USER MANUAL

Intel Corporation

Network and Cloud Edge Reference System Architectures Portfolio Release v22.06

Guided Deployments of Validated Systems at Specific Network Locations using 2nd, 3rd, and 4th Gen Intel® Xeon® Scalable processor and Intel® Xeon® D processor Platforms

Authors

Francis Cahill

Octavia Carotti

Calin Gherghe

Joel A. Gibson

Veronika Karpenko

Dana Nehama

Michael O'Reilly

Abhijit Sinha

Daniel Ugarte

1 Introduction

1.1 Purpose

The Network and Cloud Edge Reference System Architectures provide a common platform that addresses the growing need for a simple and accelerated cloud-native platform designed to support diverse use cases across network locations, such as on-premises edge, access edge, and 5G core. This user manual provides system definitions (hardware, software, and configurations) and installation instructions for the reference architectures to support key workloads in edge to 5G-core deployments.

The delivered reference systems integrate and validate Intel® platforms and open-source software with Intel best-known configurations and practices using a cloud-native approach. As a result, forward-looking, cloud-native applications and implementations can quickly be delivered with confidence.

1.2 Audience and Scope

This Network and Cloud Edge Reference System Architectures Portfolio targets developers and deployment engineers. The documentation for Network and Cloud Edge Reference System Architectures Portfolio includes this user manual and individual deployment user guides for the available deployment models. Current deployment models include the following:

- **Container bare metal Reference System Architecture (BMRA)** for containers on bare metal. Kubernetes manages the clusters. This model allows an application-ready cluster to be built of multiple servers for cloud-native applications.
- **Virtual machine Reference System Architecture (VMRA)** for virtual clusters implemented with or without Kubernetes. This model allows a virtual cluster to be built on a single system based on VMs running Kubernetes containers, ready for cloud-native applications.

This user manual describes concepts and components for the Network and Cloud Edge Reference System Architectures, including BMRA and VMRA hardware components and incorporated cloud native and Kubernetes software technologies. Reading this user manual gives you the background needed to understand the deployment concepts, components available, and processes required for deployments based on different network locations.

User guides with detailed deployment instructions are available for BMRA and VMRA. These user guides prescribe component bills of material (BOMs) and simple, step-by-step, automated deployment procedures using Ansible Scripts. In addition, examples for installation validation are available.

Note: Some capabilities supported by the reference architectures are only available under NDA. Contact your Intel representative for access to the NDA material.

Table of Contents

1 Introduction 1
1.1 Purpose 1
1.2 Audience and Scope 1
1.3 Portfolio Architecture 5
1.3.1 Key Terms 6
1.3.2 Intel Investments of Capabilities 7
1.4 Deployment Process 8

2 Reference System Architectures Overview 8
2.1 Deployment Options 8
2.2 Key Elements 10
2.3 Configuration Profiles 11
2.3.1 Configuration Profiles Overview 11
2.3.2 On-Premises Edge Configuration Profile (Customer Premises) 12
2.3.3 Access Edge Configuration Profile (Far Edge) 12
2.3.4 Remote Central Office-Forwarding Configuration Profile (Near Edge) 12
2.3.5 Regional Data Center Configuration Profile (Regional POP) 12
2.3.6 Non-Location-Specific Configuration Profiles 12
2.4 Ansible Playbooks 13

3 Hardware Components 14
3.1 4th Gen Intel Xeon Scalable Processor Capabilities 16

4 Software Components 17
4.1 Kubernetes Features 17
4.2 Platform System Features 18
4.3 Container Runtimes 18
4.4 Kubernetes Plugins 19
4.4.1 Multus CNI 19
4.4.2 SR-IOV Network Device Plugin 19
4.4.3 SR-IOV CNI 19
4.4.4 Userspace CNI 19
4.4.5 Bond CNI 19
4.4.6 Intel® QuickAssist Device Plugin 20
4.4.7 Intel® Software Guard Extensions (Intel® SGX) Device Plugin 20
4.4.8 Node Feature Discovery 20
4.4.9 Topology Manager 20
4.4.10 Kubernetes Native CPU Manager 20
4.4.11 Platform Aware Scheduling 21
4.5 Istio Service Mesh 21
4.5.1 Bypass TCP/IP Stack Using eBPF 22
4.5.2 TLS Splicing 22
4.5.3 Istio Acceleration with Intel AVX-512 Crypto (Examples Available Under NDA) 22
4.5.4 Istio Acceleration and Compression with QAT 2.0 (Examples Available Under NDA) 22
4.5.5 Istio mTLS Private Key Protection with SGX 22
4.5.6 Istio Multi-tenancy 22
4.6 Kubernetes Operators 22
4.6.1 SR-IOV Network Operator 22
4.6.2 Intel Device Plugins Operator 23
4.6.3 MinIO Operator 23
4.6.4 Intel® Ethernet Operator 23
4.6.5 Forward Error Correction (FEC) Operator 24
4.7 Dynamic Device Personalization (DDP) 24
4.7.1 DDP on Intel Ethernet 700 Series Network Adapters 24
4.7.2 DDP on Intel® Ethernet 800 Series Network Adapters 25
4.8 Kubernetes Power Manager 26
4.9 Intel® Speed Select Technology 26
4.9.1 Intel Speed Select Technology – Base Frequency 26
4.9.2 Intel® Speed Select Technology – Core Power 27
4.9.3 Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF) 27
4.9.4 Intel® Speed Select Technology – Performance Profile (Intel® SST-PP) 27
4.10 Security 27
4.10.1 Native Kubernetes Cluster Security Features 27
4.10.2 Intel® Security Libraries for Data Center (Intel® SecL – DC) 28
4.10.3 Intel® Software Guard Extensions 28
4.10.4 Key Management Reference Application with Intel® SGX 28

