User Guide

Network and Edge Virtual Machine Reference System Architecture Release v23.07

Authors

Aparna Balachandran

Octavia Carotti

Calin Gherghe

Dana Nehama

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 (Reference System[1]) 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] In this document, "Reference System" refers to the Network and Edge Reference System Architecture.

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 v23.07 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 Systems.

1.3 Version 23.07 Release Information

VMRA 23.07 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 v23.07 builds upon prior releases. The following are the key release updates:

- Software Updates (details in Reference System Architecture Software Components)
- Support for Intel® Software Guard Extensions (Intel® SGX) by upgrading QEMU and libvert
- Support for Key Management Reference Application (KMRA) as Intel® SGX is available
- Enabled Intel® SGX signer for Istio* Service Mesh

For additional details, refer to the VMRA Release Notes.

Experience Kits, the collaterals that explain in detail the technologies enabled in VMRA release 23.07, are available at Network & Edge Platform Experience Kits.

Table of Contents

1 Introduction 1
1.1 Purpose and Scope 1
1.2 User Guide Information 2
1.3 Version 23.07 Release Information 2
1.4 Key Terms 6
1.5 Intel Investments of Capabilities 7
1.6 Reference Documentation 8

2 Reference System Architecture Deployment 10
2.1 VMRA Architecture 10
2.2 Configuration Profiles 11
2.3 Reference System Architecture Installation Prerequisites 11
2.3.1 Hardware BOM Selection and Setup for VM Host 11
2.3.2 BIOS Selection for VM Host 12
2.3.3 Operating System Selection for VM Host and VMs 12
2.3.4 Network Interface Requirements for VM Host 12
2.4 Ansible Playbook 12
2.4.1 Ansible Playbooks Building Blocks 12
2.4.2 Ansible Playbook Phases 13
2.5 Deployment Using Ansible Playbook 14
2.5.1 Prepare VM Host Server 14
2.5.2 Prepare Ansible Host and Configuration Templates 14
2.5.3 Update Ansible Inventory File 16
2.5.4 Update Ansible Host and Group Variables 16
2.5.5 Run Ansible Cluster Deployment Playbook 17
2.5.6 Expand Existing VMRA Cluster 17

3 Reference System Architecture Hardware Components and BIOS 18
3.1 Hardware Component List for 3rd Gen Intel Xeon Scalable Processor VM Host 18
3.2 Hardware Component List for 4th Gen Intel Xeon Scalable Processor VM Host 18
3.3 Platform BIOS Profiles Settings 19
3.3.1 3rd Gen Intel® Xeon® Scalable Processor Platform BIOS 19
3.3.2 4th Gen Intel® Xeon® Scalable Processor Platform BIOS 20

4 Reference System Architecture Software Components 21
4.1 Software Components Supported 21

5 Post-Deployment Verification Guidelines 24
5.1 Check Grafana Telemetry Visualization 24

6 VMRA Setup – Applicable for All Configuration Profiles 26
6.1 Set Up an Ansible Host 26
6.2 Set up the VM Host – BIOS Prerequisites 26
6.3 Configuration Dictionary - Group Variables 26
6.4 Configuration Dictionary - Host Variables 29

7 VMRA Basic Configuration Profile Setup 32
7.1 Supported Hardware 32
7.2 Recommended BIOS 32
7.3 Recommended OS 32
7.4 Playbook Overview 32
7.4.1 Basic Configuration Profile Group Variables 32
7.4.2 Basic Configuration Profile Host Variables 34
7.5 Deploy and Validate Basic Configuration Profile Platform 34

8 VMRA Build-Your-Own Configuration Profile Setup 35
8.1 Supported Hardware 35
8.2 Recommended BIOS 35
8.3 Recommended OS 35
8.4 Playbook Overview 35
8.4.1 Build-Your-Own Configuration Profile Group Variables 35

8.4.2 Build-Your-Own Configuration Profile Host Variables 37
8.5 Deploy and Validate Build-Your-Own Configuration Profile Platform 38

9 VMRA On-Premises Edge Configuration Profile Setup 39
9.1 Supported Hardware 39
9.2 Recommended BIOS 39
9.3 Recommended OS 39
9.4 Playbook Overview 39
9.4.1 On-Premises Edge Configuration Profile Group Variables 39
9.4.2 On-Premises Edge Configuration Profile Host Variables 41
9.5 Deploy and Validate On-Premises Edge Configuration Profile Platform 41

10 VMRA Remote Central Office-Forwarding Configuration Profile Setup 42
10.1 Supported Hardware 42
10.2 Recommended BIOS 42
10.3 Recommended OS 42
10.4 Playbook Overview 42
10.4.1 Remote Central Office-Forwarding Configuration Profile Group Variables 42
10.4.2 Remote Central Office-Forwarding Configuration Profile Host Variables 44
10.5 Deploy and Validate Remote Central Office-Forwarding Configuration Profile Platform 45

11 VMRA Regional Data Center Configuration Profile Setup 46
11.1 Supported Hardware 46
11.2 Recommended BIOS 46
11.3 Recommended OS 46
11.4 Playbook Overview 46
11.4.1 Regional Data Center Configuration Profile Group Variables 46
11.4.2 Regional Data Center Configuration Profile Host Variables 48
11.5 Deploy and Validate Regional Data Center Configuration Profile 48

Appendix A VMRA Release Notes 50
A.1 VMRA 23.07 Release Updates 50
A.2 VMRA 23.02 Release Updates 50
A.3 VMRA 22.11.1 Release Notes 51
A.4 VMRA 22.11 Release Updates 51
A.5 VMRA 22.08 Release Updates 51
A.6 Known Issues 51

Appendix B Abbreviations 53

Figures

Figure 1 Virtual Machine Reference System Architecture Illustration with Kubernetes Cluster 10
Figure 2. VMRA Multiple-Node Deployment 10

Tables

Table 1. Terms Used 6
Table 2. Hardware and Software Configuration Taxonomy 7
Table 3. Intel Capabilities Investments and Benefits 7
Table 4. Hardware Components for Host Base – 3rd Gen Intel Xeon Scalable Processor 18
Table 5. Hardware Components for Host Base – 4th Gen Intel Xeon Scalable Processor 18
Table 6. Additional BIOS options for VMRA 19
Table 7. Platform BIOS Profile settings for 3rd Gen Intel® Xeon® Scalable Processor 19
Table 8. Platform BIOS Profile settings for 4th Gen Intel® Xeon® Scalable Processor 20
Table 9. Software Components 22
Table 10. Links to Verification Guidelines on GitHub 24
Table 11. Configuration Dictionary – Group Variables for VMRA 26
Table 12. Configuration Dictionary - Host Variables for VMRA 30
Table 13. Basic Configuration Profile – Group Variables 33
Table 14. Basic Configuration Profile – Host Variables 34

Table 15. Build-Your-Own Configuration Profile – Group Variables 35
Table 16. Build-Your-Own Configuration Profile – Host Variables 37
Table 17. On-Premises Edge Configuration Profile – Group Variables 39
Table 18. On-Premises Edge Configuration Profile – Host Variables 41
Table 19. Remote Central Office-Forwarding Configuration Profile – Group Variables 42
Table 20. Remote Central Office-Forwarding Configuration Profile – Host Variables 44
Table 21. Regional Data Center Configuration Profile – Group Variables 46
Table 22. Regional Data Center Configuration Profile – Host Variables 48

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 VMRA release 22.08.
007	December 2022	Updated for VMRA release 22.11.
008	March 2023	Updated for VMRA Release 23.02, with changes to deployments.
009	July 2023	Updated for VMRA Release 23.07

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 business and 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 System Architecture Flavor	An instance of Reference System generated by implementing a Configuration Profile specification.
Ansible Playbook	A set of validated scripts that prepare, configure, and deploy a Reference System 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*. Access the *Network and Edge Reference System Architectures - Single Server Quick Start Guide* for step-by-step instructions to build VMRA on a single server setup.

Other collaterals, including technical guides and solution briefs that explain in detail the technologies enabled in VMRA release v23.07, are available in the following location: Network & Edge Platform Experience Kits.

