template Net Cisco IOS SNMPv2 (Template Module Cisco CISCO-ENVMON-MIB SNMPv2, Template Module Cisco CISCO-MEMORY-POOL-MIB SNMPv2, Template Module Cisco CISCO-PROCESS-MIB SNMPv2, Template Module Cisco Inventory SNMPv2, Template Module EtherLike-MIB SNMPv2, Template Module Generic SNMPv2, Template Module Interfaces SNMPv2)	有効	OK	SNMP	OK	なし

名前	最新のチェック時刻	最新の値	変化
General (5アイテム)			
Device contact details	2020/07/10 10:00:03		ヒストリ
Device description	2020/07/10 10:00:03	Cisco IOS Software, ios_12 Software (ios_...	ヒストリ
Device location	2020/07/10 10:00:03		ヒストリ
Device name	2020/07/10 10:00:03	CML_IOSvL2-1	ヒストリ
System object ID	2020/07/10 10:00:03	SNMPv2-SMI:enterprises.9.1.1227	ヒストリ
Network interfaces (36アイテム)			
Interface Gi0/30 (port-ent-conn-0) : Bits received	2020/07/10 10:00:03	7.29 Kbps	-1.5 Kbps グラフ
Interface Gi0/30 (port-ent-conn-0) : Bits sent	2020/07/10 10:00:03	860 bps	-18 bps グラフ
Interface Gi0/30 (port-ent-conn-0) : Inbound packets discarded	2020/07/10 10:00:03	0	グラフ
Interface Gi0/30 (port-ent-conn-0) : Inbound packets with errors	2020/07/10 10:00:03	0	グラフ
Interface Gi0/30 (port-ent-conn-0) : Interface type	2020/07/10 10:00:03	ethernetCsmacd (8)	グラフ
Interface Gi0/30 (port-ent-conn-0) : Operational status	2020/07/10 10:02:03	up (1)	グラフ
Interface Gi0/30 (port-ent-conn-0) : Outbound packets discarded	2020/07/10 10:00:03	0	グラフ
Interface Gi0/30 (port-ent-conn-0) : Outbound packets with errors	2020/07/10 10:00:03	0	グラフ
Interface Gi0/30 (port-ent-conn-0) : Speed	2020/07/10 10:00:03	1 Gbps	グラフ
Interface Gi0/30 (port-ent-conn-0) : Bits received	2020/07/10 10:00:03	94 bps	グラフ
Interface Gi0/30 (port-ent-conn-0) : Bits sent	2020/07/10 10:00:03	194 bps	-18 bps グラフ

アジェンダ

1

前回のおさらい

2

仮想ノードのご紹介
新規仮想ノードの追加例

3

ネットワークシミュレーションツール
リモートからCML内ノードへの接続方法

4

外部アプリケーションと組み合わせた
CMLの活用

5

資格試験での活用方法

某シスコ社員からのコメント



ラーニングネットワークで公開している“勉強法”にも記述していますが、CCIE 受験にあたっては本番環境により近い構成を組んでの勉強が効果的です。
私は初回受験時に構成を覚えて、出来るだけ同じような環境を作り、そこで実際の動作を学んで次の受験に挑むというスタイルを取りました。
その際に苦労したのが、実機を揃えて、配線して、IOSを統一してと言った物理的な作業でした。環境を用意するだけで数日要したものです。