4.10.5 OpenSSL and QAT Engine 29
4.10.6 Service Mesh 29
4.10.7 Trusted Certificate Service 29
4.10.8 Trusted Attestation Controller 29
4.11 Observability 29
4.11.1 Telemetry Insight Reports 30
4.12 Object Storage 30
4.13 Cloud Native Data Plane 31
4.14 TADK 31
4.15 Access Edge - vRAN 32
4.15.1 Generic vRAN 32
4.15.2 FlexRAN™ Software Deployment (NDA) 32

5 Reference Architecture Installation 32
5.1 Introduction 32
5.2 Installation Overview 33
5.2.1 Operating System Selection 33
5.2.2 Network Interface Requirements 33
5.2.3 Software Prerequisites for Ansible Host, Control Nodes, and Worker Nodes 33
5.3 Ansible Playbook Review 33
5.3.1 Ansible Playbook Phases 35
5.4 Deployment Using Ansible Playbooks 36
5.4.1 Prepare Target Servers 36
5.4.2 Get Ansible Playbook and Prepare Configuration Templates 36
5.4.3 Update Ansible Inventory File 36
5.4.4 Update Ansible Host and Group Variables 36
5.4.5 Run Ansible Cluster Deployment Playbook 36
5.4.6 Ansible Cluster Removal Playbook 37

Appendix A Abbreviations 38

Appendix B Reference Documentation 42

Figures

Figure 1. Landscape for Network and Cloud Edge Reference System Architectures Portfolio 5
Figure 2. Network and Cloud Edge Reference System Architectures Portfolio Hierarchy 5
Figure 3. Configuration Profile Implementation Examples 6
Figure 4. Container Bare Metal Reference System Architecture (BMRA) 9
Figure 5. VMRA – Virtual Cluster on a Single Node 9
Figure 6. VMRA – Virtual Cluster on Multiple Nodes 10
Figure 7. BMRA for Storage Architecture 10
Figure 8. Reference Architecture Key Elements – BMRA Example 11
Figure 9. BMRA Ansible Playbook Overview 13
Figure 10. VMRA Ansible Playbook Overview 14
Figure 11. Generic Kubernetes CRI stack 18
Figure 12. Deprecated Kubernetes Runtime Stack Using Docker 19
Figure 13. Storage Node Deployment Example 23
Figure 14. Intel Ethernet Operator Architecture Overview 24
Figure 15. CPU Core Frequency Deployment Methods 26
Figure 16. Key Management Reference Application Infrastructure with Intel® SGX 29
Figure 17. Telemetry Insight Reports High-Level Architecture 30
Figure 18. Cloud Native Data Plane Architecture 31
Figure 19. TADK Enhanced Implementation of ModSecurity WAF 32
Figure 20. BMRA Ansible Playbooks – Full Configuration Profile Example 34
Figure 21. VMRA Ansible Playbook – Full Configuration Profile Example 35

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 Options for Control Node – 4th Gen Intel Xeon Scalable Processor 14
Table 5. Hardware Components for Worker Node Plus – 4th Gen Intel Xeon Scalable Processor 14
Table 6. Hardware Components for Worker Node Plus (Access Edge - vRAN) – 4th Gen Intel Xeon Scalable Processor 15
Table 7. Hardware Components for Storage Node – 3rd Gen Intel Xeon Scalable Processor 15
Table 8. Silicon Capabilities of 4th Gen Intel Xeon Scalable Processor 16
Table 9. DDP Default Package Profiles 25

Table 10. Abbreviations 38
Table 11. Reference Documents 42

Document Revision History

Three previous editions of the BMRA document were released, starting in April 2019.

- Covered 2nd Gen Intel® Xeon® Scalable processors
- Covered 2nd and 3rd Gen Intel® Xeon® Scalable processors
- Covered 2nd and 3rd Gen Intel® Xeon® Scalable processors and Intel® Xeon® D processor

REVISION	DATE	DESCRIPTION
001	February 2022	Initial release.
002	March 2022	Updated a few URLs.
003	May 2022	Covers the 4th Gen Intel® Xeon® Scalable processor (formerly codenamed Sapphire Rapids).
004	June 2022	The changes include updates to the discussion of the BMRA for Storage Deployment Model and Figure 2.
005	June 2022	Updated a few URLs.
006	July 2022	Added NDA support for FlexRAN™ software, updated Istio and service mesh features.
007	July 2022	The changes include updates to the discussion of the Access Edge Configuration Profile and to Figures 1 and 3.

1.3 Portfolio Architecture

Throughout an infrastructure—whether a private enterprise or public cloud—various optimized systems are needed based on the workloads being run and the sites serviced (Figure 1). The Network and Cloud Edge Reference System Architectures Portfolio prescribes deployment solutions optimized to support services based on the location in the infrastructure and the typical applications/workloads served at that location. These solutions can be deployed quickly using validated, best-known configurations (BKCs) and automated deployment tools provided in the Reference System Architectures. The configurations and tools were designed for the network location and a chosen deployment model (containerized or virtual) to deliver a predictive solution for performance-, security-, usability-, and cost-optimized operations.

