User Guide

Network and Edge Virtual Machine Reference System Architecture Release v23.02

Authors

Aparna Balachandran

Francis Cahill

Octavia Carotti

Calin Gherghe

Joel A. Gibson

Dana Nehama

Michael O'Reilly

Jiri Prokes

Abhijit Sinha

Daniel Ugarte

1 Introduction

1.1 Purpose and Scope

The **Virtual Machine Reference System Architecture (VMRA)** is part of the Network and Edge Reference System Architectures Portfolio. The VMRA is a common virtual cluster template platform. It is composed of a set of virtual machines, implemented on a single physical Intel node or multi-nodes that can be used for hosting a Kubernetes cluster.

Network locations (for example, On-Premises Edge and Remote Central Office) require deployment of different hardware, software, and configuration specifications due to varying workloads, cost, density, and performance requirements. Configuration Profiles define prescribed sets of VMRA hardware and software components designed to optimally address the diverse deployment needs. Ansible playbooks implement the Configuration Profiles for fast, automatic deployment of needed VMRA capabilities. The result is an optimized installation of the VMRA Flavor as defined by the selected Configuration Profile. This user guide covers implementation of VMRA using several Configuration Profiles for Network Location specific and generic deployments.

Network-Location Configuration Profiles covered in this document include:

- **On-Premises Edge Configuration Profile** – Typical Customer Premises deployment supporting, for example, Smart City scenarios.
- **Remote Central Office-Forwarding Configuration Profile** – Near Edge deployments supporting fast packet-forwarding workloads such as Cable Modem Termination System (CMTS), User Plane Function (UPF) and Application Gateway Function (AGF).
- **Regional Data Center Configuration Profile** – Central-office location typical Configuration Profile. Currently tailored exclusively for 5G Core (5GC) and Media Visual Processing workloads such as CDN Transcoding.

Generic Configuration Profiles enable flexible deployments and include the following:

- **Basic Configuration Profile** – A generic minimum VMRA Kubernetes cluster setup.
- **Build-Your-Own Configuration Profile** – A complete set of all available software features targeted at developers and deployers who are looking to evaluate, control, and configure all the software and hardware ingredients and dependencies individually.

More information on Configuration Profiles and implementation of VMRA Flavors using the Configuration Profiles is provided later in this document.

1.2 User Guide Information

This document contains step-by-step instructions on installation, configuration, and use of networking and device plug-in features for deploying the VMRA Release v23.02 by implementing the VMRA template platform with the above Configuration Profiles. Validated, open source Ansible playbooks automatically provision the virtual environment along with a Kubernetes cluster (if desired) for the selected Configuration Profiles enabling user to create predictable deployments quickly and easily.

By following this document, it is possible to set up a virtual cluster based on Kubernetes with optimized configurations for cloud native deployments.

This document provides the following information:

- Part 1 (Sections 2 – 5): Requirements for hardware and software to prepare for the Ansible scripts.
- Part 2 (Sections 6 – 11): Step-by-step instructions on how to build each VMRA Flavor by implementing the configuration profiles. **If you wish to start building the VMRA right away, you may directly go to these sections and start automatically provisioning the VMRA Flavor of your choice.**
- Part 3 (Appendix A): VMRA Release Notes
- Part 4 (Appendix B): Abbreviations

See the Network and Edge Reference System Architectures Portfolio User Manual for an overview of the Reference System Architectures.

1.3 Version 23.02 Release Information

VMRA 23.02 common platform is based on 3rd and 4th Gen Intel® Xeon® Scalable processors and Intel® accelerators. Other advanced Intel® hardware technologies supported include the Intel® Ethernet Controller, Intel® QuickAssist Technology (Intel® QAT), and Intel® Server GPU.

Due to the hardware abstraction in the VMRA virtual setup, some hardware-dependent software features available in a Container Bare Metal Reference System Architecture (BMRA) are not supported by the VMRA. For details, about the technologies supported refer to the Network and Edge Reference System Architectures Portfolio User Manual.

The supported software components comprise open-source cloud-native software delivered by Intel, partners, and the open-source communities (e.g., Kubernetes, Telegraf, Istio, FD.io).

Release v23.02 builds upon prior releases. The following are the key release updates:

- Software Updates (details in Reference Architecture Software Components)
- Support for non-root user deployment
- Support for custom cluster name

For additional details, refer to the VMRA Release Notes.

Experience Kits, the collaterals that explain in detail the technologies enabled in VMRA release 23.02, including benchmark information, are available at Network & Edge Platform Experience Kits.

Table of Contents

Figures

Tables

Document Revision History

Revision	Date	Description
001	February 2022	Initial release.
002	March 2022	Updated a few URLs.
003	June 2022	Covers the 4th Gen Intel® Xeon® Scalable processor (formerly code named Sapphire Rapids).
004	June 2022	Changes include updates to the Known Issues section.
005	July 2022	Updated Istio and service mesh features.
006	October 2022	Updated for RA release 22.08.
007	December 2022	Updated for RA release 22.11.
008	March 2023	Updated for RA Release 23.02, with changes to deployments.

1.4 Key Terms

Table 1 lists the key terms used throughout the portfolio. These terms are specific to Network and Edge Reference System Architectures Portfolio deployments.

Table 1. Terms Used

TERM	DESCRIPTION
Experience Kits	Guidelines delivered in the form of—manuals, user guides, application notes, solution briefs, training videos—for best-practice implementation of cloud native and Kubernetes technologies to ease developments and deployments.
Network and Edge Reference System Architectures Portfolio	A templated system-level blueprint for a range of locations in enterprise and cloud infrastructure with automated deployment tools. The portfolio integrates the latest Intel platforms and cloud-native technologies for multiple deployment models to simplify and accelerate deployments of key workloads across a service infrastructure.
Deployment Model	Provides flexibility to deploy solutions according to IT needs. The portfolio offers three deployment models: • **Container Bare Metal Reference System Architecture (BMRA)** – A deployment model of a Kubernetes cluster with containers on a bare metal platform. • **Virtual Machine Reference System Architecture (VMRA)** – A deployment model of a virtual cluster on a physical node. The virtual cluster can be a Kubernetes containers-based cluster. • **Cloud Reference System Architecture (Cloud RA)** – A deployment model of a cluster on a public Cloud Service Provider. The cluster can be Kubernetes with containers based.
Configuration Profiles	A prescribed set of components—hardware, software modules, hardware/software configuration specifications—designed for a deployment for specific workloads at a network location (such as On-Premises Edge). Configuration Profiles define the components for optimized performance, usability, and cost per network location and workload needs. In addition, generic Configuration Profiles are available to developers for flexible deployments.
Reference Architecture Flavor	An instance of reference architecture generated by implementing a Configuration Profile specification.
Ansible Playbook	A set of validated scripts that prepare, configure, and deploy a Reference Architecture Flavor per Configuration Profile specification.
Configuration Profile Ansible Scripts	Automates quick, repeatable, and predictive deployments using Ansible playbooks. Various Configuration Profiles and Ansible scripts allow automated installations that are application-ready, depending on the workload and network location.
Kubernetes cluster	A deployment that installs at least one worker node running containerized applications. Pods are the components of the application workload that are hosted on worker nodes. Control nodes manage the pods and worker nodes.
Intel® Platforms	Prescribes Intel platforms for optimized operations. The platforms are based on 3rd Gen and 4th Gen Intel® Xeon® Scalable processors. The platforms integrate Intel® Ethernet Controller 700 Series and 800 Series, Intel® QuickAssist Technology (Intel® QAT), Intel® Server GPU (Graphic Processor Unit), Intel® Optane™ technology, and more. ***Note:*** This release of VMRA does not support the Intel® Xeon® D processor.

In addition to key terms, portfolio deployment procedures follow a hardware and software configuration taxonomy. Table 2 describes the taxonomy used throughout this document.

Table 2. Hardware and Software Configuration Taxonomy

TERM	DESCRIPTION
Hardware Taxonomy	
ENABLED	Setting must be enabled in the BIOS (configured as Enabled, Yes, True, or similar value)
DISABLED	Setting must be disabled in the BIOS (configured as Disabled, No, False, or any other value with this meaning.)
OPTIONAL	Setting can be either disabled or enabled, depending on workload. Setting does not affect the Configuration Profile or platform deployment
Software Taxonomy	
TRUE	Feature is included and enabled by default
FALSE	Feature is included but disabled by default - can be enabled and configured by user
N/A	Feature is not included and cannot be enabled or configured

1.5 Intel Investments of Capabilities

Intel investments in networking solutions are designed to help IT centers accelerate deployments, improve operational efficiencies, and lower costs. Table 3 highlights Intel investments in the portfolio and their benefits.

Table 3. Intel Capabilities Investments and Benefits

CAPABILITY	BENEFIT
Performance	Intel platform innovation and accelerators, combined with packet processing innovation for cloud-native environments, deliver superior and predictive application and network performance.
Orchestration and Automation	Implementing Kubernetes containers orchestration, including Kubernetes Operators, simplifies and manages deployments and removes barriers in Kubernetes to support networking functionality.
Observability	Collecting platform metrics by using, as an example, the collectd daemon and Telegraf server agent, publishing the data, and generating reports, enables high visibility of platform status and health.
Power Management	Leveraging Intel platform innovation, such as Intel® Speed Select Technology (Intel® SST), supports optimized platform power utilization.
Security	Intel security technologies help ensure platform and transport security. These technologies include the following: • Intel® Security Libraries for Data Center (Intel® SecL - DC) • Intel® QuickAssist Technology Engine for OpenSSL* (Intel® QAT Engine for OpenSSL*) • Intel® Software Guard Extensions (Intel® SGX) • Key Management Reference Application (KMRA) implementation
Storage	Creating a disaggregated, high-performance, scalable storage platform using MinIO Object Storage supports data-intensive applications, such as media streaming, big data analytics, AI, and machine learning.
Service Mesh	Implementing a Service Mesh architecture using Istio allows application services that can be added, connected, monitored, more secure, and load-balanced with few or no code changes. Service Mesh is integrated with Trusted Certificate Service for Kubernetes* platform, providing more secure Key Management.

1.6 Reference Documentation

The *Network and Edge Reference System Architectures Portfolio User Manual* contains a complete list of reference documents. Additionally, a bare metal-based reference system architecture (BMRA) deployment allows creation of a Kubernetes cluster on multiple nodes. The *Network and Edge Container Bare Metal Reference System Architecture User Guide* provides information and installation instructions for a BMRA. The Cloud Reference System Architecture (Cloud RA) provides the means to develop and deploy cloud-native applications in a CSP environment and still experience Intel® technology benefits. Find more details in the Network and Edge Cloud Reference System Architecture User Guide.

Other collaterals, including technical guides and solution briefs that explain in detail the technologies enabled in VMRA release v23.02, are available in the following location: Network & Edge Platform Experience Kits.

Part 1:
Reference Architecture Deployment:
Ansible Playbooks
Common Hardware Components
Software Ingredients
Recommended Configurations

2 Reference Architecture Deployment

This chapter explains how a VMRA Flavor is generated and deployed. The process includes installation of the hardware setup followed by system provisioning.

2.1 VMRA Architecture

The VMRA is a virtual cluster implemented on a single or multiple physical Intel nodes (Figure 1). VMRA supports both a virtual Kubernetes cluster and a VMRA cluster with a scalable number of VMs. The VMs are connected as a virtual cluster of worker and control VMs. A VMRA allows flexible deployment options for creating networking solutions for production or testing.

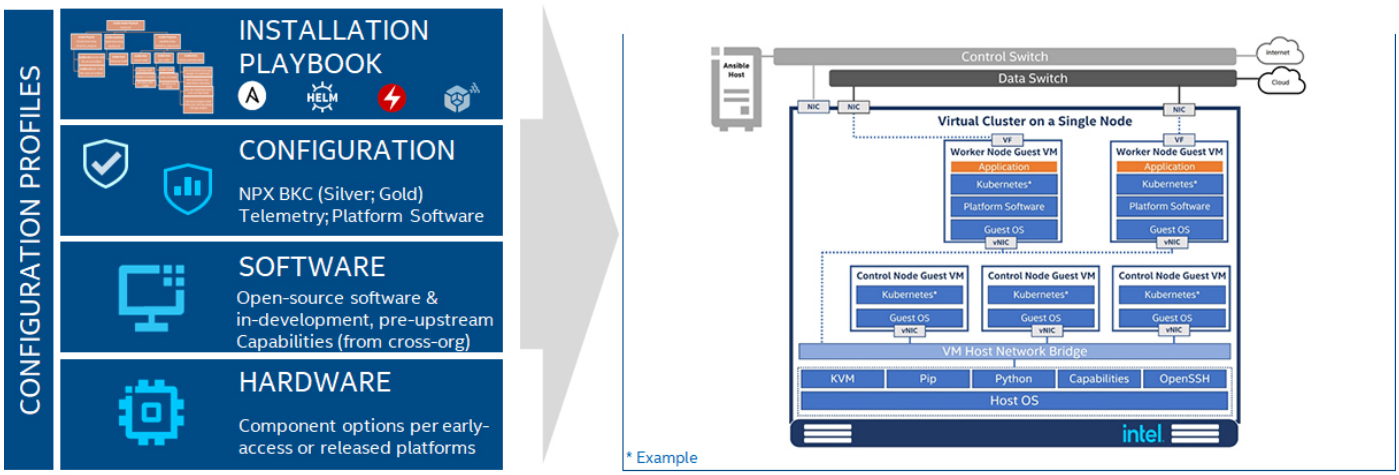

Figure 1 Virtual Machine Reference System Architecture Illustration with Kubernetes Cluster

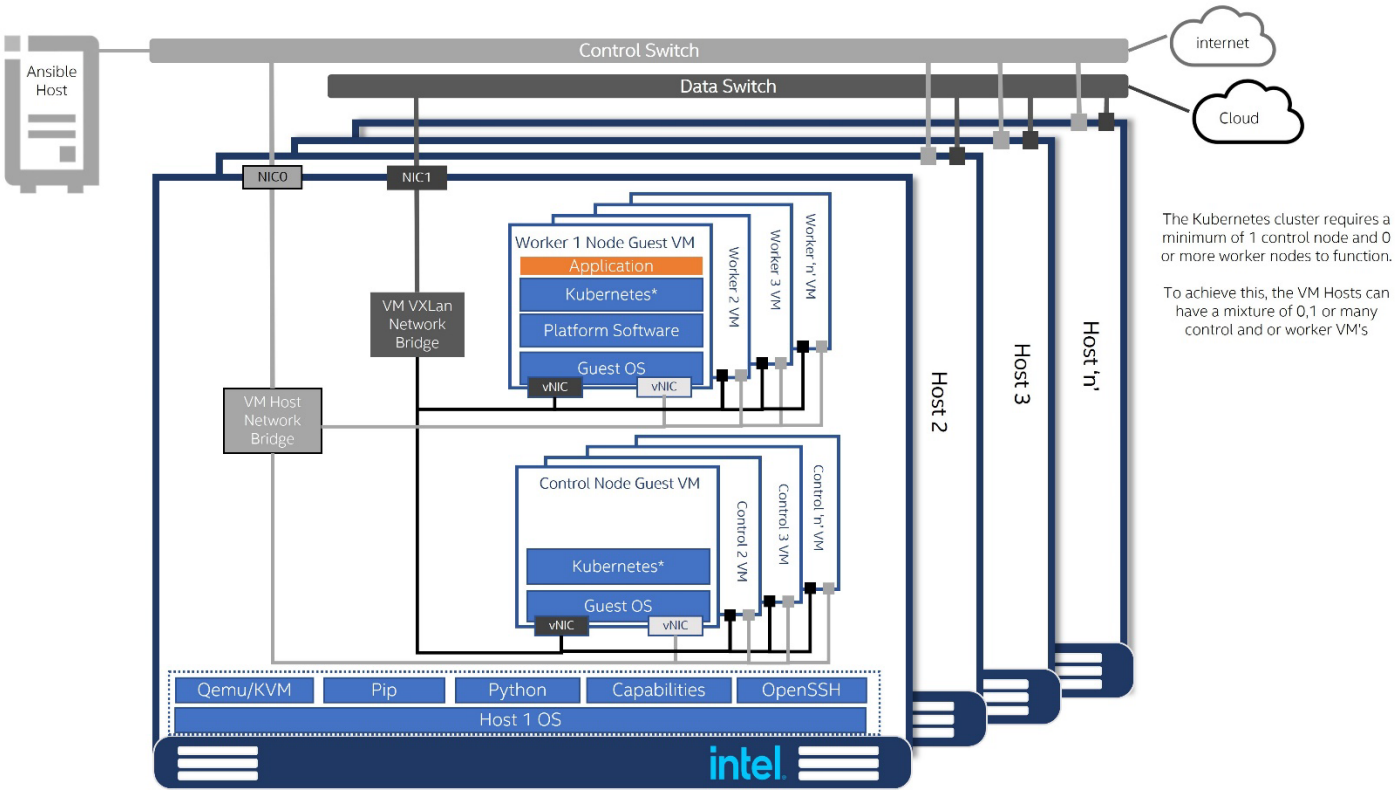

Figure 2. VMRA Multiple-Node Deployment

2.2 Configuration Profiles

A Configuration Profile describes specific hardware and software bill of materials (BOM) and configurations, applicable for a specific deployment. Configuration Profiles take into consideration the best-known configuration (BKC) validated by Intel for optimized performance.