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

2 Reference System 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.

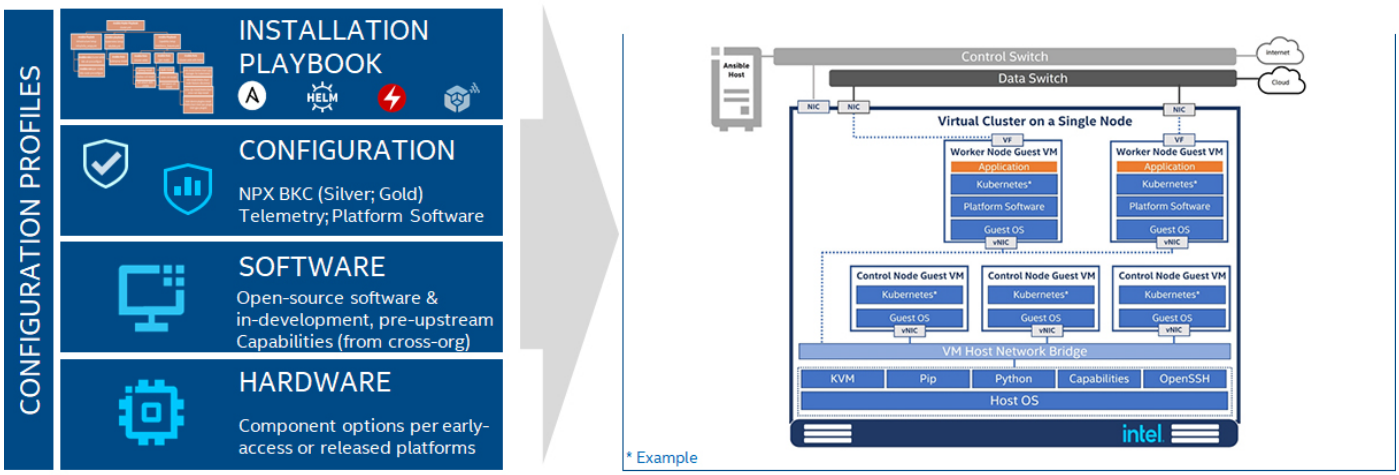

Figure 1. Virtual Machine Reference System Architecture Illustration with Kubernetes Cluster

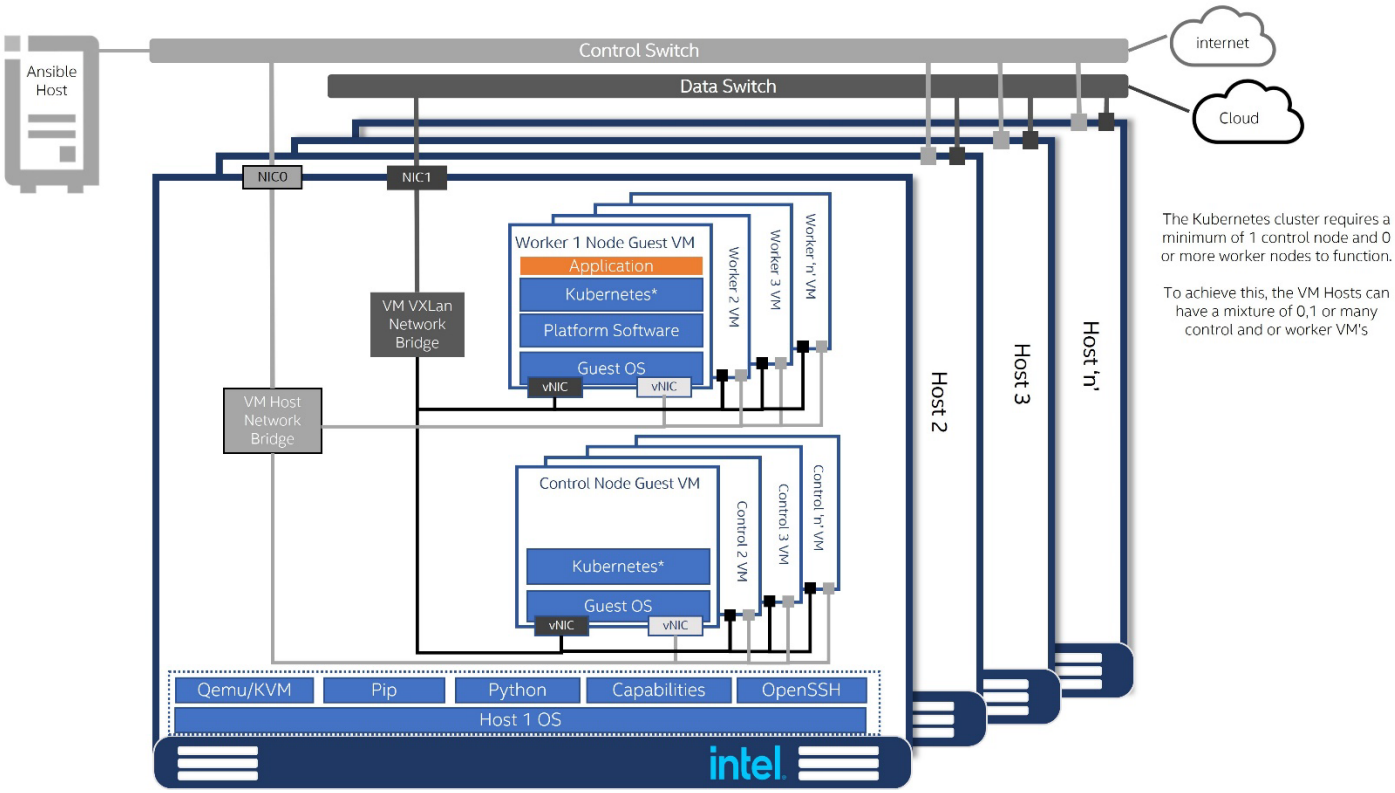

Figure 2. VMRA Multiple-Node Deployment

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 System Architecture Flavor per a Configuration Profile specification.

The following Reference System 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 System 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 System 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 VM Host

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

Hardware Component List for 4th Gen Intel Xeon Scalable Processor VM Host

Hardware Component List for 3rd Gen Intel Xeon Scalable Processor VM Host

Configuration Profile BOM: See Sections 7 through 11 for details about hardware BOM selection and setup for your chosen Configuration Profile.

2.3.2 BIOS Selection for VM Host

Enter the UEFI or BIOS menu and update the configuration as listed in Section 6 and in the tables in Section 3.3, which describe the BIOS selection in detail.

2.3.3 Operating System Selection for VM Host and VMs

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

- RHEL 9.2
- Rocky Linux 9.1
- Ubuntu 22.04 (22.04.2)

The VMs support the following Linux operating systems:

- Ubuntu 22.04 (22.04.2)
- Rocky Linux 8.5
- Rocky Linux 9.1

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 requirements must be met:

- The VM host must have network connectivity to the Ansible host.
- All systems must have public internet connectivity.

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

2.4 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

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/remote_fp.yml
- playbooks/regional_dc.yml
- playbooks/on_prem.yml
- playbooks/basic.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 following example assumes that the host is running RHEL. Other operating systems may have slightly different installation steps:
  ```
  yum install python3
  ```
- 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
  Section 11, VMRA Regional Data Center Configuration Profile Setup.