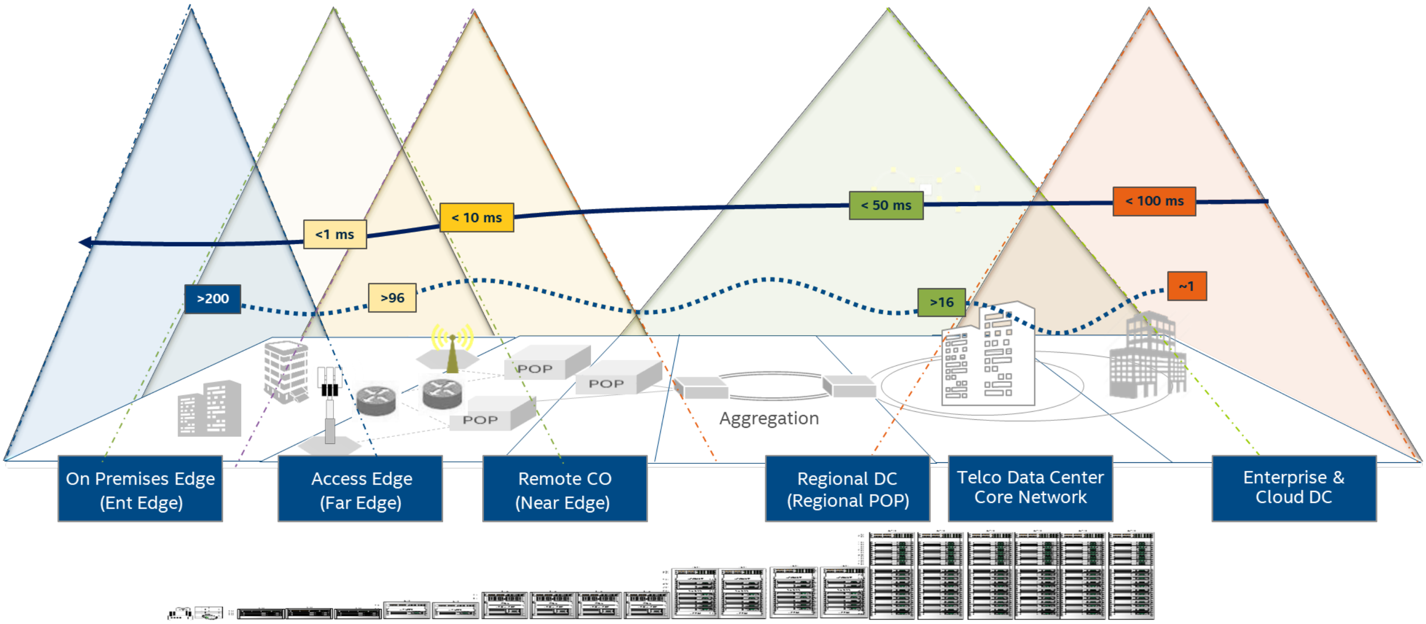
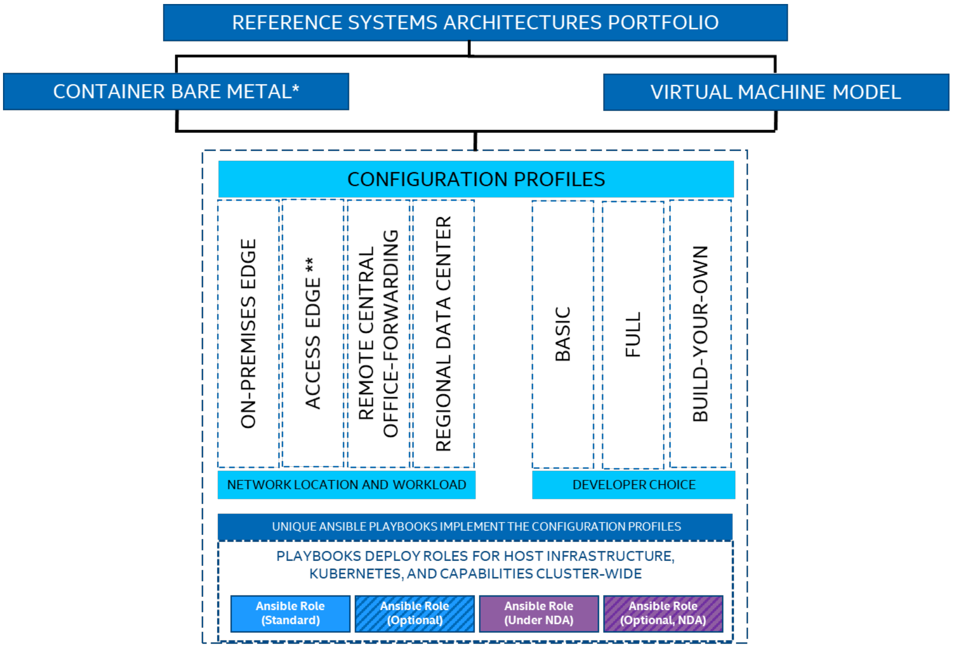


Figure 1. Landscape for Network and Cloud Edge Reference System Architectures Portfolio

The architectural hierarchy for the Network and Cloud Edge Reference System Architectures Portfolio is shown in Figure 2. The VMRA and BMRA implement Configuration Profiles that provide a recipe for implementing hardware, software, and optimized configuration per the network location. Figure 3 illustrates an example of Configuration Profiles designed for network locations.