Installation scripts implement a VMRA Flavor by deploying the required components specified by a Configuration Profile. Each VMRA Flavor is built on the following:

- **Intel Platform foundation** with Intel processors and technologies.
- **Hardware BOM** optimized for delivering an application at a specific location using a deployment model. For example, to support a UPF workload at the Remote CO, the VMRA deployment is populated with the maximum available Intel® Ethernet Adapters.
- **Software BOM** leverages the Intel platform and enables cloud-native adoption.
- **Installation (Ansible) Playbook** automates the installation of a Reference Architecture Flavor per a Configuration Profile specification.

The following Reference Architecture Configuration Profiles are network location-specific:

- **On-Premises Edge Configuration Profile** – Small cluster of stationary or mobile server platforms, ranging from one to four servers. Usage scenarios include data collection from sensors, local (edge) processing, and upstream data transmission. Sample locations are hospitals, factory floors, law enforcement, media, cargo transportation, power utilities. This Configuration Profile recommends a Kubernetes cluster hardware configuration, software capabilities, and specific hardware and software configurations that typically support enterprise edge workloads used in SMTC deployments and Ad-insertion.
- **Remote Central Office-Forwarding Configuration Profile** – Clusters ranging from a half rack to a few racks of servers, typically in a pre-existing, repurposed, unmanned structure. The usage scenarios include running latency-sensitive applications near the user (for example, real-time gaming, stock trading, video conferencing). This Configuration Profile addresses a Kubernetes cluster hardware, software capabilities, and configurations that enable high performance for packet forwarding packets. In this category, you can find workloads such as UPF, vBNG, vCMTS, and vCDN.
- **Regional Data Center Configuration Profile** – The Regional Data Center consists of a management domain with many racks of servers, typically managed and orchestrated by a single instance of resource orchestration. Usage scenarios include services such as content delivery, media, mobile connectivity, and cloud services. This Configuration Profile is tailored exclusively and defined for Media Visual Processing workloads such as CDN Transcoding.

Two additional Reference Architecture Configuration Profiles that are not location-specific enable flexible deployments per need:

- **Basic Configuration Profile** – Minimum VMRA Kubernetes cluster setup.
- **Build-Your-Own Configuration Profile** – A complete set of all available software features targeted at developers and deployers that are looking to evaluate, control, and configure all the software and hardware ingredients and dependencies individually.

2.3 Reference Architecture Installation Prerequisites

This section helps you get ready for running the Ansible scripts. Before the Ansible playbook can begin, you must identify the required hardware components, ensure hardware connectivity, and complete the initial configuration, for example BIOS setup.

This section describes the minimal system prerequisites needed for the Ansible and VM hosts. It also lists the steps required to prepare hosts for successful deployment. Detailed instructions are provided in relative sections, which are referred to in this section. Steps include:

- Hardware BOM selection and setup
- Required BIOS/UEFI configuration, including virtualization and hyper-threading settings
- Network topology requirements – a list of necessary network connections between the nodes
- Installation of software dependencies needed to execute Ansible playbooks
- Generation and distribution of SSH keys that are used for authentication between the Ansible host and VM host

After satisfying these prerequisites, Ansible playbooks for 3rd Gen Intel Xeon Scalable processors can be downloaded directly from the dedicated GitHub* page (https://github.com/intel/container-experience-kits/releases) or cloned using the Git. Be sure to complete the software prerequisites below before downloading the Ansible playbooks. Request access to the Ansible playbooks for the 4th Gen Intel Xeon Scalable processor from your regional Intel representative.

2.3.1 Hardware BOM Selection and Setup for Nodes

Before software deployment and configuration, deploy the physical hardware infrastructure for the site. To obtain ideal performance and latency characteristics for a given network location, Intel recommends the hardware configurations described in the following sections:

- VM host – Refer to the following sections for recommended host assembly:
  - Base host node – Review Section 3.1 to satisfy base performance characteristics.
  - Plus host node – Review Section 3.2 to satisfy plus performance characteristics.
- Configuration Profile BOM – See Set Up the VM Host – BIOS and HW Prerequisites and Sections 7 through 11 for details about hardware BOM selection and setup for your chosen Configuration Profile.

2.3.2 BIOS Configuration for VM Host

Enter the UEFI or BIOS menu and update the configuration based on Platform BIOS Profiles Settings, which describe the BIOS options in detail. Use Table 16 as a reference for selecting the right BIOS Profile.

2.3.3 Operating System Selection for VM Host and VMs

The following Linux operating systems are supported for the VM host and VMs:

- Ubuntu 22.04 (22.04)
- Rocky Linux 8.5 (only supported for VMs)
- Rocky Linux 9.0

For the supported distribution, the base operating system install image is sufficient to be built using the "Minimal" option during installation. In addition, the following must be met:

- The VM host must have network connectivity to the Ansible host.
- SSH connections are supported. If needed, install SSH Server with the following commands (internet access is required):

```
# sudo apt update
# sudo apt install openssh-server
```

2.3.4 Network Interface Requirements for VM Host

The following list provides a brief description of different networks and network interfaces needed for deployment:

- Internet network
  - Available for VMs through a Linux bridge on the host, providing internet connectivity through NAT
  - Ansible host accessible
  - Capable of downloading packages from the internet
  - Can be configured for Dynamic Host Configuration Protocol (DHCP) or with static IP address
- Management network and Calico pod network interface for Kubernetes installs (This can be a shared interface with the internet network)
  - Available for VMs through a Linux bridge on the host, connected to other nodes through VXLAN
  - Kubernetes control and worker node inter-node communications (for Kubernetes installs)
  - Calico pod network runs over this network (for Kubernetes installs)
  - Configured to use a private static address
- Tenant data networks
  - Dedicated networks for traffic
  - SR-IOV enabled
  - Virtual function (VF) can be DPDK bound in pod

2.3.5 Software Prerequisites for Ansible Host and VM Host

Before deployment of the VMRA Ansible playbooks, the Ansible host must be prepared. To successfully run the deployment, perform the following tasks before you download the VMRA Ansible code.

Perform the following steps:

1. Log in to the Ansible host machine using SSH or your preferred method to access the shell on that machine.
2. Install packages on Ansible Host. The following example assumes that the host is running RHEL 8. Other operating systems may have slightly different installation steps:

```
$ yum install python3
$ yum install libselinux-python3
```

3. Enable passwordless login between all VM Hosts and Ansible Host.
4. Create authentication SSH-Keygen keys on Ansible Host:

```
$ ssh-keygen
```

5. Upload generated public keys to all VM Hosts from Ansible Host:

```
$ ssh-copy-id root@<target_server_address>
```

2.4 Ansible Playbook

This section describes how the Ansible playbooks allow for an automated deployment of a fully functional VMRA cluster, including initial system configuration, Kubernetes deployment, and setup of capabilities as described in Section 2.5.

2.4.1 Ansible Playbooks Building Blocks

The following components make up the VMRA Ansible playbooks.

Note: Ansible playbooks for 3rd Gen and 4th Gen Intel Xeon Scalable processors are open source and available here.

Configuration Files provide examples of cluster-wide and host-specific configuration options for each of the Configuration Profiles. With minimal changes, they can be used directly with their corresponding playbooks. The path to these Configuration Files is:

- inventory.ini
- group_vars/all.yml
- host_vars/host-for-vms-1.yml
- host_vars/host-for-vms-2.yml (used in case of multi-node setup)
- host_vars/vm-ctrl-1.yml
- host_vars/vm-work-1.yml (each vm-work node needs own host_vars file)
- host_vars/vm-ctrl-1.cluster1.local.yml (when vm_cluster_name: "cluster1.local" is defined)
- host_vars/vm-work-1.cluster1.local.yml (each vm-work node needs own host_vars file)

For default values in these files, refer to the Configuration Profile-specific sections for VMRA installations: Section 7, VMRA Basic Configuration Profile Setup, Section 8, VMRA Build-Your-Own Configuration Profile Setup, Section 9, VMRA On-Premises Edge Configuration Profile Setup, Section 10, VMRA Remote Central Office-Forwarding Configuration Profile Setup, and Section 11, VMRA Regional Data Center Configuration Profile Setup.

Ansible Playbooks act as a user entry point and include all relevant Ansible roles and Helm charts. Top-level Ansible playbooks exist for each Configuration Profile, which allows lean use case-oriented cluster deployments. Each playbook includes only the Ansible roles and configuration files that are relevant for a given use case.

- playbooks/basic.yml
- playbooks/on_prem.yml
- playbooks/remote_fp.yml
- playbooks/regional_dc.yml
- playbooks/build_your_own.yml

VMRA is deployed through a single playbook that utilizes one of the playbooks for the Configuration Profiles you will deploy. In addition, the VMRA playbook ensures that both the host and VMs are configured as part of the infrastructure setup.

- playbooks/vm.yml

Each of these playbooks encompasses **Ansible Roles** grouped into three main execution phases, which are further explained in the next section:

- Infrastructure Setup
- Kubernetes Deployment
- Capabilities Setup

Note that several Capabilities Setup roles include nested Helm charts for easier deployment and lifecycle management of deployed applications, as well as a group of **Common Utility Roles** that provide reusable functionality across the playbooks.

2.4.2 Ansible Playbook Phases

Regardless of the selected Configuration Profile, the installation process always consists of five main phases:

1. **Host Infrastructure Setup** (sub-playbooks located in playbooks/infra/ directory)

   These playbooks modify kernel boot parameters and apply the initial system configuration for the host. Depending on the selected Configuration Profile, Host Infrastructure Setup includes:

   - Generic host OS preparation, for example, installation of required packages, Linux kernel configuration, proxy configuration, and modification of SELinux policies and firewall rules
   - Configuration of the kernel boot parameters according to the user-provided configuration to configure CPU isolation, SR-IOV related settings such as IOMMU, hugepages, or explicitly enable/disable Intel P-state technology
   - Configuration of SR-IOV capable network cards and QAT devices. This includes the creation of virtual functions and binding to appropriate Linux kernel modules
   - Network Adapter drivers and firmware updates, which help ensure that all latest capabilities such as Dynamic Device Personalization (DDP) profiles are enabled
   - Installation of Dynamic Device Personalization profiles, which can increase packet throughput, help reduce latency, and lower CPU usage by offloading packet classification and load balancing to the network adapter

2. **Host Virtualization Setup** (playbooks/infra/prepare_vms.yml)

   This playbook installs and configures the virtualization layer and VMs that will be used as Kubernetes nodes later in the installation. Host Virtualization Setup includes:

   - Installing VM hypervisor and tools to manage VMs and images, such as QEMU, KVM, Libvirt, and Genisoimage
   - Create backing and configuration images for each VM
   - Create VXLAN bridges to ensure VMs connectivity cross multiple nodes
   - Start the VMs and perform optimization tasks (ISOLCPUS, CPU pinning, and NUMA alignment)
   - Collect information from VMs, make sure they are accessible
   - Update the Ansible Inventory to include VMs as controller and worker nodes according to the configuration

3. **VM Infrastructure Setup** (sub-playbooks located in playbooks/infra/ directory)

   These playbooks modify kernel boot parameters and apply the initial system configuration for the cluster nodes. Depending on the selected Configuration Profile, VM Infrastructure Setup includes:

   - Generic host OS preparation, for example, installation of required packages, Linux kernel configuration, proxy and DNS configuration, and modification of SELinux policies and firewall rules
   - Configuration of the kernel boot parameters according to the user-provided configuration to configure CPU isolation, hugepages, or explicitly enable/disable Intel P-state technology
   - Configuration of SR-IOV and QAT devices
   - Network Adapter drivers and firmware updates.

4. **Kubernetes Setup** (located in playbooks/k8s/ directory)

   This playbook deploys a high availability (HA) Kubernetes cluster using Kubespray. Kubespray is a project under the Kubernetes community that deploys production-ready Kubernetes clusters. The Multus CNI plugin, which is specifically designed to provide support for multiple networking interfaces in a Kubernetes environment, is deployed by Kubespray along with Calico and Helm. Preferred security practices are used in the default configuration. On top of Kubespray, there's also a container registry instance deployed to store images of various control-plane Kubernetes applications.

5. **VMRA System Capabilities Setup** (sub-playbooks located in playbooks/intel directory):

   Advanced networking technologies, Enhanced Platform Awareness, and device plugin features are deployed by this playbook using operators or Helm Charts as part of the VMRA. The following capabilities are deployed:

   - Device plugins that allow using, for example, SR-IOV, and QAT devices in workloads running on top of Kubernetes.
   - CNI Plugins, which allow Kubernetes pods to be attached directly to accelerated and highly available hardware and software network interfaces.
   - Node Feature Discovery (NFD), which is a Kubernetes add-on to detect and advertise hardware and software capabilities of a platform that can, in turn, be used to facilitate intelligent scheduling of a workload.
   - Platform Aware Scheduling, which allows scheduling of workloads based on telemetry data.
   - Full Telemetry Stack consisting of Telegraf, Kube-Prometheus, and Grafana, which gives cluster and workload monitoring capabilities and acts as a source of metrics that can be used in TAS to orchestrate scheduling decisions.