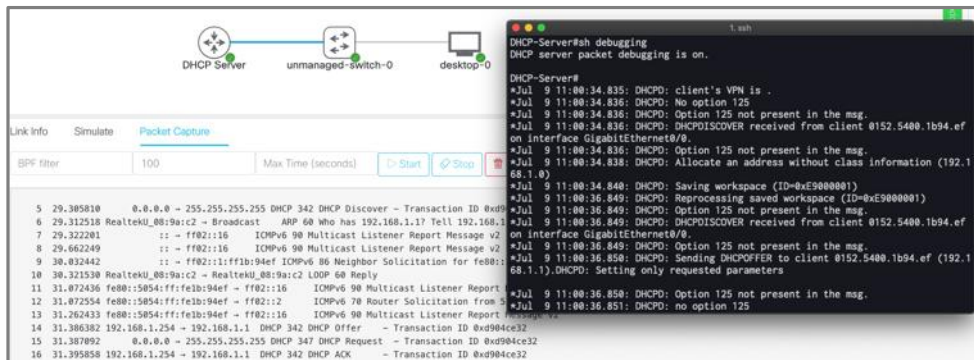
ですが、このCMLを使えば、これらの作業が数十分で済みます！

まさにCCIEを目指す方にとっては必須のツールではないかと考えます。

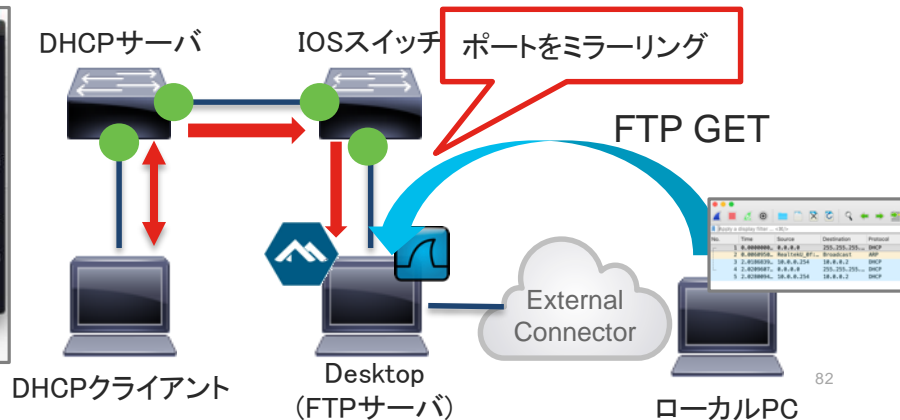
CCIE の勉強法 第1回 「合格までに要する勉強時間」
<https://learningnetwork.cisco.com/s/article/ccie-1-x>

資格試験での活用方法

- 各プロトコルの制御パケットやパケットウォークを正しく理解することが重要
例) DHCPやダイナミックルーティング、認証などの制御パケット
- ルータやスイッチのデバックコマンド、CMLの packets capture機能を使用
- ミラーリングやDesktopのWireshark、外部接続機能を使用



© 2020 Cisco and/or its affiliates. All rights reserved. Cisco Public



資格試験での活用方法

- CMLラボのインポート/エクスポート機能(YAML)を活用し、ラボの共有
 - 勉強時間の有効活用
 - ラボファイル(YAML)のファイルを変更し、トラブルシュートの練習に活用

YAMLファイルの中身は

- ラボの名前
- ノードのIDや座標(XY軸)
- show run
- リンクの状態

例)

- 対抗のIPアドレスを変更
- OSPFのネットワークタイプ変更
- EIGRPのK値を変更
- ACL、NATの編集など

```
Basic Lab (1).yaml
1 Lab:
2   description: ''
3   notes: ''
4   timestamp: 1594019567.669568
5   title: Basic Lab
6   version: 0.0.3
7 nodes:
8   - id: n0
9     label: iosv-0
10    node_definition: iosv
11    x: -700
12    y: -50
13    configuration: |-
14      Building configuration...
15
16      Current configuration : 3716 bytes
17      !
18      ! Last configuration change at 06:46:19 UTC Mon Jul 6 2020
19      !
20      version 15.8
21      service timestamps debug datetime msec
22      service timestamps log datetime msec
23      no service password-encryption
24      !
25      hostname iosv-0
```

まとめ

- 今回は10個の機能や使い方についてご紹介しました。
 1. ネットワーク機器のSyslog情報をSyslogサーバで受信
 2. SNMPマネージャコマンドを使用し情報を取得
 3. DesktopのWiresharkでパケットキャプチャを実施
 4. DockerHubの活用
 5. 新規仮想ノードの追加例
 6. WANエミュレータのご紹介
 7. TRexのご紹介
 8. コンソールサーバ、ブレイクアウトツールの使い方
 9. 外部アプリケーションと組み合わせたCMLの活用
 10. ラボ構成のインポート/エクスポート(YAML)

CMLは様々な使い方が可能なのでぜひご活用ください！！

参考リンク

- CMLご紹介ページ
<https://developer.cisco.com/modeling-labs/>
- CMLのドキュメント一覧ページ
<https://developer.cisco.com/docs/modeling-labs/>
- CML-Pのコミュニティサイト
<https://learningnetwork.cisco.com/s/topic/0TO3i00000094ZjGAI/cisco-modeling-labs-personal-community>
- GitHub 新規仮想ノード定義ファイル一覧ページ
<https://github.com/CiscoDevNet/cml-community/tree/master/node-definitions>
- TRex GUIクライアントソフトダウンロードページ
<https://github.com/cisco-system-traffic-generator/trex-stateless-gui/releases>
- Trexサーバ手動起動の設定ページ
<https://learningnetwork.cisco.com/s/feed/0D53i00000U2p7sCAB>

Cisco Modeling Labsを使ってネットワークを学ぼう！

- 2020年7月7日（火） Cisco Modeling Labs (CML)を使ってネットワークを学ぼう！（基礎編）
 - CML とは何か、ユースケースや基本機能、ラボ構築など、デモを交えながらご紹介します
- 2020年7月14日（火） Cisco Modeling Labs (CML)を使ってネットワークを学ぼう！（応用編）
 - CMLの様々な活用方法、資格試験での活用方法をデモを交えCCIEホルダーがご紹介します
- 2020年7月28日（火） Cisco Modeling Labs (CML)を使ってネットワークを学ぼう！（DevNet編）
 - CMLのAPI・プログラマビリティを用いたネットワーク自動化の観点でCMLをご紹介します
<https://learningnetwork.cisco.com/s/article/jp-webinar-cml03>