REFERENCE SYSTEMS ARCHITECTURES PORTFOLIO
CONTAINER BARE METAL*
VIRTUAL MACHINE MODEL
CONFIGURATION PROFILES
ON-PREMISES EDGE
ACCESS EDGE **
REMOTE CENTRAL OFFICE-FORWARDING
REGIONAL DATA CENTER
BASIC
FULL
BUILD-YOUR-OWN
NETWORK LOCATION AND WORKLOAD
DEVELOPER CHOICE
UNIQUE ANSIBLE PLAYBOOKS IMPLEMENT THE CONFIGURATION PROFILES
PLAYBOOKS DEPLOY ROLES FOR HOST INFRASTRUCTURE, KUBERNETES, AND CAPABILITIES CLUSTER-WIDE
Ansible Role (Standard)
Ansible Role (Optional)
Ansible Role (Under NDA)
Ansible Role (Optional, NDA)

* Includes Container Bare Metal with Storage Support
**Container Bare Metal Reference Architecture Only

Figure 2. Network and Cloud Edge Reference System Architectures Portfolio Hierarchy

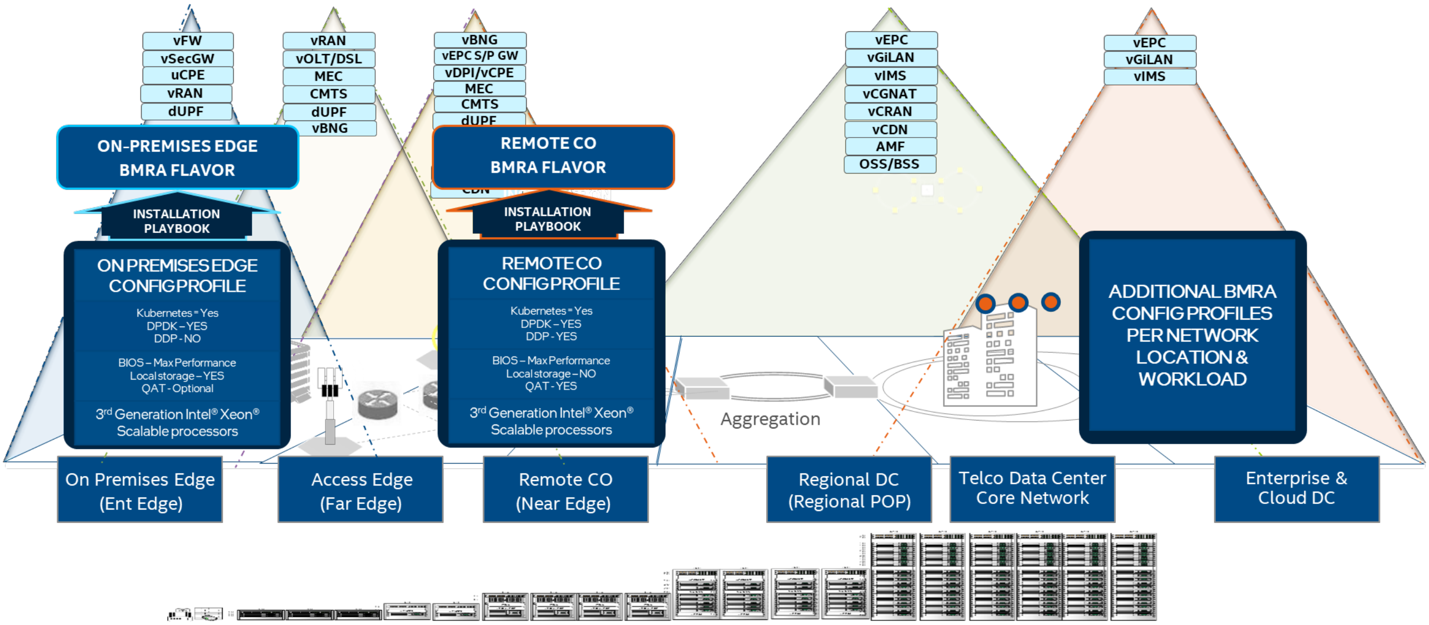


Figure 3. Configuration Profile Implementation Examples

1.3.1 Key Terms

Table 1 lists the key terms used throughout the portfolio. These terms are specific to Network and Cloud 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 Cloud Edge Reference System Architectures Portfolio	A templated system-level blueprint for a range of locations in enterprise and cloud infrastructure with automated deployment tools. The portfolio integrates the latest Intel platforms and cloud-native technologies for multiple deployment models to simplify and accelerate deployments of key workloads across a service infrastructure.
Deployment Model	Provides flexibility to deploy solutions according to IT needs. The portfolio offers two deployment models: • **Container Bare Metal Reference System Architecture (BMRA)** – A deployment model of a Kubernetes cluster with containers on a bare metal platform. A special version of BMRA is the **BMRA for Storage** that supports MinIO Object Storage. • **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.
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 Access Edge). Configuration Profiles define the components for optimized performance, usability, and cost per network location and workload needs.[1] In addition, generic Configuration Profiles are available for developers' flexible deployments.
Reference Architecture Flavor	A Reference Architecture deployment using a Configuration Profile.
Ansible Playbook	A set of validated scripts that prepare, configure, and deploy a Reference Architecture Flavor by implementing a Configuration Profile.