2.5.2 Prepare Ansible Host and Configuration Templates

Perform the following steps:

1. Log in to your Ansible host (the one that you will run these Ansible playbooks from).
2. (optional) Configure proxies if necessary:
   a. Add proxy configuration to /etc/environment (values included below are for example purposes):
      ```
      http_proxy=http://proxy.example.com:1080
      https_proxy=http://proxy.example.com:1080
      ```
   b. Update current environment to include proxy configuration from previous step:
      ```
      source /etc/environment
      ```

3. Install packages on Ansible host. The following example assumes that the host is running RHEL. Other operating systems may have slightly different installation steps and some packages may already be present:

```
yum install python3 python3-pip libselinux-python3 openssh-server git
pip3 install --upgrade pip
```

4. Enable passwordless login between all nodes in the cluster.

   Create authentication SSH-Keygen keys on Ansible host:

```
ssh-keygen
```

5. SSH is used by the Ansible host to communicate with each target node. Configure the same SSH keys on each machine. Copy your generated public keys to all the nodes from the Ansible host:

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

6. Clone the source code and change work directory.

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

7. Check out the latest version of the playbooks – using the tag from Table 9, for example:

```
git checkout v23.07
```

Note: Alternatively go to Container Experience Kits Releases,, download the latest release tarball, and unarchive it:

```
wget https://github.com/intel/container-experience-kits/archive/v23.07.tar.gz
tar xf v23.07.tar.gz
cd container-experience-kits-23.07
```

8. Decide which Configuration Profile you want to deploy and export the environmental variable.

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

```
export PROFILE=on_prem
```

   For Kubernetes **Basic** Configuration Profile deployment:

```
export PROFILE=basic
```

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

```
export PROFILE=build_your_own
```

9. Install Python dependencies using one of the following methods:

   a. (non-invasive) Virtual environment using pipenv:

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

   b. (non-invasive) Virtual environment using venv:

```
python3 -m venv venv
source venv/bin/activate
pip3 install -r requirements.txt
```

   c. (not recommended) System environment:

```
pip3 install -r requirements.txtInstall Ansible collection dependencies:
```

10. Install Ansible collection dependencies:

```
ansible-galaxy install -r collections/requirements.yml
```

11. Generate example profiles.

```
make vm-profile ARCH=<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

   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.

- (Required) Apply required patches for Kubespray:

```
ansible-playbook -i inventory.ini playbooks/k8s/patch_kubespray.yml
```

- (Optional, recommended) Verify that Ansible can connect to the target servers, by running the below command and checking the output generated in the **all_system_facts.txt** file:

```
ansible -i inventory.ini -m setup all > all_system_facts.txt
```

- (Optional, recommended) Check dependencies of components enabled in group_vars and host_vars with the packaged dependency checker. This step is also run by default as part of the main playbook:

```
# 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.

- Run the main 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.

Pay attention to logs and messages displayed on the screen. Depending on the selected Configuration Profile, network bandwidth, storage speed, and other similar factors, the execution will likely take between 30-90 minutes.

After the playbook finishes without any "Failed" tasks, you can proceed with the deployment validation described in Section 5

Note: Additional information can be found in the Ansible project root directory 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 System Architecture Hardware Components and BIOS

The VMRA supports a range of hardware that enables the different Configuration Profiles and deployment models.

The following tables list the base hardware options for the host, as well as the BIOS components available.

If your configuration needs improved processing, you may choose to upgrade the CPU, network adapter, and memory.

3.1 Hardware Component List for 3rd Gen Intel Xeon Scalable Processor VM Host

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	Recommended
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	

Some configuration profiles may need the host to have a CPU upgrade and increase in memory to 512 GB.

- CPU upgrade: Intel® Xeon® Gold 6338N CPU @ 2.2 GHz 32 C/64 T, 185W

3.2 Hardware Component List for 4th Gen Intel Xeon Scalable Processor VM Host

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	

Some configuration profiles may need the host to have a CPU upgrade and increase in memory to 512 GB.

- CPU upgrade: Intel® Xeon® Gold 6438N processor at 1.8GHz, 32 C/64 T, 205 W

3.3 Platform BIOS Profiles Settings

This section provides BIOS configuration profiles for each of the VMRA Configuration Profiles.

In addition to the BIOS settings listed in Section 3.3.1 and Section 3.3.2, VMRA requires the following settings regardless of configuration profile and hardware choice:

Table 6. Additional BIOS options for VMRA

OPTION	ENABLED
Intel® HT Technology Enabled	Yes
Intel® VT-x Enabled	Yes
Intel® VT-d Enabled	Yes
Virtualization Enabled	Yes

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.

3.3.1 3rd Gen Intel® Xeon® Scalable Processor Platform BIOS

Table 7. 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

3.3.2 4th Gen Intel® Xeon® Scalable Processor Platform BIOS

Table 8. Platform BIOS Profile settings for 4th Gen Intel® Xeon® Scalable Processor

MENU (ADVANCED)	PATH TO BIOS SETTING	BIOS SETTING	LOW LATENCY	MAX PERFORMANCE WITH TURBO	ENERGY BALANCE TURBO
Socket Configuration	Processor Configuration	Hyper-Threading	Enable	Enable	Enable
		X2APIC	Enable	Enable	Enable
		VMX	Enable	Enable	Enable
		Homeless Prefetch	Enable	Disable (default)	Disable (default)
		LLC Prefetch	Disable	Enable	Enable
		SNC	Disable	Disable	Disable
		Uncore RAPL	Disable	Disable	Enable
		Uncore frequency scaling	Disable	Disable	Enable
		Uncore frequency	1.8GHz (hex 0x12)	1.6MHz (hex 0x10)	800MHz to 2.5GHz
Power Configuration	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	Enable	Disable	Disable
		AVX ICCP Pre Grant Level	Level 5	NA	NA
		AVX P1 (ConfigTDP)	Level 2	Nominal (default)	Nominal
		Energy Efficient Turbo	Disable	Disable	Enable
		GPSS timer	0us	0us	0us
		Turbo	Enable	Enable	Enable
		Intel® SpeedStep® Technology	Enable	Enable	Enable
	Frequency Prioritization	RAPL Prioritization	Disable	Disable	Disable
	Common Ref Code	UMA-Based Clustering	Quadrant	Quadrant	Quadrant
	Hardware PM State Control	Hardware P-states	Native with no Legacy Support	Native with no Legacy Support	Native with no Legacy Support
		EPP enable	Disable	Disable	Disable
	CPU C-state Control	Enable Monitor Mwait	Enable	Enable	Enable
		CPU C1 Auto Demotion	Disable	Disable	Disable
		CPU C1 Auto unDemotion	Disable	Disable	Disable
		Processor C6 or CPU C6 Report	Enable	Enable	Enable
		Enhanced Halt State (C1E)	Enable (per Core Level)	Enable	Enable
		OS ACPI Cx	ACPI C2	ACPI C2	ACPI C2
	Energy Performance Bias	Power Performance Tuning	OS Control EPB	OS Controls EPB	OS Controls EPB
		Workload Configuration	I/O Sensitive	I/O Sensitive	Balanced
	Package C-state Control	Package C-state	C6 Retention	C0/C1 State	C0/C1 State
		Dynamic L1	Enable	Disable	Disable
Memory Configuration		Memory Configuration	8-way interleave	8-way interleave	8-way interleave
		Enforce POR / Memory Patrol Scrub	Enable/Disable	Enable/Enable	Enable/Enable
		Memory DIMM Refresh Rate	1x	1x	2x
Platform Configuration	Miscellaneous Configuration	Serial Debug Message Level	Minimum	Minimum	Minimum
	PCI Express* Configuration	PCIe* ASPM	Disable	Enable	Enable
		ECRC generation and checking	Disable	Enable	Enable
Server Management		Resume on AC Power Loss	Power On	Power On	Power On
System Acoustic and Performance Configuration		Set Fan Profile	Performance	Acoustic	Acoustic

4 Reference System Architecture Software Components

4.1 Software Components Supported

Table 9 lists all the software components automatically deployed per Configuration Profile in a VMRA and their sources.