2.5 Deployment Using Ansible Playbook

This section describes common steps to obtain the VMRA Ansible Playbooks source code, prepare target servers, configure inventory and variable files, and deploy the VMRA Kubernetes cluster.

2.5.1 Prepare VM Host Server

For the VM host server, you must make sure that it meets the following requirements:

- Python 3 is installed. The version depends on the target distribution.
- SSH keys are exchanged between the Ansible host and VM host.
- Internet access on the VM host is mandatory. Proxies are supported and can be configured in the Ansible vars.
- Additional NIC assigned with IP for VxLAN communication among all VMs on all VM hosts
- BIOS configuration matching the desired state is applied. For details, refer to the specific Configuration Profile section for your profile: Section 7, VMRA Basic Configuration Profile Setup, Section 8, VMRA Build-Your-Own Configuration Profile Setup, Section 9, VMRA On-Premises Edge Configuration Profile Setup, Section 10, VMRA Remote Central Office-Forwarding Configuration Profile Setup, and Section 11, VMRA Regional Data Center Configuration Profile Setup.

For detailed steps on how to build the Ansible host, refer to 6.1.

2.5.2 Get Ansible Playbook and Prepare Configuration Templates

Perform the following steps:

1. Log in to your Ansible host (the one that you will run these Ansible playbooks from).
2. Clone the source code and change work directory.

```
git clone https://github.com/intel/container-experience-kits/
cd container-experience-kits
```

3. Check out the latest version of the playbooks – using the tag from Table 14, for example:

```
git checkout v23.02
```

Note: Alternatively go to https://github.com/intel/container-experience-kits/releases, download the latest release tarball, and unarchive it:

```
wget https://github.com/intel/container-experience-kits/archive/v23.02.tar.gz
tar xf v23.02.tar.gz
cd container-experience-kits-23.02
```

4. Initialize Git submodules to download Kubespray code.

```
git submodule update --init
```

5. Decide which Configuration Profile you want to deploy and export the environmental variable.
   For Kubernetes **Basic** Configuration Profile deployment:

```
export PROFILE=basic
```

   For Kubernetes **On-Premises Edge** Configuration Profile deployment:

```
export PROFILE=on_prem
```

   For Kubernetes **Remote Central Office-Forwarding** Configuration Profile deployment:

```
export PROFILE=remote_fp
```

   For Kubernetes **Regional Data Center** Configuration Profile deployment:

```
export PROFILE=regional_dc
```

   For Kubernetes **Build-Your-Own** Configuration Profile r deployment:

```
export PROFILE=build_your_own
```

6. Install additional requirements.

```
pip3 install -r requirements.txt
```

7. Generate example profiles.

```
make vm-profile ARCH=<skl,clx,icx,spr> NIC=<fvl,cvl> PROFILE=$PROFILE
```

2.5.3 Update Ansible Inventory File

Perform the following steps:

1. Edit the inventory.ini file generated in the previous steps.
   a. In the section [all], specify the target VM host server with hostname and Management IP address. Set ansible_user to the system user and ansible_password to match the SSH configuration of the VM host. If the server is configured with passwordless SSH the ansible_password host variable can be removed. When using a non-root user an additional parameter 'ansible_become_pass' can also be specified for the users sudo/become password.
   For more details on non-root user deployments, see Running deployment as non root user
   Note: The hostname can be the actual or a logical hostname. If a different hostname is used, be sure to update the configuration files such that host_vars/<hostname>.yml exists.
   Note: In case of multinode setup, more VM host servers need to be added to [vm_host] section and [all] section.
   b. In the [vm_host] section, update the hostname to match that defined in [all].

```
[all]
# When ansible_user is root
host-for-vms-1 ansible_host=10.0.0.1 ip=10.0.0.1 ansible_user=root ansible_password=<root password>
# When ansible_user is non-root
host-for-vms-1 ansible_host=10.0.0.1 ip=10.0.0.1 ansible_user=<user> ansible_password=<user password>
ansible_become_pass=<user sudo password>
localhost ansible_connection=local ansible_python_interpreter=/usr/bin/python3
```

```
[vm_host]
host-for-vms-1
[kube_control_plane]
#vm-ctrl-1
[etcd]
#vm-ctrl-1
[kube_node]
#vm-work-1
[k8s_cluster:children]
kube_control_plane
kube_node
[all:vars]
ansible_python_interpreter=/usr/bin/python3
```

Do not uncomment any of the hostnames defined under [kube_control_plane], [etcd] and [kube_node], as these will be dynamically updated based on the number of virtual machines defined for the target VM host server in host_vars.

2.5.4 Update Ansible Host and Group Variables

Perform the following steps.

1. Create host_vars/<hostname>.yml for the target VM host server, matching the hostname from the inventory file. The provided host_vars/host-for-vms-1.yml can be copied to simplify this process:

   ```
   cp host_vars/host-for-vms-1.yml host_vars/<hostname>.yml
   ```

   In case of multi-node setup use host_vars/host-for-vms-2.yml as a template for all other VM hosts except the first one.

2. Update "vms" in host_vars/<hostname>.yml to match the desired number of VMs and their configuration. Note the "name" and "type" assigned to each VM, as these will be used to define host variables for each VM.

   Note: For SR-IOV or QAT functionality, the VF PCI devices must be defined for each VM. This requires that the BDF (Bus:Device.Function) IDs are known prior to deploying the cluster. For more details, see the VM case configuration guide.

   Note: An optional parameter, 'vm_cluster_name' can be set to specify a custom domain name, e.g. "cluster1.local". If this parameter is used, then the host_vars files for the VMs must include the domain name as well. Example files using "cluster1.local" are provided in the host_vars folder.

3. Create host_vars/<VM_name>.yml files for all VMs of type "work" defined in the previous step. The provided host_vars/vm-work-1.yml file can be copied to simplify this process:

   ```
   cp host_vars/vm-work-1.yml host_vars/<VM_name>.yml
   ```

   Note: If 'vm_cluster_name' has been specified the filename changes:

   ```
   cp host_vars/vm-work-1.cluster1.local.yml host_vars/<VM_name>.<vm_cluster_name>.yml
   ```

4. Edit host_vars/<hostname>.yml, host_vars/<VM_name>.yml and group_vars/all.yml files to match your desired configuration. Each Configuration Profile uses its own set of variables.

Refer to the specific Configuration Profile section to get a full list of variables and their documentation: Section 7, VMRA Basic Configuration Profile Setup, Section 8, VMRA Build-Your-Own Configuration Profile Setup, Section 9, VMRA On-Premises Edge Configuration Profile Setup, Section 10, VMRA Remote Central Office-Forwarding Configuration Profile Setup, and Section 11, VMRA Regional Data Center Configuration Profile Setup.

2.5.5 Run Ansible Cluster Deployment Playbook

After the inventory and vars are configured, you can run the provided playbooks from the root directory of the project.

It is recommended that you check dependencies of components enabled in group_vars and host_vars with the packaged dependency checker:

```
# When ansible_user is root
ansible-playbook -i inventory.ini playbooks/preflight.yml
# When ansible_user is non-root
ansible-playbook -i inventory.ini playbooks/preflight.yml --become
```

Note: This will only run the dependency checker against the VM host. The check will be run against the VM configurations during deployment

Otherwise, you can skip directly to your chosen Configuration Profile playbook:

```
# When ansible_user is root
ansible-playbook -i inventory.ini playbooks/vm.yml
# When ansible_user is non-root
ansible-playbook -i inventory.ini playbooks/vm.yml --become
```

Note: The configuration profile is based on the profile_name variable from group_vars/all.yml, which was configured there while generating the templates.

After the playbook finishes without any "Failed" tasks, you can proceed with the deployment validation described in section 5 Post Deployment Verification Guidelines.

Note: Additional information can be found in the Ansible Playbook readme.

2.5.6 Expand Existing VMRA Cluster

To use the VM Cluster Expansion feature, you need to keep configuration for current cluster nodes and add configuration for new vm-work nodes on existing or on new vm_host servers. Follow the steps described in sections 2.5.3 and 2.5.4.

After the inventory and vars are updated, you can run the provided playbooks from the root directory of the project.

```
ansible-playbook -i inventory.ini playbooks/vm.yml -e scale=true
```

For detailed configuration info, see the VM cluster expansion guide.

3 Reference Architecture Hardware Components and BIOS

For all Configuration Profiles, this section provides a menu of all possible hardware components for the VM Host as well as the BIOS components available.

3.1 Hardware Component List for Host Base

The following tables list the hardware options for the host in the "base" configuration. If your configuration needs improved processing, you may choose to use the "plus" configuration instead. See the next section for details.

Table 4. Hardware Components for Host Base – 3rd Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
3rd Gen Intel Xeon Scalable processors	Intel® Xeon® Gold 5318N processor at 2.1 GHz, 24 C/48 T, 150 W, or higher number Intel® Xeon® Gold or Platinum CPU SKU	Required
Memory	Option 1: DRAM only configuration: 256 GB (8 x 32 GB DDR4, 2666 MHz)	Required
	Option 2: DRAM only configuration: 256 GB (16 x 16 GB DDR4, 2666 MHz)	
Intel® Optane™ Persistent Memory	512 GB (4x 128 GB Intel® Optane™ persistent memory in 2-1-1 Topology)	Recommended
Network Adapter	Option 1: Intel® Ethernet Network Adapter E810-CQDA2	Required
	Option 2: Intel® Ethernet Network Adapter E810-XXVDA-2	
Intel® QAT	Intel® QuickAssist Adapter 8960 or 8970 (PCIe*) AIC or equivalent third-party Intel® C620 Series Chipset	Required
Storage (Boot Drive)	Intel® SATA Solid State Drive D3 S4510 at 480 GB or equivalent boot drive	Required
Storage (Capacity)	Intel® SSD D7-P5510 Series at 3.84 TB or equivalent drive (recommended NUMA aligned)	Required
LAN on Motherboard (LOM)	10 Gbps or 25 Gbps port for Preboot Execution Environment (PXE) and Operation, Administration, and Management (OAM)	Required
	1/10 Gbps port for Management Network Adapter	Required
Additional Plug-in cards	N/A	

Table 5. Hardware Components for Host Base – 4th Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
4th Gen Intel Xeon Scalable processors	Intel® Xeon® Gold 5418N processor at 2.0GHz, 24 C/ 48 T, 165 W	Required
Memory	DRAM only configuration: 256 GB DRAM (16x 16 GB DDR5)	Required
Intel® Optane™ Persistent Memory	512 GB (4x 128 GB Intel® Optane™ persistent memory in 2-1-1 topology)	Recommended
Network Adapter	Option 1: Intel® Ethernet Network Adapter E810-CQDA2	Required
	Option 2: Intel® Ethernet Network Adapter E810-XXVDA-2	
Intel® QAT	Integrated in the processor	
Storage (Boot Drive)	Intel® SATA Solid State Drive D3 S4510 at 480 GB or equivalent boot drive	Required
Storage (Capacity)	Intel® SSD D7-P5510 Series at 3.84 TB or equivalent drive (recommended NUMA aligned)	Required
LAN on Motherboard (LOM)	10 Gbps or 25 Gbps port for Preboot Execution Environment (PXE) and Operation, Administration, and Management (OAM)	Required
	1/10 Gbps port for Management Network Adapter	Required
Additional Plug-in cards	N/A	

3.2 Hardware Component List for Host Plus

The following tables list the hardware options for the Host in the "plus" configuration, which helps improves the processing capability due to more powerful CPU, more memory, more disk space, and an amazingly fast network.

Table 6. Hardware Components for Host Plus – 3rd Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
3rd Gen Intel Xeon Scalable processors	Intel® Xeon® Gold 6338N CPU @ 2.2 GHz 32 C/64 T, 185 W, or higher number Intel® Xeon® Gold or Platinum CPU SKU	Required
Memory	Option 1: DRAM only configuration: 512 GB (16x 32 GB DDR4, 2666 MHz)	Required
	Option 2: DRAM only configuration: 512 GB (32x 16 GB DDR4, 2666 MHz)	
Intel® QAT	Intel® C620 Series Chipset integrated on base board Intel® C627/C628 Chipset, integrated with NUMA connectivity to each CPU or minimum 16 Peripheral Component Interconnect express (PCIe) lane connectivity to one CPU	Required
Intel® Optane™ Persistent Memory	Option 1: 1 TB (8x 128 GB Intel® Optane™ persistent memory in 8+4 Topology)	Recommended
	Option 2: 2 TB (16x 128 GB Intel® Optane™ persistent memory in 8+8 Topology)	
Network Adapter	Option 1: Intel® Ethernet Network Adapter E810-CQDA2	Required
	Option 2: Intel® Ethernet Network Adapter E810-2CQDA2	
Storage (Boot Drive)	Intel® SATA Solid State Drive D3 S4510 at 480 GB or equivalent boot drive	Required
Storage (Capacity)	Intel® SSD D7-P5510 Series at 4 TB or equivalent drive (recommended NUMA aligned)	Recommended
LAN on Motherboard (LOM)	10 Gbps or 25 Gbps port for Preboot Execution Environment (PXE) and Operation, Administration, and Management (OAM)	Required
	1/10 Gbps port for Management Network Adapter	Required
Additional Plug-in cards	Intel® Server Graphics 1 card	Optional

Table 7. Hardware Components for Host Plus – 4th Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
4th Gen Intel Xeon Scalable processors	Intel® Xeon® Gold 6438N processor at 1.8GHz, 32 C/64 T, 205 W	Required
Memory	Option 1: DRAM only configuration: 512 GB (16x 32 GB DDR5)	Required
	Option 2: DRAM only configuration: 512 GB (32x 16 GB DDR5)	
Intel® QAT	Integrated in the processor	Required
Intel® Optane™ Persistent Memory	Option 1: 1 TB (8x 128 GB Intel® Optane™ persistent memory in 8+4 Topology)	Recommended
	Option 2: 2 TB (16x 128 GB Intel® Optane™ persistent memory in 8+8 Topology)	
Network Adapter	Option 1: Intel® Ethernet Network Adapter E810-CQDA2	Required
	Option 2: Intel® Ethernet Network Adapter E810-2CQDA2	
Storage (Boot Drive)	Intel® SATA Solid State Drive D3 S4510 at 480 GB or equivalent boot drive	Required
Storage (Capacity)	Intel® SSD D7-P5510 Series at 4 TB or equivalent drive (recommended NUMA aligned)	Recommended
LAN on Motherboard (LOM)	10 Gbps or 25 Gbps port for Preboot Execution Environment (PXE) and Operation, Administration, and Management (OAM)	Required
	1/10 Gbps port for Management Network Adapter	Required
Additional Plug-in cards	Intel® Server Graphics 1 card	Optional

3.3 Hardware BOMs for all VMRA Configuration Profiles

The following tables list the hardware BOMs for host base and plus.

The profiles for hosts vary with respect to network interface card, Intel® QuickAssist Technology, and BIOS profiles. You may choose based on the requirement for the workloads to be run in the cluster.