[1] Workloads and configurations. Results may vary.

TERM	DESCRIPTION
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 2nd, 3rd, and 4th Gen Intel® Xeon® Scalable processors plus the Intel® Xeon® D processor. These platforms include the Taylors Falls Reference Design. 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

1.3.2 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.[2]
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.

[2] Workloads and configurations. Results may vary.

CAPABILITY	BENEFIT
Security	Intel security technologies help ensure platform and transport security. These technologies including 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 secure key management.

1.4 Deployment Process

This user manual provides the foundational information for deployments. After the concepts here are understood, refer to one of the following, depending on the intended deployment scenario.

- Kubernetes containers deployments: Network and Cloud Edge Container Bare Metal Reference System Architecture User Guide
- Virtualized system deployments: Network and Cloud Edge Virtual Machine Reference System Architecture User Guide

Additionally, Intel provides Experience Kits that detail the technologies enabled in the Reference System Architectures Portfolio, including benchmark information. Experience Kits are available on Intel® Network Builders:

- Network Transformation Experience Kits
- Container Experience Kits
- Intel® Xeon® D-2700 and D-1700 Processor Experience Kits

For NDA material, contact your local Intel representative.

2 Reference System Architectures Overview

2.1 Deployment Options

The Network and Cloud Edge Reference System Architectures Portfolio offers two deployment options: BMRA and VMRA. These options are designed for the convergence of key applications and services, control plane, and high-performance packet processing functions in an infrastructure. Tools such as `testpmd` and `pktgen` pods and sample Cloud Native Network Function (CNF) workloads (vCMTS and vBNG) are used for validation of the Reference System Architectures delivered.

- **The BMRA** is a scalable, open Kubernetes cluster composed of physical worker and control nodes networked through switches and connected as depicted in Figure 4.
- **The VMRA** is a scalable virtual cluster supporting 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 within the single node as depicted in Figure 5 and on multiple nodes as depicted in Figure 6.
- **The BMRA for Storage** is a BMRA Flavor that delivers a MinIO Object Storage solution. Its architecture is shown in Figure 7.

The BMRA and VMRA are tuned to support diverse deployment scenarios. Based on the application and network location, a BMRA or VMRA can be configured for optimal operation and performance using one of the provided Configuration Profiles.[3] Ansible playbooks are defined based on those Configuration Profiles, allowing fast and easy auto-configuration and auto-provisioning of the Reference System Architectures.

The BMRA for Storage defines a storage cluster composed of two or more servers, with attached storage media, to serve clients with persistent data requests – read and write. This option focuses exclusively on object storage using MinIO, where the singular workload for the storage solution is the MinIO application.

[3] Workloads and configurations. Results may vary.

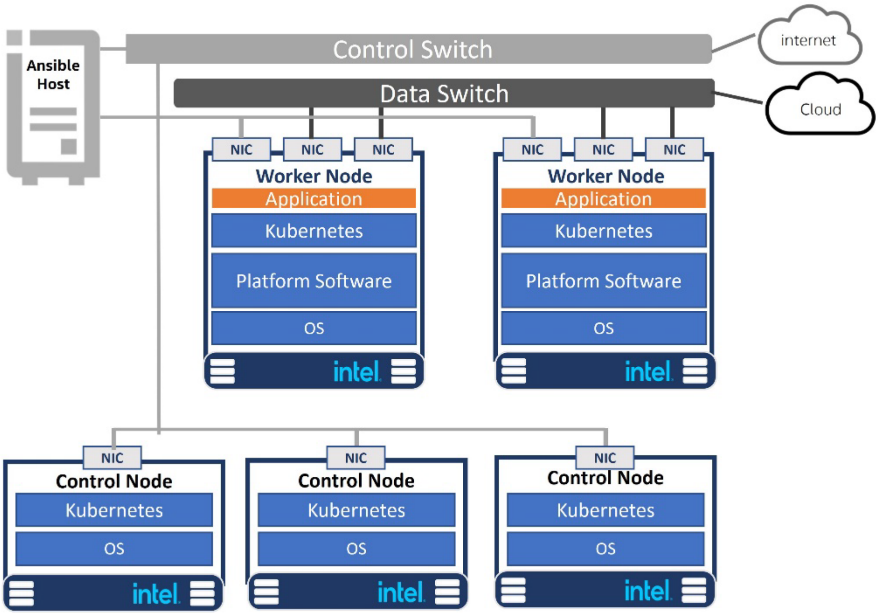
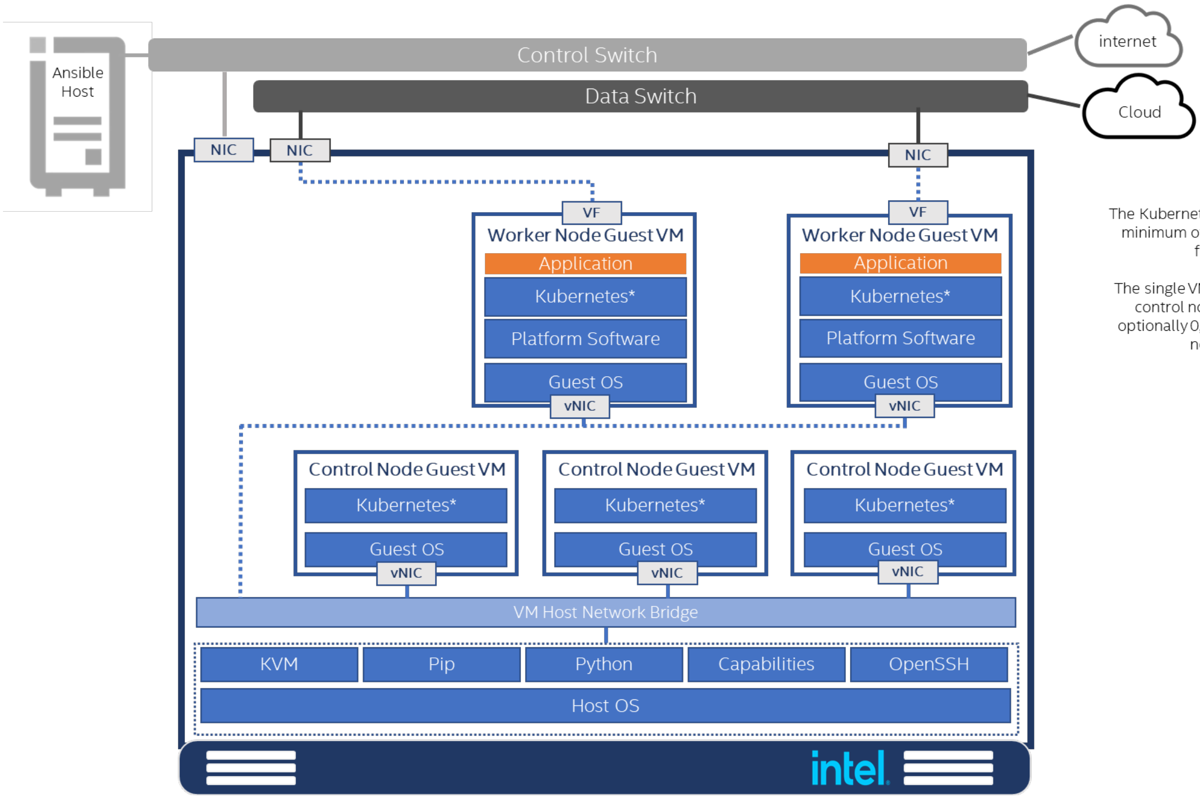