Table 9. Software Components

SOFTWARE FUNCTION	SOFTWARE COMPONENT	LOCATION
OS	Ubuntu 22.04.02	https://www.ubuntu.com
OS	RHEL 9.2	https://developers.redhat.com/products/rhel/download
OS	Rocky 9.1	https://rockylinux.org/download
Data Plane Development Kit (DPDK)	23.03	https://core.dpdk.org/download/
Open vSwitch with DPDK	3.1.1	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	0.27	https://github.com/open-telemetry/opentelemetry-operator
Jaeger	1.44.0	https://github.com/jaegertracing/jaeger-operator
Grafana	9.4.7	https://www.grafana.com/
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	v23.07	https://github.com/intel/container-experience-kits
Python	Python 3.10.4 for Ubuntu 22.04	https://www.python.org/
Kubespray	f9f5143c93f583541ccb6650eb008f7ef3d1bc3c	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	1.7.0	https://github.com/containerd/containerd/tags
CRI-O	1.26.3	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)	0.5	https://github.com/intel/platform-aware-scheduling
Platform Aware Scheduling (GAS)	0.5.2	https://github.com/intel/platform-aware-scheduling
Prometheus	2.43.0	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	0.64.1	https://quay.io/repository/prometheus-operator/prometheus-operator?tab=tags
Kubernetes RBAC Proxy	0.14.1	https://github.com/brancz/kube-rbac-proxy/tags
Container Registry	Registry: 2.8.1	https://github.com/distribution/distribution/tags
	Nginx: 1.23.4-alpine	https://github.com/docker-library/docs/tree/master/nginx
Node Feature Discovery	0.13.1-minimal	https://github.com/kubernetes-sigs/node-feature-discovery
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

SOFTWARE FUNCTION	SOFTWARE COMPONENT	LOCATION
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 v2.22.18	https://sourceforge.net/projects/e1000/files/i40e%20stable
	ice v1.11.14	https://sourceforge.net/projects/e1000/files/ice%20stable/
	iavf v4.8.2	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	1.3.40.0	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.1.0.40-00004	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.4.22.0-00001	https://www.intel.com/content/www/us/en/download/19734/intel-quickassist-technology-driver-for-linux-hw-version-1-7.html
OpenSSL	openssl-3.1.0	https://github.com/openssl/openssl/tags
OpenSSL QAT Engine	1.0.0	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	2.19.100.3	https://download.01.org/intel-sgx/sgx-dcap/1.15/linux/distro/
Intel® SGX packages	2.19.100.3	https://download.01.org/intel-sgx/sgx_repo/ubuntu/dists/jammy/main/binary-amd64/Packages
Intel® SGX DCAP packages	1.16.100.2	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	1.18.1	https://github.com/istio/istio/tags
Intel Managed Distribution of Istio Service Mesh	1.18.0-intel.0	https://github.com/intel/istio/tags

SOFTWARE FUNCTION	SOFTWARE COMPONENT	LOCATION
Trusted Attestation Controller (TAC)	0.4.0	https://github.com/intel/trusted-attestation-controller/tags
Trusted Certificate Service for Kubernetes platform	0.5.0	https://github.com/intel/trusted-certificate-issuer/tags
Go Programming Language	1.20.4	https://go.dev/dl/
libvirt	9.3.0	https://github.com/libvirt/libvirt/tags
Linkerd	2.13.3	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 10.

Table 10. 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

The Ansible host is used for configuring and deploying VMRA.

Information about supported operating systems and requirements can be found in Section 2.3.3.

Steps for installing required packages and preparing the Ansible host for deployment can be found in Section 2.5.2.

6.2 Set up the VM Host – BIOS Prerequisites

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

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

There are four different BIOS profiles listed in Section 3.1 and Section 3.2:

- Energy Balance
- Max Performance
- Deterministic (3rd Gen Intel® Xeon® Scalable Processor)
- Low Latency (4th Gen Intel® Xeon® Scalable Processor)

In addition, all VM Hosts must be configured with additional virtualization options, as described in Section 3.3

6.3 Configuration Dictionary - Group Variables

Table 11 lists the parameters available as group variables with their type (for example, Boolean, string, list, integer). Refer to the section that describes your Configuration Profile to see the parameters enabled for that Configuration Profile.

Table 11. Configuration Dictionary – Group Variables for VMRA

OPTIONS	TYPE
profile_name	String
configured_arch	String
vm_enabled	Boolean
preflight_enabled	Boolean
unconfirmed_cpu_models	List
project_root_dir	String
vm_recreate_existing	Boolean
retry_stagger	Integer
post_deployment_hook_enabled	Boolean
hooks_local	String

hooks_remote	String
container_runtime_only_deployment	Boolean
update_all_packages	Boolean
update_kernel	Boolean
additional_grub_parameters_enabled	Boolean
additional_grub_parameters	String
selinux_state	String
firewall_enabled	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
kubernetes	Boolean
kube_provisioner	String
kube_version	String
rke2_version	String
kube_dashboard_enabled	Boolean
cluster_name	String
cert_manager_enabled	Boolean
audit_policy_custom_rules	String
container_runtime	String
kube_controller_manager_bind_address	String
kube_proxy_metrics_bind_address	String
kube_proxy_nodeport_addresses_cidr	String
kube_pods_subnet	String
kube_service_addresses	String
kube_network_plugin	String
calico_network_backend	String
kube_network_plugin_multus	Boolean
calico_bpf_enabled	Boolean
example_net_attach_defs.sriov_net_dp	Boolean
example_net_attach_defs.userspace_ovs_dpdk	Boolean
example_net_attach_defs.userspace_vpp	Boolean
nfd_enabled	Boolean
nfd_namespace	String
nfd_sleep_interval	String
pas_namespace	String
tas_enabled	Boolean
tas_build_image_locally	Boolean
tas_enable_demo_policy	Boolean
gas_enabled	Boolean
gas_build_image_locally	Boolean
native_cpu_manager_enabled	Boolean
topology_manager_enabled	Boolean

topology_manager_policy	String
sriov_net_dp_enabled	Boolean
sriov_net_dp_namespace	String
sriov_net_dp_build_image_locally	Boolean
sriovdp_config_data	Dictionary
intel_dp_namespace	String
qat_dp_enabled	Boolean
qat_dp_verbosity	Integer
qat_dp_max_num_devices	Integer
qat_dp_build_image_locally	Boolean
allocation_policy (commented)	String
qat_supported_pf_dev_ids	List
qat_supported_vf_dev_ids	List
gpu_dp_enabled	Boolean
gpu_dp_verbosity	Integer
gpu_dp_build_image_locally	Boolean
gpu_dp_shared_devices	Integer
gpu_dp_monitor_resources	Boolean
gpu_dp_fractional_manager	Boolean
gpu_dp_prefered_allocation	String
sgx_dp_enabled	Boolean
sgx_dp_verbosity	Integer
sgx_dp_build_image_locally	Boolean
sgx_aesmd_namespace	String
sgx_aesmd_demo_enable	Boolean
sgx_dp_provision_limit	Integer
sgx_dp_enclave_limit	Integer
sgx_memory_size	Integer
sriov_network_operator_enabled	Boolean
sriov_network_operator_namespace	String
intel_ethernet_operator_enabled	Boolean
intel_ethernet_operator_flow_config_enabled	Boolean
istio_service_mesh.enabled	Boolean
istio_service_mesh.profile	String
istio_service_mesh.intel_preview.enabled	Boolean
istio_service_mesh.tcpip_bypass_ebpf.enabled	Boolean
istio_service_mesh.tls_splicing.enabled	Boolean
linkerd_service_mesh.enabled	Boolean
prometheus_operator	Boolean
collectd_enabled	Boolean
telegraf_enabled	Boolean
jaeger_operator	Boolean
opentelemetry_enabled	Boolean
elasticsearch_enabled	Boolean
kibana_enabled	Boolean

collectd_scrap_interval	Integer
telegraf_scrap_interval	Integer
cadvisor_enabled	Boolean
cadvisor_sample_perf_events_enabled	Boolean
cadvisor_pik_perf_events_enabled	Boolean
intel_power_manager.enabled	Boolean
intel_power_manager.power_nodes	List
intel_power_manager.build_image_locally	Boolean
intel_power_manager.deploy_example_pods	Boolean
intel_power_manager.global_shared_profile_enabled	Boolean
intel_power_manager.global_max_frequency	Integer
intel_power_manager.global_min_frequency	Integer
intel_power_manager.global_pstate_governor	String
openssl_engine_enabled	Boolean
kmra.sbx	Boolean
kmra.oran.enabled	Boolean
kmra.oran.local_build	Boolean
kmra.oran_netopeer2_server.enabled	Boolean
kmra.oran_netopeer2_client.enabled	Boolean
kmra.pccs.enabled	Boolean
kmra.pccs.api_key	String
kmra.apphsm.enabled	Boolean
kmra.ctk_loadkey_demo.enabled	Boolean
sigstore_policy_controller_install	Boolean
intel_oneapi_enabled	Boolean
intel_oneapi.basekit	Boolean
intel_oneapi.ai_analytics	Boolean
registry_enable	Boolean
registry_nodeport	Integer
registry_local_address	String
always_pull_enabled	Boolean
docker_registry_mirrors (commented)	List
docker_insecure_registries (commented)	List
containerd_registries (commented)	List
crio_registries (commented)	List
crio_insecure_registries (commented)	List