Table 8. Host Base Hardware Setup for all Configuration Profiles – 3rd Gen Intel Xeon Scalable Processor

NAME	Host_3rdGen_Base_1	Host_3rdGen_Base_2	Host_3rdGen_Base_3
Platform	M50CYP	M50CYP	M50CYP
CPU/node	2x 5318N 24c	2x 5318N 24c	2x 5318N 24c
Mem	512 GB	512 GB	512 GB
Intel Optane Persistent Memory	Recommended – 512 GB	Recommended – 512 GB	Recommended – 512 GB
Network Adapter	2x E810-CQDA2	2x E810-CQDA2	2x E810-2CQDA2 or 4x E810-CQDA2
Storage (Boot Media)	Required - 2x	Required - 2x	Required - 2x
Storage (Capacity)	Required- 2x (1 per NUMA)	Required- 2x (1 per NUMA)	Required- 2x (1 per NUMA)
LOM	No	Yes	No
Intel® QAT	No	Yes	Optional
Additional Plug-in cards	No	No	No
BIOS Configuration			
Intel® HT Technology enabled	Yes	Yes	Yes
Intel® VT-x enabled	Yes	Yes	Yes
Intel® VT-d enabled	Yes	Yes	Yes
BIOS Profile	Energy Balance	Max Performance	Deterministic
Virtualization enabled	Yes	Yes	Yes

Table 9. Host Plus Hardware Setup for all Configuration Profiles – 3rd Gen Intel Xeon Scalable Processor

NAME	Host_3rdGen_Plus_1	Host_3rdGen_Plus_2	Host_3rdGen_Plus_3
Platform	M50CYP	M50CYP	M50CYP
CPU/node	2x 6338N 32c	2x 6338N 32c	2x 6338N 32c
Mem	512 GB	512 GB	512 GB
Intel Optane Persistent Memory	Recommended – 512 GB	Recommended – 512 GB	Recommended – 512 GB
Network Adapter	2x E810-2CQDA2 or 4x E810-CQDA2	2x E810-2CQDA2	2x E810-2CQDA2 or 4x E810-CQDA2
Storage (Boot Media)	Required - 2x	Required - 2x	Required - 2x
Storage (Capacity)	Required- 4x (2 per NUMA)	Required- 4x (2 per NUMA)	Required- 4x (2 per NUMA)
LOM	Yes	Yes	No
Intel® QAT	Yes	No	Optional
Additional Plug-in cards	No	Intel Server GPU	No
BIOS Configuration			
Intel® HT Technology enabled	Yes	Yes	Yes
Intel® VT-x enabled	Yes	Yes	Yes
Intel® VT-d enabled	Yes	Yes	Yes

NAME	Host_3rdGen_Plus_1	Host_3rdGen_Plus_2	Host_3rdGen_Plus_3
BIOS Profile	Max Performance	Max Performance	Deterministic
Virtualization enabled	Yes	Yes	Yes

Table 10. Host Base Hardware Setup for all Configuration Profiles – 4th Gen Intel Xeon Scalable Processor

NAME	Host_4thGen_Base_1	Host_4thGen_Base_2
Platform	Archer City / Quanta - S6Q	Archer City / Quanta - S6Q
CPU/node	2x 5418N	2x 5418N
Mem	512 GB	512 GB
Intel Optane Persistent Memory	Recommended – 512 GB	Recommended – 512 GB
Network Adapter	2x E810-CQDA2	2x E810-CQDA2
Storage (Boot Media)	Required - 2x	Required - 2x
Storage (Capacity)	Required- 2x (1 per NUMA)	Required- 2x (1 per NUMA)
LOM	No	Yes
Intel® QAT	Integrated in the processor	Integrated in the processor
Additional Plug-in cards	No	No
BIOS Configuration		
Intel® HT Technology enabled	Yes	Yes
Intel® VT-x enabled	Yes	Yes
Intel® VT-d enabled	Yes	Yes
BIOS Profile	Energy Efficiency Turbo	Max Performance Turbo
Virtualization Enable	Yes	Yes

Table 11. Host Plus Hardware Setup for all Configuration Profiles – 4th Gen Intel Xeon Scalable Processor

NAME	Host_4thGen_Plus_1	Host_4thGen_Plus_2	Host_4thGen_Plus_3
Platform	Archer City / Quanta - S6Q	Archer City / Quanta - S6Q	Archer City / Quanta - S6Q
CPU/node	2x 6438N	2x 6438N	2x 6438N
Mem	512 GB	512 GB	512 GB
Intel Optane Persistent Memory	Recommended – 2 TB	Recommended – 1 TB	Recommended – 2 TB
Network Adapter	2x E810-2CQDA2 or 4x E810-CQDA2	2x E810-2CQDA2 or 8x E810 XXVAM-DA4	2x E810-2CQDA2 or 4x E810-CQDA2
Storage (Boot Media)	Required - 2x	Required - 2x	Required - 2x
Storage (Capacity)	Required- 4x (2 per NUMA)	Required- 4x (2 per NUMA)	Required- 4x (2 per NUMA)
LOM	Yes	Yes	No

NAME	Host_4thGen_Plus_1	Host_4thGen_Plus_2	Host_4thGen_Plus_3
Intel® QAT	Integrated in the processor	Integrated in the processor	Integrated in the processor
Additional Plug-in cards	No	Intel Server GPU	No
BIOS Configuration			
Intel® HT Technology enabled	Yes	Yes	Yes
Intel® VT-x enabled	Yes	Yes	Yes
Intel® VT-d enabled	Yes	Yes	Yes
BIOS Profile	Energy Balance Turbo	Energy Balance Turbo	Max Performance Turbo
Virtualization enabled	Yes	Yes	Yes

3.4 Platform BIOS Profiles Settings

This section provides BIOS configuration profiles for each of the VMRA Configuration Profiles. For details on how the BIOS configuration should be set per each Configuration Profile, go to tables in section 3.3.

For more information about BIOS settings, visit https://www.intel.com/content/dam/support/us/en/documents/server-products/Intel_Xeon_Processor_Scalable_Family_BIOS_User_Guide.pdf.

Table 12. Platform BIOS Profile settings for 3rd Gen Intel® Xeon® Scalable Processor

MENU (Advanced)	Path to BIOS Setting	BIOS Setting	Energy Balance	Max Performance with Turbo	Deterministic
Socket Configuration	Processor Configuration	Hyper-Threading	Enable	Enable	Enable
		XAPIC	Enable	Enable	Enable
		VMX	Enable	Enable	Enable
		Uncore frequency scaling	Enable	Enable	Disable
		Uncore frequency	800-2400	800-2400	2400
Power Configuration	Power and Performance	CPU Power and Performance Policy	Balance Performance	Performance	Performance
		Workload Configuration	I/O sensitive	I/O sensitive	I/O sensitive
	CPU P State Control	EIST PSD Function	HW_ALL	HW_ALL	HW_ALL
		Boot Performance Mode	Max. Performance	Max. Performance	Max. Performance
		AVX License Pre-Grant	Disable	Disable	Disable
		AVX ICCP Pre Grant Level	NA	NA	NA
		AVX P1	Nominal	Nominal	Nominal
		Energy Efficient Turbo	Enable	Enable	Disable
		WFR Uncore GV rate Reduction	Enable	Enable	Enable
		GPSS timer	500us	0us	0us
		Intel Turbo Boost Technology	Enable	Enable	Disable
		Intel SpeedStep® Technology (P-states)	Enable	Enable	Disable
	Frequency Prioritization	RAPL Prioritization	Enable	Disable	Disable

	Hardware PM State Control	Hardware P-States	Native Mode with no legacy Support	Disable	Disable
		EPP enable	Enable	Disable	Disable
	CPU C State Control	Enable Monitor Mwait	Enable	Enable	Enable
		CPU C1 Auto Demotion	Enable	Disable	Disable
		CPU C1 Auto unDemotion	Enable	Disable	Disable
		CPU C6 Report	Enable	Enable	Disable
		Processor C6	Enable	Enable	Disable
		Enhanced Halt State (C1E)	Enable	Enable	Disable
		OS ACPI Cx	ACPI C2	ACPI C2	ACPI C2
	Energy Performance Bias	Power Performance Tuning	OS Controls EPB	OS Controls EPB	OS Controls EPB
		ENERGY_PERF_BIAS_CFG mode	Performance	Performance	Performance
		Workload Configuration	I/O Sensitive	I/O Sensitive	I/O Sensitive
	Package C State Control	Package C State	C6 Retention	C6 Retention	C0/C1 State
		Dynamic L1	Enable	Disable	Disable
		Package C-state Latency Negotiation	Disable	Disable	Disable
		PKGC_SA_PS_CRITERIA	Disable	Disable	Disable
Memory Configuration		Memory Configuration	2-way interleave	2-way interleave	2-way interleave
		Enforce POR	Enable	Enable	Enable
Platform Configuration	Miscellaneous Configuration	Serial Debug Message Level	Minimum	Minimum	Minimum
	PCI Express* Configuration	PCIe* ASPM Support	Per Port	Per Port	Per Port
	PCI Express* Configuration	PCIe* ASPM	Enable	Disable	Disable
	PCI Express* Configuration	ECRC generation and checking	Enable	Enable	Enable
Server Management		Resume on AC Power Loss	Power On	Power On	Power On
System Acoustic and Performance Configuration		Set Fan Profile	Acoustic	Performance	Performance

Table 13. Platform BIOS Profile settings for 4th Gen Intel® Xeon® Scalable Processor

MENU (Advanced)	Path to BIOS Setting	BIOS Setting	Max Performance with Turbo	Energy Balance Turbo
Socket Configuration	Processor Configuration	Hyper-Threading	Enable	Enable
		X2APIC	Enable	Enable
		VMX	Enable	Enable
		LLC Prefetch	Enable	Enable
		SNC	Disable	Disable
		Uncore RAPL	Disable	Enable
		Uncore frequency scaling	Disable	Enable
		Uncore frequency	1.6MHz (hex 0x10)	800MHz to 2.5GHz
Power Configuration	CPU P-state Control	EIST PSD Function	HW_ALL	HW_ALL

		Boot Performance Mode	Max. Performance	Max. Performance
		AVX License Pre-Grant	Disable	Disable
		AVX ICCP Pre Grant Level	NA	NA
		AVX P1 (ConfigTDP)	Nominal (default)	Nominal
		Energy Efficient Turbo	Disable	Enable
		GPSS timer	0us	0us
		Turbo	Enable	Enable
		Intel® SpeedStep®(Pstates) Technology	Enable	Enable
	Frequency Prioritization	RAPL Prioritization	Disable	Disable
	Common Ref Code	UMA-Based Clustering	Quadrant	Quadrant
	Hardware PM State Control	Hardware P-states	Native with no Legacy Support	Native with no Legacy Support
		EPP enable	Disable	Disable
	CPU C-state Control	Enable Monitor Mwait	Enable	Enable
		CPU C1 Auto Demotion	Disable	Disable
		CPU C1 Auto unDemotion	Disable	Disable
		Processor C6 or CPU C6 Report	Enable	Enable
		Enhanced Halt State (C1E)	Enable	Enable
		OS ACPI Cx	ACPI C2	ACPI C2
	Energy Performance Bias	Power Performance Tuning	OS Controls EPB	OS Controls EPB
		Workload Configuration	I/O Sensitive	Balanced
	Package C-state Control	Package C-state	C0/C1 State	C0/C1 State
		Dynamic L1	Disable	Disable
Memory Configuration		Memory Configuration	8-way interleave	8-way interleave
		Enforce POR / Memory Patrol Scrub	Enable/Enable	Enable/Enable
		Memory DIMM Refresh Rate	1x	2x
Platform Configuration	Miscellaneous Configuration	Serial Debug Message Level	Minimum	Minimum
	PCI Express* Configuration	PCIe* ASPM	Enable	Enable
		ECRC generation and checking	Enable	Enable
Server Management		Resume on AC Power Loss	Power On	Power On
System Acoustic and Performance Configuration		Set Fan Profile	Acoustic	Acoustic

4 Reference Architecture Software Components

4.1 Software Components Supported

Table 14 lists all the software components automatically deployed for generating VMRA Flavors per Configuration Profile specifications and their sources.

Table 14. Software Components

SOFTWARE FUNCTION	SOFTWARE COMPONENT	LOCATION
OS	Ubuntu 22.04	https://www.ubuntu.com
Data Plane Development Kit (DPDK)	22.11.1	https://core.dpdk.org/download/
Open vSwitch with DPDK	3.0.3	https://github.com/openvswitch/ovs
Vector Packet Processing (VPP)	23.02	https://github.com/FDio/vpp
Telegraf	1.2	https://github.com/intel/observability-telegraf
Collectd	1.0	https://github.com/intel/observability-collectd/tags
Collectd Exporter	0.5.0	https://github.com/prometheus/collectd_exporter/tags
OpenTelemetry	0.24.0	https://github.com/open-telemetry/opentelemetry-operator
Jaeger	1.42.0	https://github.com/jaegertracing/jaeger-operator
Grafana	9.3.6	https://www.grafana.com/
Prometheus	2.42.0	https://quay.io/prometheus/prometheus
cAdvisor	2.2.4	https://artifacthub.io/packages/helm/ckotzbauer/cadvisor/
Ansible	Ansible: 5.7.1 Ansible-core: 2.12.5	https://www.ansible.com/
VMRA Ansible Playbook	v23.02	https://github.com/intel/container-experience-kits
Python	Python 3.10.4 for Ubuntu 22.04	https://www.python.org/
Kubespray	d81978625c7548eba10c4e4d455b3996235e6b91	https://github.com/kubernetes-sigs/kubespray
etcd	3.5.6	https://github.com/etcd-io/etcd/tags
Docker	20.10.20	https://www.docker.com/
Containerd	1.6.16	https://github.com/containerd/containerd/tags
CRI-O	1.26.0	https://github.com/cri-o/cri-o/tags
Container orchestration engine	Kubernetes v1.26.1 Kubernetes v1.25.6 Kubernetes v1.24.10	https://github.com/kubernetes/kubernetes
Platform Aware Scheduling (TAS)	0.4	https://github.com/intel/platform-aware-scheduling
Platform Aware Scheduling (GAS)	0.5.1	https://github.com/intel/platform-aware-scheduling
Prometheus	2.42.0	https://quay.io/repository/prometheus/prometheus?tab=tags
Prometheus node-exporter	1.5.0	https://quay.io/repository/prometheus/node-exporter?tab=tags
Prometheus Operator	0.63.0	https://quay.io/repository/prometheus-operator/prometheus-operator?tab=tags
Kubernetes RBAC Proxy	0.14.0	https://github.com/brancz/kube-rbac-proxy/tags
Container Registry	Registry: 2.8.1	https://github.com/distribution/distribution/tags
	Nginx: 1.23.3-alpine	https://github.com/docker-library/docs/tree/master/nginx
Node Feature Discovery	0.12.1-minimal	https://github.com/kubernetes-sigs/node-feature-discovery