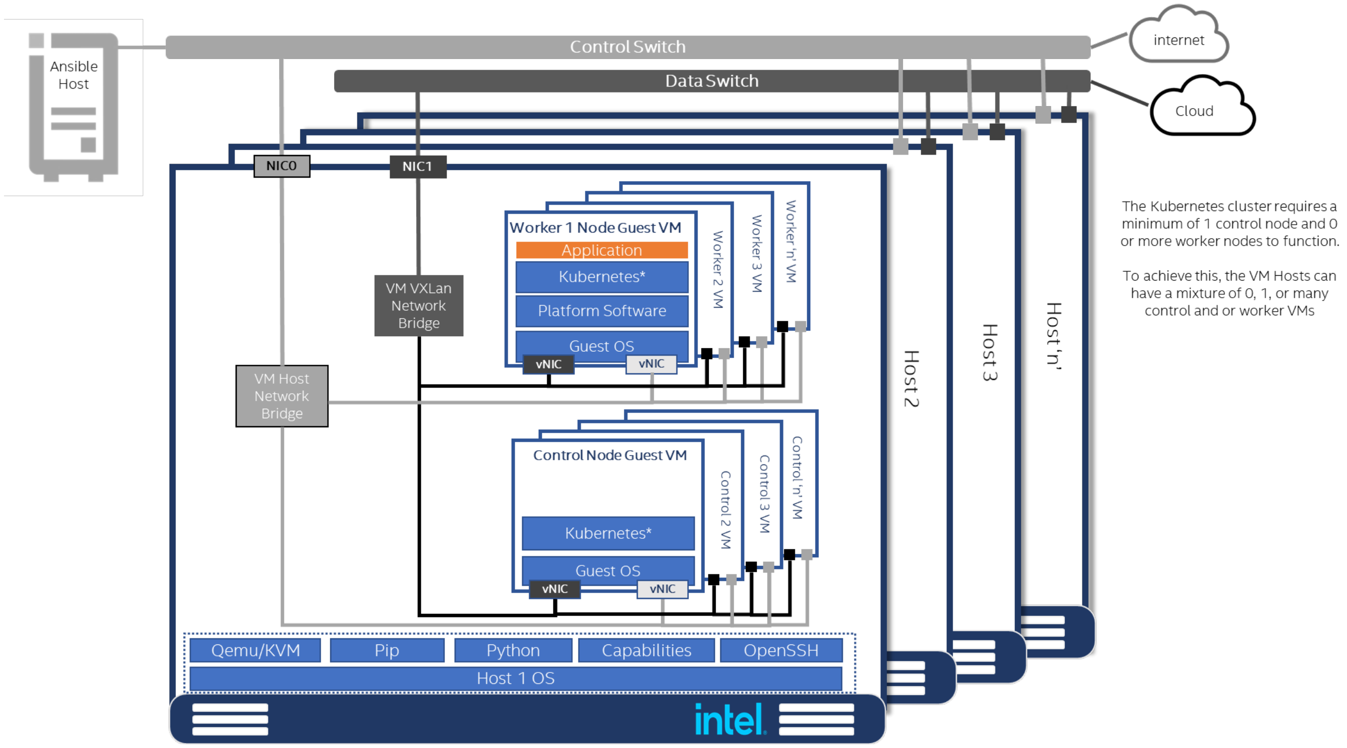
Figure 4. Container Bare Metal Reference System Architecture (BMRA)



The Kubernetes cluster requires a minimum of 1 control node to function.