6.4 Configuration Dictionary - Host Variables

Table 12 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 12. Configuration Dictionary - Host Variables for VMRA

OPTIONS	TYPE
profile_name	String
configured_arch	String
configured_nic	String
iommu_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
install_dpdk	Boolean
dpdk_version	String
dpdk_local_patches_dir (commented)	String
dpdk_local_patches_strip (commented)	Integer
openssl_install	Boolean
enable_dhclient_systemd_service	Boolean
dataplane_interfaces	List
update_nic_drivers	Boolean
i40e_driver_version (commented)	String
i40e_driver_checksum (commented)	String
ice_driver_version (commented)	String
ice_driver_checksum (commented)	String
iavf_driver_version (commented)	String
iavf_driver_checksum (commented)	String
update_nic_firmware	Boolean
nvmupdate (commented)	List
install_ddp_packages	Boolean
enable_ice_systemd_service	Boolean
sriov_cni_enabled	Boolean
bond_cni_enabled	Boolean
userspace_cni_enabled	Boolean
ovs_dpdk_enabled	Boolean
ovs_version	String
ovs_dpdk_lcore_mask	String
ovs_dpdk_socket_mem	String
vpp_enabled	Boolean
native_cpu_manager_system_reserved_cpus	String
native_cpu_manager_kube_reserved_cpus	String
native_cpu_manager_reserved_cpus (commented)	String
configure_sgx	Boolean
configure_gpu	Boolean

update_qat_drivers	Boolean
configure_qat	Boolean
enabled_qat_service	String
disabled_qat_service	String
enable_qat_svm	Boolean
qat_sriov_numvfs_required	Integer
qat_vf_driver_required	String
qat_devices	List
intel_ethernet_operator.ddp_update	Boolean
intel_ethernet_operator.fw_update	Boolean
intel_ethernet_operator.node_flow_config_enabled	Boolean
intel_ethernet_operator.flow_config_dir (commented)	String
custom_sriov_network_policies_dir (commented)	String
enable_intel_pmu_plugin	Boolean
enable_pkgpower_plugin	Boolean
intel_pmu_plugin_monitored_cores	String
intel_rdt_plugin_monitored_cores	String
exclude_collectd_plugins	List
power_profiles	List
local_shared_profile.enabled	Boolean
local_shared_profile.local_max_frequency	Integer
local_shared_profile.local_min_frequency	Integer
local_shared_profile.local_pstate_governor	String
shared_workload.enabled	Boolean
shared_workload.reserved_cpus	List
shared_workload.shared_workload_type	String
uncore_frequency.enabled	Boolean
uncore_frequency.system_max_frequency	Integer
uncore_frequency.system_min_frequency	Integer
uncore_frequency.die_selector	List
cstates.enabled	Boolean
cstates.shared.C<1,6>	Boolean
cstates.profile_exclusive.balance-performance.C<1,6>	Boolean
cstates.core	Dictionary
intel_pstate_enabled	Boolean
intel_pstate	String
turbo_boost_enabled	Boolean
sst_pp_configuration_enabled	Boolean
sst_pp_config_list.sst_bf	String
sst_pp_config_list.sst_cp	String
sst_pp_config_list.sst_tf	String
sst_pp_config_list.sst_tf.online_cpus_range	String
vm_image_distribution (commented)	String
vm_image_version_ubuntu (commented)	String
vm_image_version_rocky (commented)	String

dhcp	Integer
vxlan_gw_ip	String
vm_hashed_passwd	String
vm_hashed_passwd_non_root (commented)	String
vxlan_physical_network	String
cpu_host_os (commented)	Integer
vm_cluster_name (commented)	String
vms.type	String
vms.name	String
vms.cpus (commented)	String
vms.numa (commented)	Integer
vms.cpu_total	Integer
vms.alloc_all (commented)	Boolean
vms.memory	Integer
vms.vxlan	Integer
vms.pci	List

7 VMRA Basic Configuration Profile Setup

This section contains information on how to set up a VMRA Basic Configuration Profile.

7.1 Supported Hardware

This configuration profile can be run on the following hardware platforms.

- 4th Gen Intel Xeon Scalable Processor: ARCH=spr
- 3rd Gen Intel Xeon Scalable Processor: ARCH=icx

It is recommended that the VM Hosts have memory size increased to 512 GB.

7.2 Recommended BIOS

Energy Balanced configuration is recommended. See Section 3.3 for details.

7.3 Recommended OS

Any of the supported operating systems in Section 2.3.3 can be used.

7.4 Playbook Overview

The Ansible playbook for this configuration profile allows you to provision a production-ready Kubernetes cluster. Every capability included in the playbook can be disabled or enabled.

The tables below are a summary of group and host variables, including only Boolean values. For lists showing all configurable properties, see Section 6.3 and Section 6.4.

Variables are grouped into two main categories:

- Group variables – apply to VM Host and VMs (both control and worker nodes) and have cluster-wide impact.
- Host variables – scope is limited to a single host. For VMRA there are two different set of host variables:
  - VM Host – configuration for the VM Host where VMs are deployed.
  - VM – configuration for each VM (both control and worker nodes)

7.4.1 Basic Configuration Profile Group Variables

For the list of all configurable properties, see Section 6.3.

Table 13. Basic Configuration Profile – Group Variables

OPTIONS	TYPE
vm_enabled	true
preflight_enabled	true
vm_recreate_existing	false
post_deployment_hook_enabled	false
container_runtime_only_deployment	false
update_all_packages	false
update_kernel	false
additional_grub_parameters_enabled	false
firewall_enabled	false
dns_disable_stub_listener	true
remove_kubespray_host_dns_settings	false
kubernetes	true
kube_dashboard_enabled	true
cert_manager_enabled	true
kube_network_plugin_multus	true
calico_bpf_enabled	false
example_net_attach_defs.sriov_net_dp	true
nfd_enabled	true
topology_manager_enabled	true
sriov_net_dp_enabled	true
sriov_net_dp_build_image_locally	false
sriov_network_operator_enabled	false
intel_ethernet_operator_enabled	false
intel_ethernet_operator_flow_config_enabled	false
prometheus_operator	true
collectd_enabled	false
telegraf_enabled	true
jaeger_operator	true
opentelemetry_enabled	true
elasticsearch_enabled	true
kibana_enabled	true
cadvisor_enabled	true
cadvisor_sample_perf_events_enabled	false
cadvisor_pik_perf_events_enabled	false
intel_power_manager.enabled	false
intel_power_manager.build_image_locally	false
intel_power_manager.deploy_example_pods	true
intel_power_manager.global_shared_profile_enabled	true
sigstore_policy_controller_install	false
intel_oneapi_enabled	false
intel_oneapi.basekit	false
intel_oneapi.ai_analytics	false
registry_enable	true

always_pull_enabled	false

7.4.2 Basic Configuration Profile Host Variables

For the list of all configurable properties, see Section 6.4.

Table 14. Basic Configuration Profile – Host Variables

OPTIONS	VM Host	VMs
iommu_enabled	true	true
hugepages_enabled	false	false
isolcpus_enabled	false	false
cpusets_enabled	false	false
install_dpdk	true	true
enable_dhclient_systemd_service	false	false
update_nic_drivers	true	true
update_nic_firmware	false	false
sriov_cni_enabled	true	true
intel_ethernet_operator.fw_update	false	false
intel_ethernet_operator.node_flow_config_enabled	false	false
enable_intel_pmu_plugin	false	false
enable_pkgpower_plugin		false
local_shared_profile.enabled	false	false
shared_workload.enabled	true	true
uncore_frequency.enabled	false	false
cstates.enabled	false	false
cstates.shared.C<1,6>	true	true
cstates.profile_exclusive.balance-performance.C<1,6>	false	false
intel_pstate_enabled	false	false
turbo_boost_enabled	true	false

7.5 Deploy and Validate Basic Configuration Profile Platform

Deploy the Basic Configuration Profile Ansible playbook using the steps described in Section 2.5.5.

Validate the setup of your Kubernetes cluster. Refer to the tasks in Section 5 and run the validation processes according to the hardware and software components that you have installed.