SOFTWARE FUNCTION	SOFTWARE COMPONENT	LOCATION
Multus CNI	3.9.3	https://github.com/k8snetworkplumbingwg/multus-cni/tags
SR-IOV CNI	2.7.0	https://github.com/intel/sriov-cni
SR-IOV network device plugin	3.5.1	https://github.com/intel/sriov-network-device-plugin
SR-IOV Network Operator	1.2.0	https://github.com/k8snetworkplumbingwg/sriov-network-operator
Device Plugins Operator	0.26.0	https://github.com/intel/intel-device-plugins-for-kubernetes
QAT device plugin	0.26.0	https://github.com/intel/intel-device-plugins-for-kubernetes
GPU device plugin	0.26.0	https://github.com/intel/intel-device-plugins-for-kubernetes
Intel® SGX device plugin	0.26.0	https://github.com/intel/intel-device-plugins-for-kubernetes
Userspace CNI	1.3	https://github.com/intel/userspace-cni-network-plugin
Bond CNI plugin	9800813	https://github.com/intel/bond-cni
Intel® Ethernet Drivers	i40e v2.22.8	https://sourceforge.net/projects/e1000/files/i40e%20stable
	ice v1.10.1.2.2	https://sourceforge.net/projects/e1000/files/ice%20stable/
	iavf v4.7.0	https://sourceforge.net/projects/e1000/files/iavf%20stable/
Intel® Ethernet NVM Update Package for Intel Ethernet 700 Series	9.20	https://www.intel.com/content/www/us/en/download/18635/non-volatile-memory-nvm-update-utility-for-intel-ethernet-adapters-700-series-linux.html
Intel® Ethernet NVM Update Package for Intel Ethernet 800 Series	4.20	https://www.intel.com/content/www/us/en/download/19626/non-volatile-memory-nvm-update-utility-for-intel-ethernet-network-adapters-e810-series-linux.html
Intel® Ethernet Operator	22.11	https://github.com/intel/intel-ethernet-operator/tags
Intel Unified Flow Tool	22.11	https://github.com/intel/UFT/tags
Operator SDK	1.26.0	https://github.com/operator-framework/operator-sdk/tags
Operator Lifecycle Manager (OLM)	0.22.0	https://github.com/operator-framework/operator-lifecycle-manager/tags
Dynamic Device Personalization for Intel® Ethernet 700 Series	Version 25.4	https://downloadmirror.intel.com/27587/eng/gtp.zip https://downloadmirror.intel.com/28940/eng/mplsogreudp.zip https://downloadmirror.intel.com/28040/eng/ppp-oe-ol2tpv2.zip https://downloadmirror.intel.com/29446/eng/esp-ah.zip https://downloadmirror.intel.com/29780/eng/ecpri.zip
Intel® Ethernet 800 Series Dynamic Device Personalization (DDP) for Telecommunication (Comms) Package	1.3.37.0	https://www.intel.com/content/www/us/en/download/19660/intel-ethernet-800-series-dynamic-device-personalization-ddp-for-telecommunication-comms-package.html
Intel® QAT Drivers	(HW 2.0) QAT20.L.1.0.0-00021	https://www.intel.com/content/www/us/en/download/765501/intel-quickassist-technology-driver-for-linux-hw-version-2-0.html
	(HW 1.7) QAT.L.4.20.0-00001	https://www.intel.com/content/www/us/en/download/19734/intel-quickassist-technology-driver-for-linux-hw-version-1-7.html
OpenSSL	openssl-3.0.8	https://github.com/openssl/openssl/tags
OpenSSL QAT Engine	0.6.18	https://github.com/intel/QAT_Engine/tags
Intel QATLib	23.02.0	https://github.com/intel/qatlib/tags
Intel® Multi-Buffer Crypto for IPsec Library	1.3	https://github.com/intel/intel-ipsec-mb/tags
Intel® SGX DCAP Drivers	1.41	https://download.01.org/intel-sgx/sgx-dcap/1.15/linux/distro/
Intel® SGX SDK	2.18.100.3	https://download.01.org/intel-sgx/sgx-dcap/1.15/linux/distro/
Intel® SGX packages	2.18.100.3	https://download.01.org/intel-sgx/sgx_repo/ubuntu/dists/jammy/main/binary-amd64/Packages

SOFTWARE FUNCTION	SOFTWARE COMPONENT	LOCATION
Intel® SGX DCAP packages	1.15.100.3	https://download.01.org/intel-sgx/sgx_repo/ubuntu/dists/jammy/main/binary-amd64/Packages
Intel KMRA	2.3	https://01.org/key-management-reference-application-kmra
Istio Service Mesh	1.17.1	https://github.com/istio/istio/tags
Intel Managed Distribution of Istio Service Mesh	1.16.1-intel.0	https://github.com/intel/istio/tags
Trusted Attestation Controller (TAC)	0.4.0	https://github.com/intel/trusted-attestation-controller/tags
Trusted Certificate Service for Kubernetes platform	0.4.0	https://github.com/intel/trusted-certificate-issuer/tags
Cloud Native Data Plane (CNDP) Device Plugin	0.0.2	https://github.com/intel/afxdp-plugins-for-kubernetes/tags
CNDP CNI	22.08.0	https://github.com/CloudNativeDataPlane/cndp/tags
Go Programming Language	1.19.3	https://go.dev/dl/
QEMU	Latest (7.0, 6.2)	https://www.qemu.org/
Libvirt (SGX)	Custom (8.7.0)	https://github.com/hhb584520/libvirt
Virt-manager	Latest (4.0.0-1)	https://virt-manager.org/
Linkerd	stable-2.12.4	https://github.com/linkerd/linkerd2/releases

5 Post-Deployment Verification Guidelines

This section describes a set of processes that you can use to verify the components deployed by the scripts. The processes are not Configuration Profile-specific but relate to individual components that may not be available in all profiles. Details for each of the Configuration Profiles are described in Sections 7 through 11.

The VMs can be accessed from the Ansible Host. Start by changing to the root user. If the name of the VMs has not been changed, they can be accessed directly through SSH:

```
$ ssh vm-ctrl-1
$ ssh vm-work-1
```

Note: If different VM names have been specified, the above commands should use the updated names.

In the following sections, whenever "kubectl" is used it is assumed that you are connected to one of the controller nodes.

Verification guidelines and output examples can be found on GitHub, as listed in Table 15.

Table 15. Links to Verification Guidelines on GitHub

VERIFICATION STEP
Check the Kubernetes Cluster
Check DDP Profiles on Intel® Ethernet 700 and 800 Series Network Adapters
Check Node Feature Discovery
Check Topology Manager
Check SR-IOV Device Plugin
Check QAT Device Plugin
Check Multus CNI Plugin
Check SR-IOV CNI Plugin
Check Userspace CNI Plugin
Check Telemetry Aware Scheduling
Check Intel QAT Engine with OpenSSL

5.1 Check Grafana Telemetry Visualization

VMRA deploys Grafana for telemetry visualization. It is available on every cluster node on port 30000. Due to security reasons, this port is not exposed outside the cluster by default. Default credentials are admin/admin and you should change the default password after first login.

The Grafana TLS certificate is signed by the cluster CA and it is available in /etc/kubernetes/ssl/ca.crt

As the VMs use an internal network, port forwarding must be configured before Grafana is accessible. From the Ansible host, as the root user, run the following command to set up forwarding:

```
$ ssh -L <Ansible Host IP>:30000:localhost:30000 vm-ctrl-1
```

Note: If the VM names have been changed, replace "vm-ctrl-1" with the updated name.

Note: If there are additional jumps between your machine and the Ansible Host, it might be necessary to configure additional forwarding or proxies. These steps will depend on your local setup.

Visit Grafana at https://<Ansible Host IP>:30000/

VMRA comes with a set of dashboards from the kube-prometheus project (https://github.com/prometheus-operator/kube-prometheus). Dashboards are available in the Dashboards -> Manage menu.

Part 2:
Building a VMRA Step-by-Step

6 VMRA Setup – Applicable for All Configuration Profiles

This section is relevant for generating VMRA Flavors based on their Configuration Profiles. It provides the prerequisites for a system setup and includes information that enables you to review BIOS prerequisites and software BOMs at a glance. The information is presented in multi-column tables to provide an easy way to compare and assess the differences between the VMRA Flavors that are available.

After setting up the Kubernetes system, refer to the specific section from the following list to build the Configuration Profile Flavor:

Section 7, VMRA Basic Configuration Profile Setup
Section 8, VMRA Build-Your-Own Configuration Profile Setup
Section 9, VMRA On-Premises Edge Configuration Profile Setup
Section 10, VMRA Remote Central Office-Forwarding Configuration Profile Setup
Section 11, VMRA Regional Data Center Configuration Profile Setup

Note: The taxonomy for the VMRA Configuration Profile settings is defined in section 1.4.

Note: Not all features supported in Container Bare Metal Reference Architecture are supported in Virtual Machine Reference Architecture due to the nature of virtualization and abstraction of hardware.

6.1 Set Up an Ansible Host

VMRA Kubernetes clusters require an Ansible host that stores information about all managed remote nodes. In general, any machine running a recent Linux distribution can be used as Ansible host for any of the supported VMRA deployments, as long as it meets the following basic requirements:

- Network connectivity to the VM host server, including SSH
- Internet connection (using Proxy if necessary)
- Git utility installed
- Python 3 installed

Step-by-step instructions for building the Ansible host are provided below for the same list of operating systems that are supported for the VM host server (see 2.3.3).

6.1.1 Rocky 9 as Ansible Host

1. Install the Linux OS using any method supported by the vendor. If using the iso image, choose the Minimal iso version, or select the "Minimal Install" (Basic functionality) option under Software Selection.
2. Make the proper configuration during installation for the following key elements: Network (Ethernet) port(s) IP Address; Host Name, Proxies (if necessary), and Network Time Protocol (NTP).
3. After the installation completes and the machine reboots, log in as root, and confirm that it has a valid IP address and can connect (ping) to the VM host.
4. Make sure that the http and https proxies are set, if necessary, for internet access. The configuration can be completed with the *export* command or by including the following lines in the /etc/environment file:

   ```
   http_proxy=http://proxy.example.com:1080
   https_proxy=http://proxy.example.com:1080
   ```

   Then, load the proxy configuration in the current environment:

   ```
   # source /etc/environment
   ```

5. Install Git:

   ```
   # dnf install -y git
   ```

6. Install Python 3:

   ```
   # dnf -y install python3
   ```

The Ansible host box is now ready to deploy the VMRA. Follow the instructions in 2.5.

6.1.2 Ubuntu 20.04 LTS as Ansible Host

1. Install the OS using any method supported by the vendor. Either the Desktop or Server distribution can be used. Select the “Minimal installation” option under “Updates and Other software”.
2. Follow steps 2, 3, and 4 as described above for Rocky.
3. Update the installation:

   ```
   # sudo apt update
   ```

4. Install SSH utilities:

```
# sudo apt install openssh-server
```

5. Install Git:

```
# sudo apt install -y git
```

6. Install Python 3-pip:

```
# sudo apt install -y python3-pip
```

The Ansible host box is now ready to deploy the VMRA. Follow the instructions in section 2.5.

6.2 Set Up the VM Host – BIOS and HW Prerequisites

This section is applicable for all **VMRA Configuration Profiles.**

Each of the reference architecture configuration profiles are aligned with one or more of the BIOS profiles listed in Platform BIOS Profiles Settings.

For each BIOS profile, there is a set of hardware configurations that apply. These configurations are described in Hardware BOMs for all VMRA Configuration Profiles.

Combining this information, it is possible to create a list of hardware configurations relevant for each of the reference architecture configuration profiles. This information can be seen in

Table 16. Hardware configurations per profile

REFERENCE ARCHITECTURE PROFILE:	BASIC	BUILD-YOUR-OWN	ON-PREMISES EDGE	REMOTE CENTRAL OFFICE-FORWARDING	REGIONAL DATA CENTER
BIOS Profile:	Energy Balance	Any	Max Performance	Deterministic / Energy Balance	Max Performance
VM Host 3rd Gen Intel® Xeon® Scalable Processor	3rdGen_Base_1	3rdGen_Base_1 3rdGen_Base_2 3rdGen_Base_3 3rdGen_Plus_1 3rdGen_Plus_2 3rdGen_Plus_3	3rdGen_Base_2 3rdGen_Plus_1 3rdGen_Plus_2	3rdGen_Base_1 3rdGen_Base_3 3rdGen_Plus_3	3rdGen_Base_2 3rdGen_Plus_1 3rdGen_Plus_2
VM Host 4th Gen Intel® Xeon® Scalable Processor	4thGen_Plus_1 4thGen_Plus_2	4thGen_Base_1 4thGen_Base_2 4thGen_Plus_1 4thGen_Plus_2 4thGen_Plus_3	4thGen_Base_2 4thGen_Plus_3	4thGen_Plus_1 4thGen_Plus_2	4thGen_Base_2 4thGen_Plus_3

Additional details about BIOS configuration and hardware components can be found in Reference Architecture Hardware Components and BIOS.

6.3 Configuration Dictionary - Group Variables

Table 17 lists the parameters available as group variables with their type (for example, Boolean, string, list, integer), possible values, and descriptions. Refer to the section that describes your Configuration Profile to see the parameters enabled for that Configuration Profile.

Table 17. Configuration Dictionary – Group Variables for VMRA

OPTION	TYPE
profile_name	String
configured_arch	String
unconfirmed_cpu_models	List
project_root_dir	String

OPTION	TYPE
vm_enabled	Boolean
on_cloud	Boolean
post_deployment_hook_enabled	Boolean
hooks_local	String
hooks_remote	String
kubernetes	Boolean
kube_version	String
container_runtime_only_deployment	Boolean
audit_policy_custom_rules	String
container_runtime	String
intel_cpu_controlplane.enabled	Boolean
intel_cpu_controlplane.allocator	String
intel_cpu_controlplane.agent_namespace_prefix	String
cadvisor_enabled	Boolean
cadvisor_custom_events_config_on	Boolean
preflight_enabled	Boolean
update_all_packages	Boolean
update_kernel	Boolean
additional_grub_parameters_enabled	Boolean
additional_grub_parameters	String
selinux_state	String
nfd_enabled	Boolean
nfd_namespace	String
nfd_sleep_interval	String
sriov_network_operator_enabled	Boolean
istio_service_mesh.enabled	Boolean
istio_service_mesh.profile	String
istio_service_mesh.intel_preview.enabled	Boolean
istio_service_mesh.profile.tls_splicing.enabled	Boolean
linkerd_service_mesh.enabled	Boolean
prometheus_operator	Boolean
telegraf_enabled	Boolean
jaeger_operator	Boolean
opentelemetry_enabled	Boolean
elasticsearch_enabled	Boolean
kibana_enabled	Boolean
collectd_scrap_interval	Integer
telegraf_scrap_interval	Integer
example_net_attach_defs.userspace_ovs_dpdk	Boolean
example_net_attach_defs.userspace_vpp	Boolean
http_proxy (Commented)	String
https_proxy (Commented)	String
additional_no_proxy (Commented)	String
dns_disable_stub_listener	Boolean
remove_kubespray_host_dns_settings	Boolean
cluster_name	String

OPTION	TYPE
retry_stagger	Integer
cert_manager_enabled	Boolean
kube_controller_manager_bind_address	String
kube_proxy_metrics_bind_address	String
kube_network_plugin	String
calico_network_backend	String
calico_advanced_options	Boolean
wireguard_enabled	Boolean
kube_network_plugin_multus	Boolean
kube_pods_subnet	String
kube_service_addresses	String
kube_proxy_mode	String
calico_bpf_enabled	Boolean
kube_proxy_nodeport_addresses_cidr	String
docker_registry_mirrors (Commented)	List
docker_insecure_registries (Commented)	List
containerd_registries (Commented)	List
crio_registries (Commented)	List
crio_insecure_registries (Commented)	List
always_pull_enabled	Boolean
tadk_install	Boolean

6.4 Configuration Dictionary - Host Variables

Table 18 lists the parameters available as host variables with their type (for example, Boolean, string, list, integer), possible values, and descriptions. Refer to the section that describes your Configuration Profile to see the parameters enabled for that Configuration Profile.