The single VM Host must have 1 control node VM and then optionally 0, 1, or many worker node VMs

Figure 5. VMRA – Virtual Cluster on a Single Node



The Kubernetes cluster requires a minimum of 1 control node and 0 or more worker nodes to function.

To achieve this, the VM Hosts can have a mixture of 0, 1, or many control and or worker VMs

Figure 6. VMRA – Virtual Cluster on Multiple Nodes

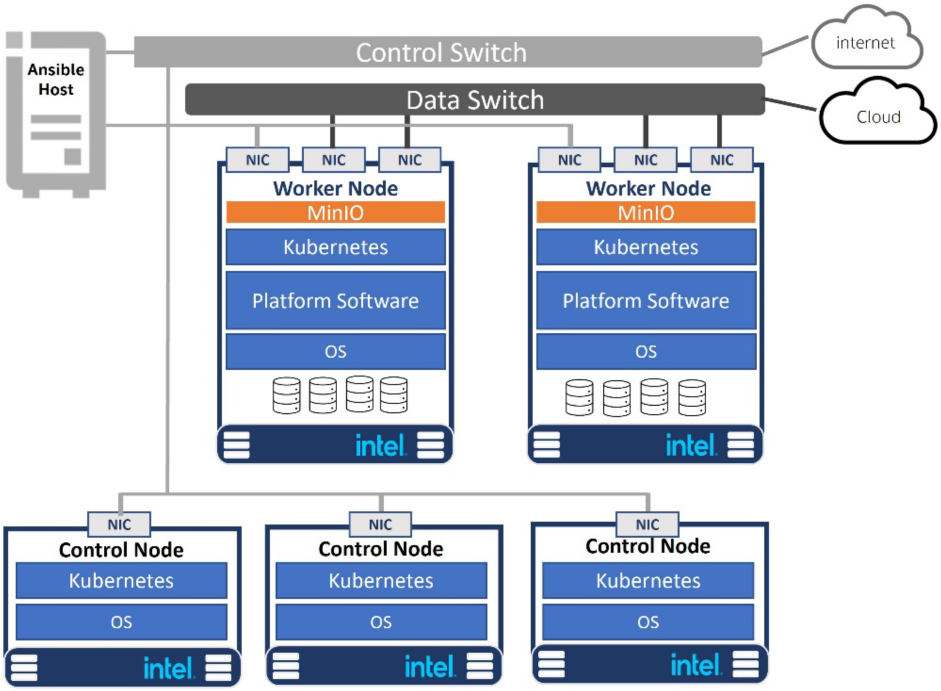


Figure 7. BMRA for Storage Architecture

2.2 Key Elements

The main elements of the Reference System Architectures include the following (refer to Figure 8 for a BMRA example):

- **Hardware Components**: Multiple platform hardware options are available, including a variety of 4th, 3rd, and 2nd Gen Intel® Xeon® Scalable processor SKUs, Intel® Xeon® D processor SKUs, Intel® Ethernet Network Adapters, Intel® QAT, and Intel® Server GPU. BIOS options are listed in each Reference System Architecture user guide. Deployment engineers should refer to the appropriate user guide during deployment to select and configure optimal BIOS values before cluster provisioning. **For details, see Section 3 – Hardware Components.**

- **Software Capabilities**: The software capabilities are based on open-source software delivered by cloud-native and CNCF communities driving Kubernetes, Istio, observability, DPDK, FD.io. OVS, OVS-DPDK, and through Intel GitHub. Options for RHEL and Ubuntu Linux operating systems are available. Container environments use Docker container runtime as well as containerd and CRI-O. **For details, see Section 4 – Software Components**.
- **Configuration Profiles**: Specific hardware and software configurations are provided in the Configuration Profiles based on Intel assessment and verification. Hardware configurations address two performance capabilities: "base" and "plus".
- **Installation Playbooks**: Ansible playbooks implement the best-practice configuration and setup for each Configuration Profile to ease and accelerate installation. For more details about the hardware and software configuration options available and ready to use, see Section 5 – Installation Playbooks.