8 VMRA Build-Your-Own Configuration Profile Setup

This section contains information on how to set up a VMRA Build-Your-Own Configuration Profile.

8.1 Supported Hardware

This configuration profile can be run on the following hardware platforms.

- 4th Gen Intel Xeon Scalable Processor: ARCH=spr
- 3rd Gen Intel Xeon Scalable Processor: ARCH=icx

It is recommended that the VM Hosts have memory size increased to 512GB.

8.2 Recommended BIOS

Either Energy Balanced or Max Performance configuration is recommended. See Section 3.3 for details.

8.3 Recommended OS

Any of the supported operating systems in Section 2.3.3 can be used.

8.4 Playbook Overview

The Ansible playbook for this configuration profile allows you to provision a production-ready Kubernetes cluster. Every capability included in the playbook can be disabled or enabled.

The tables below are a summary of group and host variables, including only Boolean values. For lists showing all configurable properties, see Section 6.3 and Section 6.4.

Variables are grouped into two main categories:

- Group variables – apply to VM Host and VMs (both control and worker nodes) and have cluster-wide impact.
- Host variables – scope is limited to a single host. For VMRA there are two different set of host variables:
  - VM Host – configuration for the VM Host where VMs are deployed.
  - VM – configuration for each VM (both control and worker nodes)

8.4.1 Build-Your-Own Configuration Profile Group Variables

For the list of all configurable properties, see Section 6.3.

Table 15. Build-Your-Own Configuration Profile – Group Variables

OPTIONS	TYPE
vm_enabled	true
preflight_enabled	true
vm_recreate_existing	false
post_deployment_hook_enabled	false
container_runtime_only_deployment	false
update_all_packages	false
update_kernel	false
additional_grub_parameters_enabled	false
firewall_enabled	false
dns_disable_stub_listener	true
remove_kubespray_host_dns_settings	false
kubernetes	true
kube_dashboard_enabled	false
cert_manager_enabled	false
kube_network_plugin_multus	false
calico_bpf_enabled	false
example_net_attach_defs.sriov_net_dp	false
example_net_attach_defs.userspace_ovs_dpdk	false

example_net_attach_defs.userspace_vpp	false
nfd_enabled	false
tas_enabled	false
tas_build_image_locally	false
tas_enable_demo_policy	false
gas_enabled	false
gas_build_image_locally	false
native_cpu_manager_enabled	false
topology_manager_enabled	false
sriov_net_dp_enabled	false
sriov_net_dp_build_image_locally	false
qat_dp_enabled	false
qat_dp_build_image_locally	false
gpu_dp_enabled	false
gpu_dp_build_image_locally	false
gpu_dp_monitor_resources	false
gpu_dp_fractional_manager	false
sgx_dp_enabled	false
sgx_dp_build_image_locally	false
sgx_aesmd_demo_enable	false
sriov_network_operator_enabled	false
intel_ethernet_operator_enabled	false
intel_ethernet_operator_flow_config_enabled	false
istio_service_mesh.enabled	false
istio_service_mesh.intel_preview.enabled	false
istio_service_mesh.tcpip_bypass_ebpf.enabled	false
istio_service_mesh.tls_splicing.enabled	false
linkerd_service_mesh.enabled	false
prometheus_operator	false
collectd_enabled	false
telegraf_enabled	false
jaeger_operator	false
opentelemetry_enabled	false
elasticsearch_enabled	false
kibana_enabled	false
cadvisor_enabled	false
cadvisor_sample_perf_events_enabled	false
cadvisor_pik_perf_events_enabled	false
intel_power_manager.enabled	false
intel_power_manager.build_image_locally	false
intel_power_manager.deploy_example_pods	true
intel_power_manager.global_shared_profile_enabled	true
openssl_engine_enabled	false
kmra.sbx	false
kmra.oran.enabled	false

kmra.oran.local_build	false
kmra.oran_netopeer2_server.enabled	false
kmra.oran_netopeer2_client.enabled	false
kmra.pccs.enabled	false
kmra.apphsm.enabled	false
kmra.ctk_loadkey_demo.enabled	false
sigstore_policy_controller_install	false
intel_oneapi_enabled	false
intel_oneapi.basekit	false
intel_oneapi.ai_analytics	false
registry_enable	false
always_pull_enabled	false

8.4.2 Build-Your-Own Configuration Profile Host Variables

For the list of all configurable properties, see Section 6.4.

Table 16. Build-Your-Own Configuration Profile – Host Variables

OPTIONS	VM Host	VMs
iommu_enabled	false	false
hugepages_enabled	false	false
isolcpus_enabled	false	false
cpusets_enabled	false	false
install_dpdk	false	false
openssl_install	false	false
enable_dhclient_systemd_service	false	false
update_nic_drivers	false	false
update_nic_firmware	false	false
install_ddp_packages	false	false
enable_ice_systemd_service	false	false
sriov_cni_enabled	false	false
bond_cni_enabled	false	false
userspace_cni_enabled	false	false
ovs_dpdk_enabled	false	false
vpp_enabled	false	false
configure_sgx	false	false
configure_gpu	false	false
update_qat_drivers	false	false
configure_qat	false	false
enable_qat_svm	false	false
intel_ethernet_operator.ddp_update	false	false
intel_ethernet_operator.fw_update	false	false
intel_ethernet_operator.node_flow_config_enabled	false	false
enable_intel_pmu_plugin	false	false
enable_pkgpower_plugin		false
local_shared_profile.enabled	false	false

shared_workload.enabled	true	true
uncore_frequency.enabled	false	false
cstates.enabled	false	false
cstates.shared.C<1,6>	true	true
cstates.profile_exclusive.balance-performance.C<1,6>	false	false
intel_pstate_enabled	false	false
turbo_boost_enabled	true	false
sst_pp_configuration_enabled	false	false

8.5 Deploy and Validate Build-Your-Own Configuration Profile Platform

Deploy the Build-Your-Own Configuration Profile Ansible playbook using the steps described in Section 2.5.5.

Validate the setup of your Kubernetes cluster. Refer to the tasks in Section 5 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 information on how to set up a VMRA On-Premises Edge Configuration Profile.

9.1 Supported Hardware

This configuration profile can be run on the following hardware platforms.

- 4th Gen Intel Xeon Scalable Processor: ARCH=spr
- 3rd Gen Intel Xeon Scalable Processor: ARCH=icx

It is recommended that the VM Hosts have memory size increased to 512GB.

9.2 Recommended BIOS

Max Performance configuration is recommended. See Section 3.3 for details.

9.3 Recommended OS

Any of the supported operating systems in Section 2.3.3 can be used.

9.4 Playbook Overview

The Ansible playbook for this configuration profile allows you to provision a production-ready Kubernetes cluster. Every capability included in the playbook can be disabled or enabled.

The tables below are a summary of group and host variables, including only Boolean values. For lists showing all configurable properties, see Section 6.3 and Section 6.4.

Variables are grouped into two main categories:

- Group variables – apply to VM Host and VMs (both control and worker nodes) and have cluster-wide impact.
- Host variables – scope is limited to a single host. For VMRA there are two different set of host variables:
  - VM Host – configuration for the VM Host where VMs are deployed.
  - VM – configuration for each VM (both control and worker nodes)

9.4.1 On-Premises Edge Configuration Profile Group Variables

For the list of all configurable properties, see Section 6.3.