Table 18. Configuration Dictionary - Host Variables for VMRA

OPTION	TYPE
profile_name	String
configured_arch	String
configured_nic	String
dataplane_interfaces	List
update_nic_firmware	Boolean
nvmupdate (Commented)	List
install_ddp_packages	Boolean
install_dpdk	Boolean
dpdk_version	String
dpdk_local_patches_dir (Commented)	String
dpdk_local_patches_strip (Commented)	Integer
userspace_cni_enabled	Boolean
ovs_dpdk_enabled	Boolean
ovs_version	String
ovs_dpdk_lcore_mask	String
ovs_dpdk_socket_mem	String
vpp_enabled	Boolean
hugepages_enabled	Boolean
default_hugepage_size	String
number_of_hugepages_1G	Integer
number_of_hugepages_2M	Integer
isolcpus_enabled	Boolean
isolcpus	String
cpusets_enabled	Boolean
cpusets	String
cstate_enabled	Boolean
cstates.C<1,6>.cpu_range	String
cstates.C<1,6>.enable	Boolean

6.5 Taxonomy for Configuration Profiles

In sections 7 through 11, the following definitions of terminology are used.

TERM	DESCRIPTION
Hardware Taxonomy	
ENABLED	Setting must be enabled in the BIOS (configured as Enabled, Yes, True, or similar value)
DISABLED	Setting must be disabled in the BIOS (configured as Disabled, No, False, or any other value with this meaning.)
OPTIONAL	Setting can be either disabled or enabled, depending on user's workload. Setting does not affect the Configuration Profile or platform deployment.
Software Taxonomy	
TRUE	Feature is included and enabled by default.
FALSE	Feature is included but disabled by default - can be enabled and configured by user.
N/A	Feature is not included and cannot be enabled or configured.

7 VMRA Basic Configuration Profile Setup

This section contains a step-by-step description of how to set up a VMRA Basic Configuration Profile.

To use the Basic Configuration Profile, perform the following steps:

1. Choose your hardware, set it up, and configure the BIOS. Refer to 6.2 for details.
2. Download and configure the Ansible playbook for your Configuration Profile. Refer to 7.2 for details.
3. Set up the optional Ansible parameters using the information in the Configuration Profile tables. Refer to 7.3 for details.
4. Deploy the platform. Refer to 7.4 for details.
5. Validate the setup of your Kubernetes cluster. Refer to 5 for details.

7.1 Step 1 - Set Up Basic Configuration Profile Hardware

Use the information in Set Up the VM Host – BIOS and HW Prerequisites to determine the hardware and BIOS configuration needed for this profile.

7.2 Step 2 - Download Basic Configuration Profile Ansible Playbook

This section contains details for downloading the Basic Configuration Profile Ansible playbook. It also provides an overview of the Ansible playbook and lists the software that is automatically installed when the playbook is deployed.

Download the Basic Configuration Profile Ansible playbook using the steps described in 2.5.

7.2.1 Basic Configuration Profile Ansible Playbook Overview

The Ansible playbook for the Basic Configuration Profile allows you to provision a production-ready virtual infrastructure with or without a Kubernetes cluster. Every capability included in the Basic Configuration Profile playbook can be disabled or enabled. Refer to Figure 3 and the group and host variables tables below.

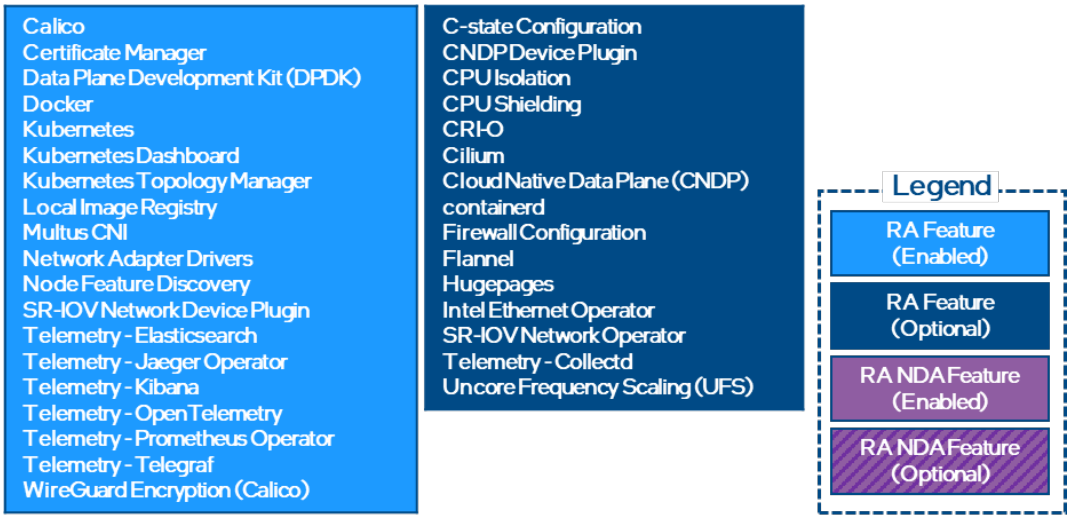

Figure 3. Basic Configuration Profile Ansible Playbook

7.3 Step 3 - Set Up Basic Configuration Profile

Review the optional Ansible group and host variables in this section and select options that match your desired configuration.

1. Update the inventory.ini file with your environment details as described in 2.5.3.
2. Create host_vars/<hostname>.yml for the target VM host server and all VMs of type "work" as specified in 2.5.4.
3. Update group and host variables to match your desired configuration as specified in 2.5.4.

Variables are grouped into two main categories:

1. Group variables – apply to both control and worker nodes and have cluster-wide impact.
1. Host variables – their scope is limited to a single worker node or the VM host.

The tables below are a summary of group and host variables. For lists showing all configurable properties, see 6.3 and 6.4. All of the variables are important but pay special attention to variables in **bold** as they almost always need to be updated to match the target environment.

7.3.1 Basic Configuration Profile Group Variables

Table 19. Basic Configuration Profile – Group Variables

COMPONENT	VALUE	
Kubernetes	true	For the list of all configurable properties, see Section 6.3
nfd_enabled	true	
topology_manager_enabled	true	
sriov_network_operator_enabled	false	
sriov_net_dp_enabled	false	
example_net_attach_defs, sriov_net_dp	false	

7.3.2 Basic Configuration Profile Host Variables[1]

Table 20. Basic Configuration Profile – Host Variables

COMPONENT	VALUE	
iommu_enabled	false	For the list of all configurable properties, see Section 6.4
sriov_cni_enabled	false	
isolcpus_enabled	false	
dns_disable_stub_listener	true	

7.4 Step 4 - Deploy Basic Configuration Profile Platform

Note: You must download the Configuration Profile playbook as described in 7.2 and set it up as described in 7.3 before you complete this step.

In order to deploy the Basic Configuration Profile playbook, change the working directory to where you have cloned or unarchived the VMRA Ansible Playbook source code (as described in 2.5.2) and execute the command below:

```
$ ansible-playbook -i inventory.ini playbooks/vm.yml
```

7.5 Step 5 - Validate Basic Configuration Profile

Validate the setup of your Kubernetes cluster. Refer to the tasks in Post-Deployment Verification Guidelines and run the validation processes according to the hardware and software components that you have installed.

[1] Workloads and configurations. Results may vary.

8 VMRA Build-Your-Own Configuration Profile Setup

This section contains a step-by-step description of how to set up a VMRA Build-Your-Own Configuration Profile.

To use the Build-Your-Own Configuration Profile, perform the following steps:

1. Choose your hardware, set it up, and configure the BIOS. Refer to 6.2 for details.
2. Download and configure the Ansible playbook for your Configuration Profile. Refer to 8.2 for details.
3. Set up the optional Ansible parameters using the information in the Configuration Profile tables. Refer to 8.3 for details.
4. Deploy the platform. Refer to 8.4 for details.
5. Validate the setup of your Kubernetes cluster. Refer to 5 for details

8.1 Step 1 - Set Up Build-Your-Own Configuration Profile Hardware

Use the information in Set Up the VM Host – BIOS and HW Prerequisites to determine the hardware and BIOS configuration needed for this profile.

8.2 Step 2 - Download Build-Your-Own Configuration Profile Ansible Playbook

This section contains details for downloading the Build-Your-Own Configuration Profile Ansible playbook. It also provides an overview of the Ansible playbook and lists the software that is automatically installed when the playbook is deployed.

Download the Build-Your-Own Configuration Profile Ansible playbook using the steps described in 2.5.

8.2.1 Build-Your-Own Configuration Profile Ansible Playbook Overview

The Ansible playbook for the Build-Your-Own Configuration Profile allows you to provision a production-ready virtual infrastructure with or without a Kubernetes cluster. Every capability included in the Build-Your-Own Configuration Profile playbook can be disabled or enabled. Refer to Figure 4 and the group and host variables tables below.

Figure 4. Build-Your-Own Configuration Profile Ansible Playbook

8.3 Step 3 - Set Up Build-Your-Own Configuration Profile

Review the optional Ansible group and host variables in this section and select options that match your desired configuration.

1. Update the inventory.ini file with your environment details as described in 2.5.3.
2. Create host_vars/<hostname>.yml for the target VM host server and all VMs of type "work" as specified in 2.5.4.
3. Update group and host variables to match your desired configuration as specified in 2.5.4.

Variables are grouped into two main categories:

1. Group variables – apply to both control and worker nodes and have cluster-wide impact.
2. Host variables – their scope is limited to a single worker node or the VM host.

The tables below are a summary of group and host variables. For lists showing all configurable properties, see 6.3 and 6.4. All of the variables are important but pay special attention to variables in **bold** as they almost always need to be updated to match the target environment.

8.3.1 Build-Your-Own Configuration Profile Group Variables

Table 21. Build-Your-Own Configuration Profile – Group Variables

COMPONENT	VALUE	
Kubernetes	true	For the list of all configurable properties, see Section 6.3
nfd_enabled	true	
topology_manager_enabled	true	
sriov_network_operator_enabled	false	
sriov_net_dp_enabled	false	
example_net_attach_defs, sriov_net_dp	false	

8.3.2 Build-Your-Own Configuration Profile Host Variables

Table 22. Build-Your-Own Configuration Profile – Host Variables

COMPONENT	VALUE	
iommu_enabled	false	For the list of all configurable properties, see Section 6.4
sriov_cni_enabled	false	
isolcpus_enabled	false	
dns_disable_stub_listener	true	

8.4 Step 4 - Deploy Build-Your-Own Configuration Profile Platform

Note: You must download the Configuration Profile playbook as described in 8.2 and set it up as described in 8.3 before you complete this step.

In order to deploy the Build-Your-Own Configuration Profile playbook, change the working directory to where you have cloned or unarchived the VMRA Ansible Playbook source code (as described in 2.5.2) and execute the command below:

```
$ ansible-playbook -i inventory.ini playbooks/vm.yml
```

8.5 Step 5 - Validate Build-Your-Own Configuration Profile

Validate the setup of your Kubernetes cluster. Refer to the tasks in Post-Deployment Verification Guidelines and run the validation processes according to the hardware and software components that you have installed.

9 VMRA On-Premises Edge Configuration Profile Setup

This section contains a step-by-step description of how to set up a VMRA On-Premises Edge Configuration Profile.

To use the On-Premises Edge Configuration Profile, perform the following steps:

1. Choose your hardware, set it up, and configure the BIOS. Refer to 6.2 for details.
2. Download and configure the Ansible playbook for your Configuration Profile. Refer to 9.2 for details.
3. Set up the optional Ansible parameters using the information in the Configuration Profile tables. Refer to 9.3 for details.
4. Deploy the platform. Refer to 9.4 for details.
5. Validate the setup of your Kubernetes cluster. Refer to 5 for details.

9.1 Step 1 - Set Up On-Premises Edge Configuration Profile Hardware

Use the information in Set Up the VM Host – BIOS and HW Prerequisites to determine the hardware and BIOS configuration needed for this profile.

9.2 Step 2 - Download On-Premises Edge Configuration Profile Ansible Playbook

This section contains details for downloading the On-Premises Edge Configuration Profile Ansible playbook. It also provides an overview of the Ansible playbook and lists the software that is automatically installed when the playbook is deployed.

Download the On-Premises Edge Configuration Profile Ansible playbook using the steps described in 2.5.

9.2.1 On-Premises Edge Configuration Profile Ansible Playbook Overview

The Ansible playbook for the On-Premises Edge Configuration Profile allows you to provision a production-ready virtual infrastructure with or without a Kubernetes cluster. It also applies any additional requirements, such as host OS configuration or Network Adapter drivers and firmware updates. Every capability included in the On-Premises Edge Configuration Profile playbook can be disabled or enabled. To see which Ansible roles are included and executed by default, refer to Figure 5 and group and host variables tables below.

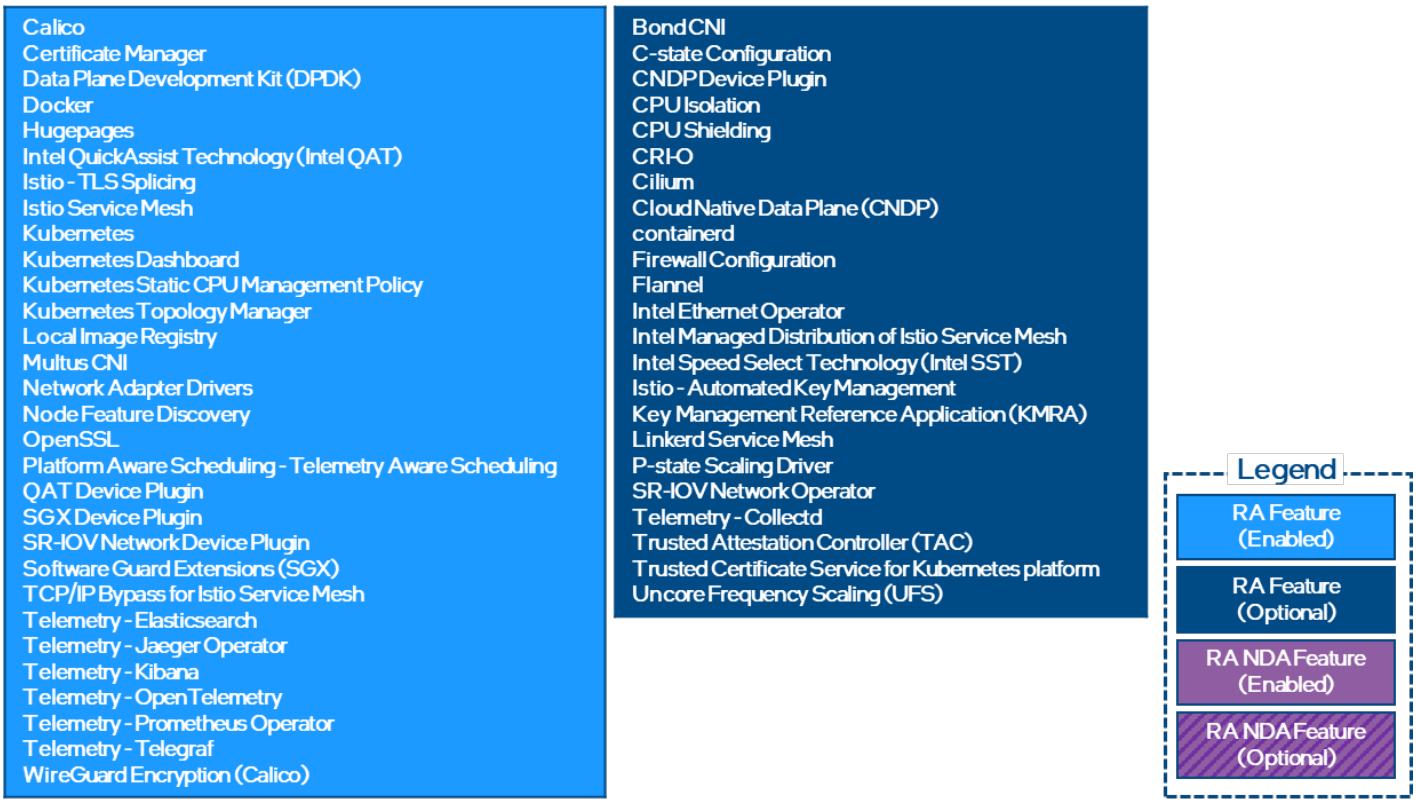

Figure 5. On-Premises Edge Configuration Profile Ansible Playbook

9.3 Step 3 - Set Up On-Premises Edge Configuration Profile

Review the optional Ansible group and host variables in this section and select options that match your desired configuration.