Table 17. On-Premises Edge Configuration Profile – Group Variables

OPTIONS	TYPE
vm_enabled	true
preflight_enabled	true
vm_recreate_existing	false
post_deployment_hook_enabled	false
container_runtime_only_deployment	false
update_all_packages	false
update_kernel	false
additional_grub_parameters_enabled	false
firewall_enabled	false
dns_disable_stub_listener	true
remove_kubespray_host_dns_settings	false
kubernetes	true
kube_dashboard_enabled	true
cert_manager_enabled	true
kube_network_plugin_multus	true
calico_bpf_enabled	false
example_net_attach_defs.sriov_net_dp	true
nfd_enabled	true

tas_enabled	true
tas_build_image_locally	false
tas_enable_demo_policy	false
native_cpu_manager_enabled	true
topology_manager_enabled	true
sriov_net_dp_enabled	true
sriov_net_dp_build_image_locally	false
qat_dp_enabled	true
qat_dp_build_image_locally	false
sgx_dp_enabled	true
sgx_dp_build_image_locally	false
sgx_aesmd_demo_enable	false
sriov_network_operator_enabled	false
intel_ethernet_operator_enabled	false
intel_ethernet_operator_flow_config_enabled	false
istio_service_mesh.enabled	true
istio_service_mesh.intel_preview.enabled	false
istio_service_mesh.tcpip_bypass_ebpf.enabled	true
istio_service_mesh.tls_splicing.enabled	true
linkerd_service_mesh.enabled	false
prometheus_operator	true
collectd_enabled	false
telegraf_enabled	true
jaeger_operator	true
opentelemetry_enabled	true
elasticsearch_enabled	true
kibana_enabled	true
cadvisor_enabled	true
cadvisor_sample_perf_events_enabled	false
cadvisor_pik_perf_events_enabled	false
intel_power_manager.enabled	false
intel_power_manager.build_image_locally	false
intel_power_manager.deploy_example_pods	true
intel_power_manager.global_shared_profile_enabled	true
openssl_engine_enabled	true
kmra.sbx	false
kmra.oran.enabled	false
kmra.oran.local_build	false
kmra.oran_netopeer2_server.enabled	false
kmra.oran_netopeer2_client.enabled	false
kmra.pccs.enabled	true
kmra.apphsm.enabled	true
kmra.ctk_loadkey_demo.enabled	true
sigstore_policy_controller_install	false
intel_oneapi_enabled	false

intel_oneapi.basekit	false
intel_oneapi.ai_analytics	false
registry_enable	true
always_pull_enabled	false

9.4.2 On-Premises Edge Configuration Profile Host Variables

For the list of all configurable properties, see Section 6.4.

Table 18. On-Premises Edge Configuration Profile – Host Variables

OPTIONS	VM Host	VMs
iommu_enabled	true	true
hugepages_enabled	true	true
isolcpus_enabled	false	false
cpusets_enabled	false	false
install_dpdk	true	true
openssl_install	true	true
enable_dhclient_systemd_service	false	false
update_nic_drivers	true	true
update_nic_firmware	false	false
sriov_cni_enabled	true	true
bond_cni_enabled	false	false
configure_sgx	true	true
update_qat_drivers	true	true
configure_qat	true	true
enable_qat_svm	false	false
intel_ethernet_operator.fw_update	false	false
intel_ethernet_operator.node_flow_config_enabled	false	false
enable_intel_pmu_plugin	false	false
enable_pkgpower_plugin		false
local_shared_profile.enabled	false	false
shared_workload.enabled	true	true
uncore_frequency.enabled	false	false
cstates.enabled	false	false
cstates.shared.C<1,6>	true	true
cstates.profile_exclusive.balance-performance.C<1,6>	false	false
intel_pstate_enabled	false	false
turbo_boost_enabled	true	false
sst_pp_configuration_enabled	false	false

9.5 Deploy and Validate On-Premises Edge Configuration Profile Platform

Deploy the On-Premises Edge Configuration Profile Ansible playbook using the steps described in Section 2.5.5.

Validate the setup of your Kubernetes cluster. Refer to the tasks in Section 5 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 information on how to set up a VMRA Remote Central Office-Forwarding Configuration Profile. A step-by-step description of how to set up an example VMRA Remote Central Office-Forwarding Configuration Profile on a single server is covered in the Network and Edge Reference System Architectures - Single Server Quick Start Guide.

10.1 Supported Hardware

This configuration profile can be run on the following hardware platforms.

- 4th Gen Intel Xeon Scalable Processor: ARCH=spr
- 3rd Gen Intel Xeon Scalable Processor: ARCH=icx

It is recommended that the VM Hosts have memory size increased to 512GB.

10.2 Recommended BIOS

Max Performance configuration is recommended. See Section 3.3 for details.

10.3 Recommended OS

Any of the supported operating systems in Section 2.3.3 can be used.

10.4 Playbook Overview

The Ansible playbook for this configuration profile allows you to provision a production-ready Kubernetes cluster. Every capability included in the playbook can be disabled or enabled.

The tables below are a summary of group and host variables, including only Boolean values. For lists showing all configurable properties, see Section 6.3 and Section 6.4.

Variables are grouped into two main categories:

- Group variables – apply to VM Host and VMs (both control and worker nodes) and have cluster-wide impact.
- Host variables – scope is limited to a single host. For VMRA there are two different set of host variables:
  - VM Host – configuration for the VM Host where VMs are deployed.
  - VM – configuration for each VM (both control and worker nodes)

10.4.1 Remote Central Office-Forwarding Configuration Profile Group Variables

For the list of all configurable properties, see Section 6.3.

Table 19. Remote Central Office-Forwarding Configuration Profile – Group Variables

OPTIONS	TYPE
vm_enabled	true
preflight_enabled	true
vm_recreate_existing	false
post_deployment_hook_enabled	false
container_runtime_only_deployment	false
update_all_packages	false
update_kernel	false
additional_grub_parameters_enabled	false
firewall_enabled	false
dns_disable_stub_listener	true
remove_kubespray_host_dns_settings	false
kubernetes	true
kube_dashboard_enabled	true
cert_manager_enabled	true
kube_network_plugin_multus	true
calico_bpf_enabled	false
example_net_attach_defs.sriov_net_dp	true

example_net_attach_defs.userspace_ovs_dpdk	false
example_net_attach_defs.userspace_vpp	false
nfd_enabled	true
tas_enabled	true
tas_build_image_locally	false
tas_enable_demo_policy	false
native_cpu_manager_enabled	true
topology_manager_enabled	true
sriov_net_dp_enabled	true
sriov_net_dp_build_image_locally	false
qat_dp_enabled	true
qat_dp_build_image_locally	false
sgx_dp_enabled	true
sgx_dp_build_image_locally	false
sgx_aesmd_demo_enable	false
sriov_network_operator_enabled	false
intel_ethernet_operator_enabled	false
intel_ethernet_operator_flow_config_enabled	false
istio_service_mesh.enabled	false
istio_service_mesh.intel_preview.enabled	false
istio_service_mesh.tcpip_bypass_ebpf.enabled	false
istio_service_mesh.tls_splicing.enabled	false
linkerd_service_mesh.enabled	false
prometheus_operator	true
collectd_enabled	true
telegraf_enabled	false
jaeger_operator	false
opentelemetry_enabled	false
elasticsearch_enabled	false
kibana_enabled	false
cadvisor_enabled	false
cadvisor_sample_perf_events_enabled	false
cadvisor_pik_perf_events_enabled	false
intel_power_manager.enabled	false
intel_power_manager.build_image_locally	false
intel_power_manager.deploy_example_pods	true
intel_power_manager.global_shared_profile_enabled	true
openssl_engine_enabled	true
kmra.sbx	false
kmra.oran.enabled	false
kmra.oran.local_build	false
kmra.oran_netopeer2_server.enabled	false
kmra.oran_netopeer2_client.enabled	false
kmra.pccs.enabled	false
kmra.apphsm.enabled	false

kmra.ctk_loadkey_demo.enabled	false
sigstore_policy_controller_install	false
intel_oneapi_enabled	false
intel_oneapi.basekit	false
intel_oneapi.ai_analytics	false
registry_enable	true
always_pull_enabled	false

10.4.2 Remote Central Office-Forwarding Configuration Profile Host Variables

For the list of all configurable properties, see Section 6.4.