1. Update the inventory.ini file with your environment details as described in 2.5.3.
2. Create host_vars/<hostname>.yml for the target VM host server and all VMs of type "work" as specified in 2.5.4.
3. Update group and host variables to match your desired configuration as specified in 2.5.4.

Variables are grouped into two main categories:

1. Group variables – they apply to both control and worker nodes and have cluster-wide impact.
2. Host variables – their scope is limited to a single worker node or the VM host.

The tables below are a summary of group and host variables. For lists showing all configurable properties, see 6.3 and 6.4. All of the variables are important but pay special attention to variables in **bold** as they almost always need to be updated to match the target environment.

9.3.1 On-Premises Edge Configuration Profile Group Variables

Table 23. On-Premises Edge Configuration Profile – Group Variables

COMPONENT	VALUE	
Kubernetes	true	For the list of all configurable properties, see Section 6.3
nfd_enabled	true	
native_cpu_manager_enabled	true	
topology_manager_enabled	true	
sriov_network_operator_enabled	false	
sriov_net_dp_enabled	true	
sgx_dp_enabled	false	
gpu_dp_enabled	false	
qat_dp_enabled	true	
openssl_enabled	true	
kmra_enabled	false	
istio_enabled	true	
tas_enabled	true	
sst_pp_configuration_enabled	false	
example_net_attach_defs. userspace_ovs_dpdk	false	

9.3.2 On-Premises Edge Configuration Profile Host Variables

Table 24. On-Premises Edge Configuration Profile – Host Variables

COMPONENT	VALUE	
iommu_enabled	false	For the list of all configurable properties, see Section 6.4
sriov_cni_enabled	true	
bond_cni_enabled	false	
hugepages_enabled	true	
isolcpus_enabled	false	
dns_disable_stub_listener	true	
install_dpdk	true	
qat_devices	**[]**	

9.4 Step 4 - Deploy On-Premises Edge Configuration Profile Platform

Note: You must download the Configuration Profile playbook as described in 9.2 and configure it as described in 9.3 before you complete this step.

In order to deploy the On-Premises Edge Configuration Profile playbook, change the working directory to where you have cloned or unarchived the VMRA Ansible Playbook source code (as described in 2.5.2) and execute the command below:

```
ansible-playbook -i inventory.ini playbooks/vm.yml
```

9.5 Step 5 - Validate On-Premises Edge Configuration Profile

Validate the setup of your Kubernetes cluster. Refer to the tasks in Post-Deployment Verification Guidelines and run the validation processes according to the hardware and software components that you have installed.

10 VMRA Remote Central Office-Forwarding Configuration Profile Setup

This section contains a step-by-step description of how to set up a VMRA Remote Central Office-Forwarding Configuration Profile.

To use the Remote Central Office-Forwarding Configuration Profile, perform the following steps:

1. Choose your hardware, set it up, and configure the BIOS. Refer to 6.2 for details.
2. Download and configure the Ansible playbook for your Configuration Profile. Refer to 10.2 for details.
3. Set up the optional Ansible parameters using the information in the Configuration Profile tables. Refer to 10.3 for details.
4. Deploy the platform. Refer to 10.4 for details.
5. Validate the setup of your Kubernetes cluster. Refer to 5 for details

10.1 Step 1 - Set Up Remote Central Office-Forwarding Configuration Profile Hardware

Use the information in Set Up the VM Host – BIOS and HW Prerequisites to determine the hardware and BIOS configuration needed for this profile.

10.2 Step 2 - Download Remote Central Office-Forwarding Configuration Profile Ansible Playbook

This section contains details for downloading the Remote Central Office-Forwarding Configuration Profile Ansible playbook. It also provides an overview of the Ansible playbook and lists the software that is automatically installed when the playbook is deployed.

Download the Remote Central Office-Forwarding Configuration Profile Ansible playbook using the steps described in 2.5.

10.2.1 Remote Central Office-Forwarding Configuration Profile Ansible Playbook Overview

The Ansible playbook for the Remote Central Office-Forwarding Configuration Profile allows you to provision a production-ready virtual infrastructure with or without a Kubernetes cluster. It also applies any additional requirements, such as host OS configuration or network adapter drivers and firmware updates. Every capability included in the Remote Central Office-Forwarding Configuration Profile playbook can be disabled or enabled. To see which Ansible roles are included and executed by default, refer to Figure 6 and the group and host variables tables below.

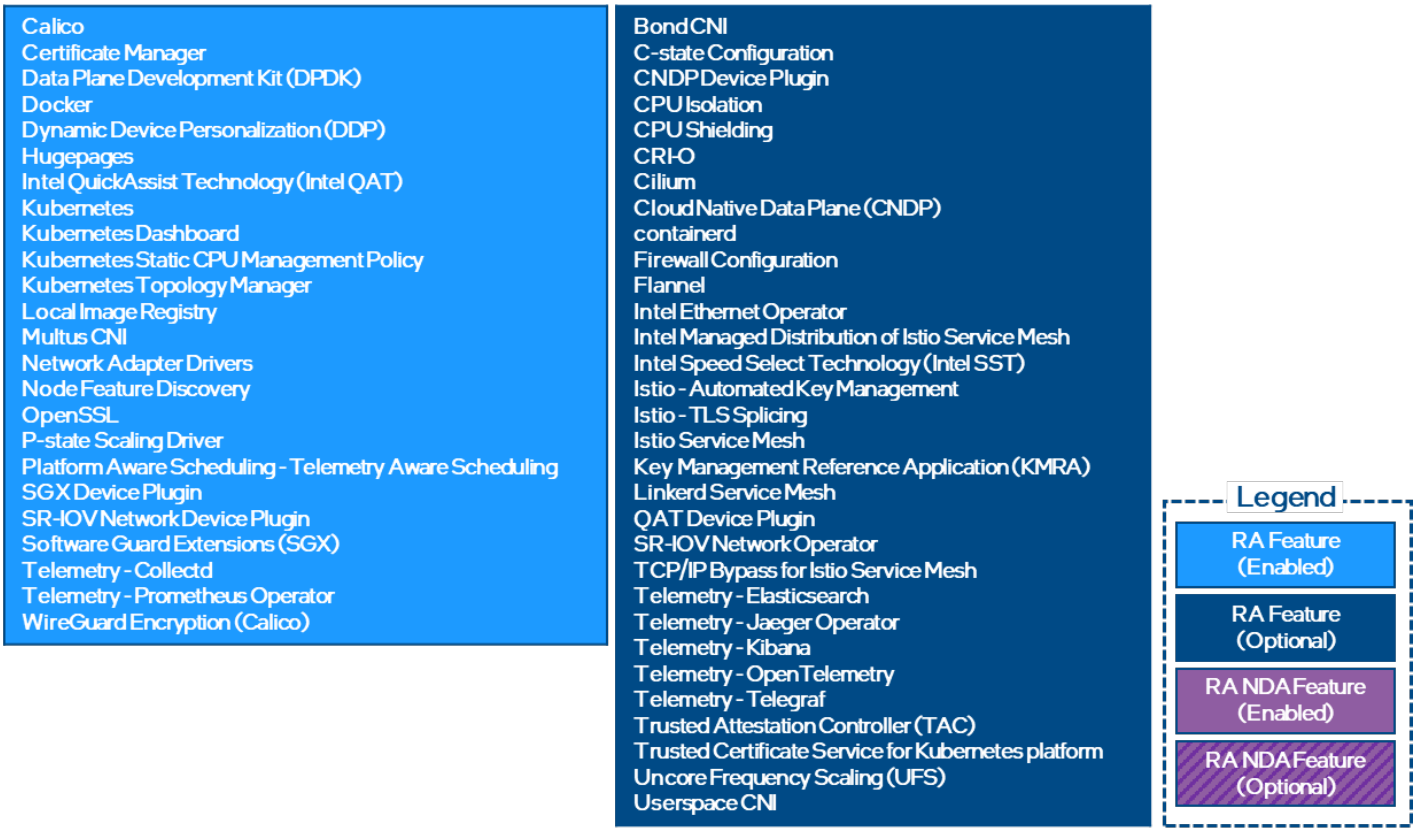

Figure 6. Remote Central Office-Forwarding Configuration Profile Ansible Playbook

10.3 Step 3 - Set Up Remote Central Office-Forwarding Configuration Profile

Review the optional Ansible group and host variables in this section and select options that match your desired configuration.

1. Update the inventory.ini file with your environment details as described in 2.5.3.
2. Create host_vars/<hostname>.yml for the target VM host server and all VMs of type "work" as specified in 2.5.4.
3. Update group and host variables to match your desired configuration as specified in 2.5.4.

Variables are grouped into two main categories:

1. Group variables – they apply to both control and worker nodes and have cluster-wide impact.
2. Host variables – their scope is limited to a single worker node or the VM host.

The tables below are a summary of group and host variables. For lists showing all configurable properties, see 6.3 and 6.4. All of the variables are important but pay special attention to variables in **bold** as they almost always need to be updated to match the target environment.

10.3.1 Remote Central Office-Forwarding Configuration Profile Group Variables

Table 25. Remote Central Office-Forwarding Configuration Profile – Group Variables

COMPONENT	VALUE	
Kubernetes	true	For the list of all configurable properties, see Section 6.3
nfd_enabled	true	
native_cpu_manager_enabled	true	
topology_manager_enabled	true	
sriov_network_operator_enabled	false	
sriov_net_dp_enabled	true	
sgx_dp_enabled	false	
gpu_dp_enabled	false	
qat_dp_enabled	false	
openssl_enabled	true	
kmra_enabled	false	
istio_enabled	true	
tas_enabled	true	
sst_cp_configuration_enabled	false	
example_net_attach_defs. userspace_ovs_dpdk	false	

10.3.2 Remote Central Office-Forwarding Configuration Profile Host Variables

Table 26. Remote Central Office-Forwarding Configuration Profile – Host Variables

COMPONENT	VALUE	
iommu_enabled	false	For the list of all configurable properties, see Section 6.4
sriov_cni_enabled	true	
bond_cni_enabled	false	
userspace_cni_enabled	false	
hugepages_enabled	true	
isolcpus_enabled	false	
dns_disable_stub_listener	true	
install_dpdk	true	
install_ddp_packages	false	
qat_devices	**[]**	

10.4 Step 4 - Deploy Remote Central Office-Forwarding Configuration Profile Platform

Note: You must download the Configuration Profile playbook as described in 10.2 and configure it as described in 10.3 before you complete this step.

In order to deploy the Remote Central Office-Forwarding Configuration Profile playbook, change the working directory to where you have cloned or unarchived the VMRA Ansible Playbook source code (as described in 2.5.2) and execute the command below:

```
ansible-playbook -i inventory.ini playbooks/vm.yml
```

10.5 Step 5 - Validate Remote-Central Office Forwarding Configuration Profile

Validate the setup of your Kubernetes cluster. Refer to the tasks in Post-Deployment Verification Guidelines and run the validation processes according to the hardware and software components that you have installed.

11 VMRA Regional Data Center Configuration Profile Setup

This section contains a step-by-step description of how to set up a VMRA Regional Data Center Configuration Profile.

To use the Regional Data Center Configuration Profile, perform the following steps:

1. Choose your hardware, set it up, and configure the BIOS. Refer to 6.2 for details.
2. Download and configure the Ansible playbook for your Configuration Profile. Refer to 11.2 for details.
3. Set up the optional Ansible parameters using the information in the Configuration Profile tables. Refer to 11.3 for details.
4. Deploy the platform. Refer to 11.4 for details.
5. Validate the setup of your Kubernetes cluster. Refer to 5 for details

11.1 Step 1 - Set Up Regional Data Center Configuration Profile Hardware

Use the information in Set Up the VM Host – BIOS and HW Prerequisites to determine the hardware and BIOS configuration needed for this profile.

11.2 Step 2 - Download Regional Data Center Configuration Profile Ansible Playbook

This section contains details for downloading the Regional Data Center Configuration Profile Ansible playbook. It also provides an overview of the Ansible playbook and lists the software that is automatically installed when the playbook is deployed.

Download the Regional Data Center Configuration Profile Ansible playbook using the steps described in 2.5.

11.2.1 Regional Data Center Configuration Profile Ansible Playbook Overview

The Ansible playbook for the Regional Data Center Configuration Profile allows you to provision a production-ready virtual infrastructure with or without a Kubernetes cluster. It also applies any additional requirements, such as host OS configuration or Network Adapter drivers and firmware updates. Every capability included in the Regional Data Center Configuration Profile playbook can be disabled or enabled. To see which Ansible roles are included and executed by default, refer to Figure 7 and the group and host variables tables below.

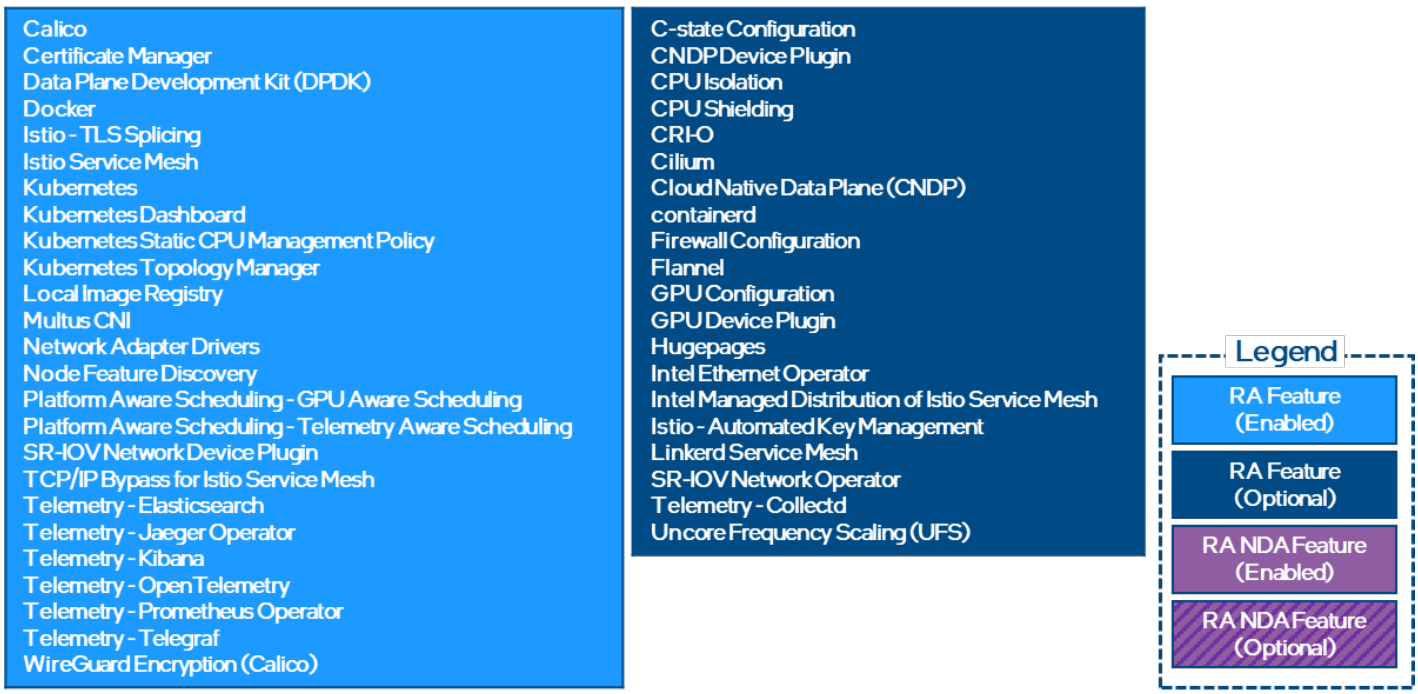

Figure 7. Regional Data Center Configuration Profile Ansible Playbook

11.3 Step 3 - Set Up Regional Data Center Configuration Profile

Review the optional Ansible group and host variables in this section and select options that match your desired configuration.

1. Update the inventory.ini file with your environment details as described in 2.5.3.
2. Create host_vars/<hostname>.yml for the target VM host server and all VMs of type "work" as specified in 2.5.4.
3. Update group and host variables to match your desired configuration as specified in 2.5.4.

Variables are grouped into two main categories:

1. Group variables – they apply to both control and worker nodes and have cluster-wide impact.
2. Host variables – their scope is limited to a single worker node or the VM host.

The tables below are a summary of group and host variables. For lists showing all configurable properties, see 6.3 and 6.4. All of the variables are important but pay special attention to variables in **bold** as they almost always need to be updated to match the target environment.

11.3.1 Regional Data Center Configuration Profile Group Variables

Table 27. Regional Data Center Configuration Profile – Group Variables

COMPONENT	VALUE	
Kubernetes	true	For the list of all configurable properties, see Section 6.3
nfd_enabled	true	
native_cpu_manager_enabled	true	
topology_manager_enabled	true	
sriov_network_operator_enabled	false	
sriov_net_dp_enabled	false	
gpu_dp_enabled	true	
Istio_enabled	true	
tas_enabled	true	
example_net_attach_defs. sriov_net_dp	false	

11.3.2 Regional Data Center Configuration Profile Host Variables

Table 28. Regional Data Center Configuration Profile – Host Variables

COMPONENT	VALUE	
iommu_enabled	false	For the list of all configurable properties, see Section 6.4
sriov_cni_enabled	false	
hugepages_enabled	false	
isolcpus_enabled	false	
dns_disable_stub_listener	true	
install_dpdk	false	

11.4 Step 4 - Deploy Regional Data Center Configuration Profile Platform

Note: You must download the Configuration Profile playbook as described in 11.2 and configure it as described in 11.3 before you complete this step.

In order to deploy the Regional Data Center Configuration Profile playbook, change the working directory to where you have cloned or unarchived the VMRA Ansible Playbook source code (as described in 2.5.2) and execute the command below:

```
ansible-playbook -i inventory.ini playbooks/vm.yml
```

11.5 Step 5 - Validate Regional Data Center Configuration Profile

Validate the setup of your Kubernetes cluster. Refer to the tasks in Post-Deployment Verification Guidelines and run the validation processes according to the hardware and software components that you have installed.

Part 3:
Release Notes

Appendix A VMRA Release Notes

This section lists the notable changes from the previous releases, including new features, bug fixes, and known issues.[2]

A.1 VMRA 23.02 Release Updates

New Components/Features:

- Non-root user deployment of VMRA
- Custom cluster naming

Updates/Changes:

- Versions upgraded for the vast majority of RA components (See User Guide for complete BOM and versions)

  Notable updates:
  - Kubernetes to v1.26.1
  - DPDK to v22.11.1
  - Service Mesh to v1.17.1
  - VPP to v2302
- Support of geo-specific mirrors for Kubespray (for example, in the People's Republic of China)

New Hardware (Platforms/CPUs/GPUs/Accelerators):

- N/A

Removed Support:

- full_nfv profile
- Ubuntu 20.04 as base operating system
- Rocky Linux 9.0 as base operating system

Known Limitations/Restrictions:

- VMRA cluster expansion with additional VM nodes might fail
- Trusted Certificate Attestation (TCA) is not fully functional in VMRA

A.2 RA 22.11.1 Release Notes

New Components/Features:

- N/A (same as RA 22.11)

Updates/Changes:

- Intel® QAT 2.0 drivers for 4th Gen Intel® Xeon® Scalable processors (formerly code named Sapphire Rapids [SPR]) are sourced from public repo. No longer under NDA. Ignore Guide requirement to provide the *QAT20.L.0.9.9-00019.tar.gz* driver package file.
- Resolved issue regarding downloading CPUID for Rocky Linux 8.5 and RHEL 9.

New Hardware (Platforms/CPUs/GPUs/Accelerators):

- N/A (same as RA 22.11)

Removed Support:

- N/A (same as RA 22.11)

Known Limitations/Restrictions:

- N/A (same as RA22.11)

A.3 VMRA 22.11 Release Updates

- VMRA now supports telemetry options such as Jaeger/OpenTelemetry
- Support for Cilium as a Container Network Interface (CNI)
- This release is now based on Kubespray 2.20.0, which enables Kubernetes 1.25.x
- Several components have been updated to improve functionality and security. See the software component table in this document for version information.

[2] Workloads and configurations. Results may vary.

A.4 VMRA 22.08 Release Updates

- In this release, automatic CPU pinning from version 22.05 is enhanced to reserve CPUs for both the host OS and vCPUs for the guest OS in addition to assigning vCPUs to cores within the same NUMA zone.
- VMRA also supports extending an existing cluster.
- This release also provides the user an option to specify a .py,.sh or .yaml file to run on either the Ansible Host and/or the Kubernetes control plane.
- VMRA now supports Linkerd for service mesh implementation.

A.5 Known Issues

Issue: VFs specified in host_vars "dataplane_interfaces are not bound to the expected VF driver

Detail: Due to VMs not having access to physical functions (PFs) on Ethernet adapters attached to the VM Host, the configuration of VFs is skipped inside VMs. As a result, VFs will be bound to the Linux "iavf" driver and show up as "netdevice" devices through the SR-IOV Network Device Plugin.

Workaround: Follow the steps listed in Table 15 (Check SR-IOV device plugin) to rebind VFs to the correct driver.

Issue: VMRA 22.05 introduced AF_XDP and CNDP support for SR-IOV Virtual Functions (VFs) using the Linux kernel iavf driver. The iavf driver does not currently support XDP or AF_XDP zero-copy, so the kernel's generic eXpress Data Path (XDP) is used. This results in extra per-packet overhead due to allocation of SKBs and requires a copy to get the packet data from the kernel to the AF_XDP socket in user space.

Detail: Applications using AF_XDP sockets on devices that do not support XDP or AF_XDP zero-copy will generally result in lower performance than applications using AF_XDP sockets on devices that support XDP and AF_XDP zero-copy.

Workaround: AF_XDP in XDP_SKB mode is used for devices that do not support AF_XDP zero-copy.

Issue: GPU Aware Scheduling (GAS) is enabled for the Regional Data Center profile, even though it is not supported or tested for VMRA.

Detail: As part of Platform Aware Scheduling (PAS), the GPU Aware Scheduling (GAS) extender is enabled for the Regional Data Center profile. GPU virtualization is not currently supported in VMRA, which might cause unexpected behavior of the extender.

Workaround: If configuring the Regional Data Center profile (regional_dc), manually update "gas_enabled" in group_vars/all.yml to "false"

Issue: QAT Devices are not providing additional performance for 3rd Gen Intel® Xeon® Scalable processors through offloading.

Detail: While QAT Devices can be configured and will show up in the Kubernetes cluster as an allocatable resource, they do not provide the expected performance increase. When testing with the OpenSSL Engine, the performance is similar regardless of QAT offloading, which indicates that OpenSSL will default to software as a fallback solution.

Workaround: There is currently no workaround available. OpenSSL will still work, but without the performance increase from HW offloading.

Issue: Pods requesting additional networks using SR-IOV CNI will fail to start

Detail: The SR-IOV CNI needs access to the physical functions (PFs) on Ethernet adapters attached to the VM Host. As these are not available in the VMs, SR-IOV CNI will fail to create pod interfaces and the pod will not be started.

Workaround: There is currently no workaround available. VFs that are listed as allocatable resources can still be requested and added to pods, but the additional functionality of SR-IOV CNI such as IPAM and making the interface available in the pod will not work.

Issue: Occasionally the sriov-network-device-plugin does not detect new or updated VF resources.

Detail: There is a known issue with sriov-network-device-plugin where the service fails to detect new or updated VF resources if not available when the service creates its ConfigMap and loads the daemonset. See https://github.com/k8snetworkplumbingwg/sriov-network-device-plugin/issues/276.

Workaround:

Delete the sriov-device-plugin-pod and resources will be present when pod is automatically restarted.

Issue: Collectd plugin fails to start.

Detail: On some platforms, the collectd pod fails to start due to various plugin incompatibilities.

Workaround: Disable problematic collectd plugins by adding to the exclude_collectd_plugins list in the Ansible host_vars configuration file.

Appendix B Abbreviations

The following abbreviations are used in this document.

ABBREVIATION	DESCRIPTION
5GC	5G Core
AGF	Access Gateway Function
AIA	Accelerator Interfacing Architecture
AMX	Advance Matrix Multiply
BIOS	Basic Input/Output System
BMRA	Bare Metal Reference Architecture
BOM	Bill of Material
CA	Certificate Authority
CDN	Content Delivery Network
CLOS	Class of Service
CMTS	Cable Modem Termination System
CNF	Cloud Native Network Function
CNI	Container Network Interface
CO	Central Office
CRI	Container Runtime Interface
CSP	Cloud Service Provider
CXL	Compute Express Link
DDP	Dynamic Device Personalization
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Service
DPDK	Data Plane Development Kit
DRAM	Dynamic Random Access Memory
DSA	Intel® Data Streaming Accelerator (Intel® DSA)
FP	Floating Point
FPGA	Field-Programmable Gate Array
FW	Firmware
GPU	Graphics Processor Unit
HA	High Availability
HCC	High Core Count
HSM	Hardware Security Model
HT	Hyper Threading
IAX	In-Memory Analytics
IMC	Integrated Memory Controller
Intel® AVX	Intel® Advanced Vector Extensions (Intel® AVX)
Intel® AVX-512	Intel® Advanced Vector Extension 512 (Intel® AVX-512)
Intel® DLB	Intel® Dynamic Load Balancer (Intel® DLB)
Intel® DSA	Intel® Data Streaming Accelerator (Intel® DSA)
Intel® HT Technology	Intel® Hyper-Threading Technology (Intel® HT Technology)
Intel® QAT	Intel® QuickAssist Technology (Intel® QAT)
Intel® RDT	Intel® Resource Director Technology (Intel® RDT)
Intel® SecL – DC	Intel® Security Libraries for Data Center (Intel® SecL – DC)
Intel® SGX	Intel® Software Guard Extensions (Intel® SGX)
Intel® Scalable IOV	Intel® Scalable I/O Virtualization

ABBREVIATION	DESCRIPTION
Intel® SST-BF	Intel® Speed Select Technology – Base Frequency (Intel® SST-BF)
Intel® SST-CP	Intel® Speed Select Technology – Core Power (Intel® SST-CP)
Intel® SST-PP	Intel® Speed Select Technology – Performance Profile (Intel® SST-PP)
Intel® SST-TF	Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF)
Intel® VT-d	Intel® Virtualization Technology (Intel® VT) for Directed I/O (Intel® VT-d)
Intel® VT-x	Intel® Virtualization Technology (Intel® VT) for IA-32, Intel® 64 and Intel® Architecture (Intel® VT-x)
IOMMU	Input/Output Memory Management Unit
ISA	Instruction Set Architecture
I/O	Input/Output
K8s	Kubernetes
KMS	Key Management Service (KMS)
LCC	Low Core Count
LLC	Last Level Cache
LOM	LAN on Motherboard
NFD	Node Feature Discovery
NFV	Network Function Virtualization
NIC	Network Interface Card
NTP	Network Time Protocol
NVM	Non-Volatile Memory
NVMe	Non-Volatile Memory
OAM	Operation, Administration, and Management
OCI	Open Container Initiative
OS	Operating System
OVS	Open vSwitch
OVS DPDK	Open vSwitch with DPDK
PBF	Priority Based Frequency
PCCS	Provisioning Certification Caching Service
PCI	Physical Network Interface
PCIe	Peripheral Component Interconnect express
PMD	Poll Mode Driver
PXE	Preboot Execution Environment
QAT	Intel® QuickAssist Technology
QoS	Quality of Service
RA	Reference Architecture
RAS	Reliability, Availability, and Serviceability
RDT	Intel® Resource Director Technology
S-IOV	Intel® Scalable I/O Virtualization (Intel® Scalable IOV)
SA	Service Assurance
SGX	Intel® Software Guard Extensions (Intel® SGX)
SR-IOV	Single Root Input/Output Virtualization
SSD	Solid State Drive
SSH	Secure Shell Protocol
SVM	Shared Virtual Memory
TAS	Telemetry Aware Scheduling
TDP	Thermal Design Power

ABBREVIATION	DESCRIPTION
TLS	Transport Layer Security
TME	Total Memory Encryption
TMUL	Tile Multiply
UEFI	Unified Extensible Firmware Interface
UPF	User Plane Function
vBNG	Virtual Broadband Network Gateway
vCMTS	Virtual Cable Modem Termination System
VF	Virtual Function
VMRA	Virtual Machine Reference Architecture
VNF	Virtual Network Function
VPP	Vector Packet Processing

0323/DN/WIT/PDF 721803-008US