Table 20. Remote Central Office-Forwarding Configuration Profile – Host Variables

OPTIONS	VM Host	VMs
iommu_enabled	true	true
hugepages_enabled	true	true
isolcpus_enabled	false	false
cpusets_enabled	false	false
install_dpdk	true	true
openssl_install	true	true
enable_dhclient_systemd_service	false	false
update_nic_drivers	true	true
update_nic_firmware	false	false
install_ddp_packages	false	false
enable_ice_systemd_service	true	false
sriov_cni_enabled	true	true
bond_cni_enabled	false	false
userspace_cni_enabled	false	false
ovs_dpdk_enabled	false	false
vpp_enabled	false	false
configure_sgx	true	true
update_qat_drivers	true	true
configure_qat	true	true
enable_qat_svm	false	false
intel_ethernet_operator.ddp_update	false	false
intel_ethernet_operator.fw_update	false	false
intel_ethernet_operator.node_flow_config_enabled	false	false
enable_intel_pmu_plugin	false	false
enable_pkgpower_plugin		false
local_shared_profile.enabled	false	false
shared_workload.enabled	true	true
uncore_frequency.enabled	false	false
cstates.enabled	false	false
cstates.shared.C<1,6>	true	true
cstates.profile_exclusive.balance-performance.C<1,6>	false	false
intel_pstate_enabled	false	false

turbo_boost_enabled	true	false
sst_pp_configuration_enabled	false	false

10.5 Deploy and Validate Remote Central Office-Forwarding Configuration Profile Platform

Deploy the Remote Central Office-Forwarding Configuration Profile Ansible playbook using the steps described in Section 2.5.5.

Validate the setup of your Kubernetes cluster. Refer to the tasks in Section 5 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 information on how to set up a VMRA Regional Data Center Configuration Profile.

11.1 Supported Hardware

This configuration profile can be run on the following hardware platforms.

- 4th Gen Intel Xeon Scalable Processor: ARCH=spr
- 3rd Gen Intel Xeon Scalable Processor: ARCH=icx

It is recommended that the VM Hosts have memory size increased to 512GB.

11.2 Recommended BIOS

Max Performance configuration is recommended. See Section 3.3 for details.

11.3 Recommended OS

Any of the supported operating systems in Section 2.3.3 can be used.

11.4 Playbook Overview

The Ansible playbook for this configuration profile allows you to provision a production-ready Kubernetes cluster. Every capability included in the playbook can be disabled or enabled.

The tables below are a summary of group and host variables, including only Boolean values. For lists showing all configurable properties, see Section 6.3 and Section 6.4.

Variables are grouped into two main categories:

- Group variables – apply to VM Host and VMs (both control and worker nodes) and have cluster-wide impact.
- Host variables – scope is limited to a single host. For VMRA there are two different set of host variables:
  - VM Host – configuration for the VM Host where VMs are deployed.
  - VM – configuration for each VM (both control and worker nodes)

11.4.1 Regional Data Center Configuration Profile Group Variables

For the list of all configurable properties, see Section 6.3.

Table 21. Regional Data Center Configuration Profile – Group Variables

OPTIONS	TYPE
vm_enabled	true
preflight_enabled	true
vm_recreate_existing	false
post_deployment_hook_enabled	false
container_runtime_only_deployment	false
update_all_packages	false
update_kernel	false
additional_grub_parameters_enabled	false
firewall_enabled	false
dns_disable_stub_listener	true
remove_kubespray_host_dns_settings	false
kubernetes	true
kube_dashboard_enabled	true
cert_manager_enabled	true
kube_network_plugin_multus	true
calico_bpf_enabled	false
example_net_attach_defs.sriov_net_dp	true
nfd_enabled	true

tas_enabled	true
tas_build_image_locally	false
tas_enable_demo_policy	false
gas_enabled	false
gas_build_image_locally	false
native_cpu_manager_enabled	true
topology_manager_enabled	true
sriov_net_dp_enabled	true
sriov_net_dp_build_image_locally	false
gpu_dp_enabled	false
gpu_dp_build_image_locally	false
gpu_dp_monitor_resources	false
gpu_dp_fractional_manager	false
sgx_dp_enabled	true
sgx_dp_build_image_locally	false
sgx_aesmd_demo_enable	false
sriov_network_operator_enabled	false
intel_ethernet_operator_enabled	false
intel_ethernet_operator_flow_config_enabled	false
istio_service_mesh.enabled	true
istio_service_mesh.intel_preview.enabled	false
istio_service_mesh.tcpip_bypass_ebpf.enabled	true
istio_service_mesh.tls_splicing.enabled	true
linkerd_service_mesh.enabled	false
prometheus_operator	true
collectd_enabled	false
telegraf_enabled	true
jaeger_operator	true
opentelemetry_enabled	true
elasticsearch_enabled	true
kibana_enabled	true
cadvisor_enabled	true
cadvisor_sample_perf_events_enabled	false
cadvisor_pik_perf_events_enabled	false
intel_power_manager.enabled	false
intel_power_manager.build_image_locally	false
intel_power_manager.deploy_example_pods	true
intel_power_manager.global_shared_profile_enabled	true
kmra.sbx	false
kmra.oran.enabled	false
kmra.oran.local_build	false
kmra.oran_netopeer2_server.enabled	false
kmra.oran_netopeer2_client.enabled	false
kmra.pccs.enabled	true
kmra.apphsm.enabled	true

kmra.ctk_loadkey_demo.enabled	true
sigstore_policy_controller_install	false
intel_oneapi_enabled	false
intel_oneapi.basekit	false
intel_oneapi.ai_analytics	false
registry_enable	true
always_pull_enabled	false

11.4.2 Regional Data Center Configuration Profile Host Variables

For the list of all configurable properties, see Section 6.4.

Table 22. Regional Data Center Configuration Profile – Host Variables

OPTIONS	VM Host	VMs
iommu_enabled	true	true
hugepages_enabled	false	false
isolcpus_enabled	false	false
cpusets_enabled	false	false
install_dpdk	true	true
enable_dhclient_systemd_service	false	false
update_nic_drivers	true	true
update_nic_firmware	false	false
sriov_cni_enabled	true	true
configure_sgx	true	true
configure_gpu	false	false
intel_ethernet_operator.fw_update	false	false
intel_ethernet_operator.node_flow_config_enabled	false	false
enable_intel_pmu_plugin	false	false
enable_pkgpower_plugin		false
local_shared_profile.enabled	false	false
shared_workload.enabled	true	true
uncore_frequency.enabled	false	false
cstates.enabled	false	false
cstates.shared.C<1,6>	true	true
cstates.profile_exclusive.balance-performance.C<1,6>	false	false
intel_pstate_enabled	false	false
turbo_boost_enabled	true	false

11.5 Deploy and Validate Regional Data Center Configuration Profile

Deploy the Regional Data Center Configuration Profile Ansible playbook using the steps described in Section 2.5.5.

Validate the setup of your Kubernetes cluster. Refer to the tasks in Section 5 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 23.07 Release Updates

New Components/Features:

- Support for Intel® SGX by upgrading QEMU and libvirt
- Support for KMRA as Intel® SGX is available
- Intel® SGX signer enabled for Istio Service Mesh

Updates/Changes:

- Kubespray* is provided via ansible-galaxy collection instead of git submodule
- Implement support and option for Intel® QuickAssist Technology (Intel® QAT) in-tree versus out-of-tree drivers and libraries
- RHEL 9.2 as base OS on VM host
- Ubuntu 22.04.2 as base OS on both VM host and VMs
- Improved VMRA deployment stability
- Version upgraded for the majority of Reference System components (See User Guide for complete BOM and versions)
    - Notable updates:
        - Kubernetes to v1.26.3
        - Service Mesh Istio to v1.18.1 or v1.18.0-intel.0
        - Data Plane Development Kit (DPDK) to v23.05
        - Open vSwitch with DPDK to 3.11
        - OpenSSL* to openssl-3.1.0

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

- N/A

Removed Support:

- Discontinued supporting Cloud Native Data Plane (CNDP)
- Discontinued supporting RHEL 9.0 as base OS

Known Limitations/Restrictions:

- Only in-tree Intel® QuickAssist Technology (Intel® QAT) and Intel® Ethernet Network Adapter E810 drivers supported on RHEL 9.2
- UserSpace CNI with VPP is not supported

A.2 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 Reference System 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

[2] Workloads and configurations. Results may vary.

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.3 VMRA 22.11.1 Release Notes

New Components/Features:

- N/A (same as VMRA Release 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 VMRA Release 22.11)

Removed Support:

- N/A (same as VMRA Release 22.11)

Known Limitations/Restrictions:

- N/A (same as RA22.11)

A.4 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.

A.5 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.6 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 10 (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
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
TLS	Transport Layer Security

ABBREVIATION	DESCRIPTION
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

0723/DN/WIT/PDF 721803-009US