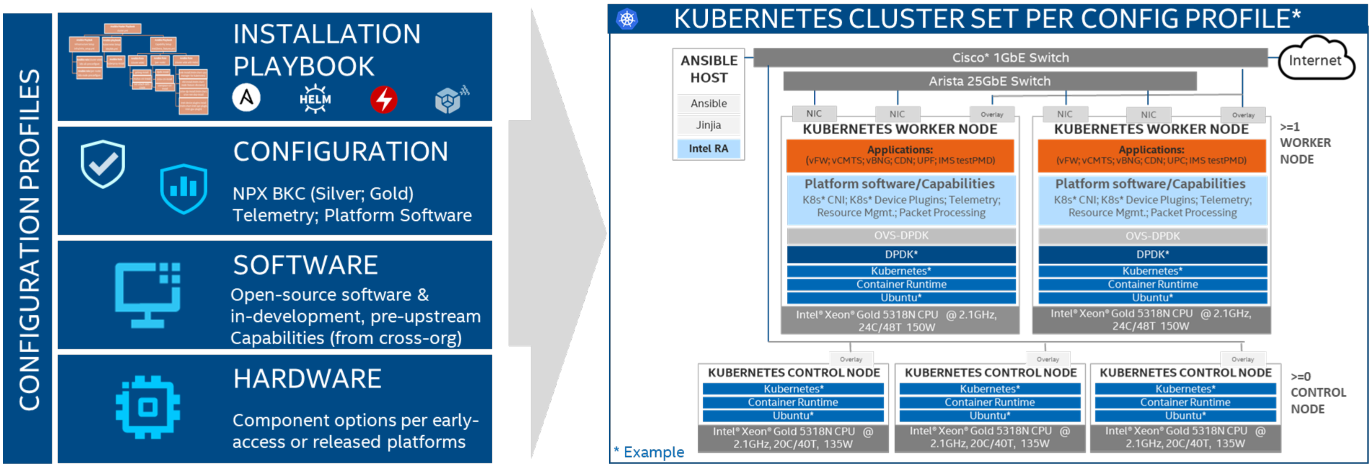

CONFIGURATION PROFILES
INSTALLATION PLAYBOOK
HELM
CONFIGURATION
NPX BKC (Silver; Gold)
Telemetry; Platform Software
SOFTWARE
Open-source software &
in-development, pre-upstream
Capabilities (from cross-org)
HARDWARE
Component options per early-access or released platforms
KUBERNETES CLUSTER SET PER CONFIG PROFILE*
ANSIBLE HOST
Ansible
Jinjia
Intel RA
Cisco* 1GbE Switch
Internet
Arista 25GbE Switch
NIC
NIC
Overlay
KUBERNETES WORKER NODE
Applications:
(vFW; vCMTS; vBNG; CDN; UPF; IMS testPMD)
Platform software/Capabilities
K8s* CNI; K8s* Device Plugins; Telemetry; Resource Mgmt.; Packet Processing
OVS-DPDK
DPDK*
Kubernetes*
Container Runtime
Ubuntu*
Intel® Xeon® Gold 5318N CPU @ 2.1GHz, 24C/48T 150W
NIC
NIC
Overlay
KUBERNETES WORKER NODE
Applications:
(vFW; vCMTS; vBNG; CDN; UPC; IMS testPMD)
Platform software/Capabilities
K8s* CNI; K8s* Device Plugins; Telemetry; Resource Mgmt.; Packet Processing
OVS-DPDK
DPDK*
Kubernetes*
Container Runtime
Ubuntu*
Intel® Xeon® Gold 5318N CPU @ 2.1GHz, 24C/48T 150W
>=1 WORKER NODE
Overlay
KUBERNETES CONTROL NODE
Kubernetes*
Container Runtime
Ubuntu*
Intel® Xeon® Gold 5318N CPU @ 2.1GHz, 20C/40T, 135W
Overlay
KUBERNETES CONTROL NODE
Kubernetes*
Container Runtime
Ubuntu*
Intel® Xeon® Gold 5318N CPU @ 2.1GHz, 20C/40T, 135W
Overlay
KUBERNETES CONTROL NODE
Kubernetes*
Container Runtime
Ubuntu*
Intel® Xeon® Gold 5318N CPU @ 2.1GHz, 20C/40T, 135W
>=0 CONTROL NODE
* Example

Figure 8. Reference Architecture Key Elements – BMRA Example

2.3 Configuration Profiles

2.3.1 Configuration Profiles Overview

A Configuration Profile describes a specific hardware and software BOM and configurations, applicable for a specific deployment (Figure 3). Configuration Profiles take into consideration the best-known configuration (BKC) validated by Intel for optimized performance.[4]

Each Configuration Profile includes installation scripts to deploy the required components for a Reference Architecture Flavor.

Installation scripts are available to deploy the required components for a Reference Architecture Flavor. Each Reference Architecture 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 BMRA deployment is populated with the maximum available network interface cards (NICs).
- **Software BOM** leverages the Intel platform and enables cloud-native adoption.
- **Installation (Ansible) Playbook** automates the installation of a Reference Architecture Flavor per a Configuration Profile specification.

This version of the Network and Cloud Edge Reference System Architectures Portfolio includes the following Configuration Profiles that support per network locations.

- On-Premises Edge Configuration Profile (Customer Premises)
- Access Edge (Far Edge)
- Remote Central Office-Forwarding Configuration Profile (Near Edge)
- Regional Data Center Configuration Profile (Regional POP)

Additional Configuration Profiles support non-location-specific deployments: Basic, Full, and the newly introduced Build-Your-Own Configuration Profile.

Note: The BMRA with Storage is available for deployment using a dedicated Storage Configuration Profile.

[4] Workloads and configurations. Results may vary.

2.3.2 On-Premises Edge Configuration Profile (Customer Premises)

This Configuration Profile is designed for a 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 include hospitals, factory floors, law enforcement, media, cargo transportation, and 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 Smart City deployments, CDN, and Ad-insertion. This Configuration Profile is designed to meet the following:

- Optimized data processing (to gain insights and reduce upstream transmission volumes) is priority
- Power efficiency is important in most use cases
- Automated infrastructure orchestration capabilities are mandatory

2.3.3 Access Edge Configuration Profile (Far Edge)

The Access Edge configuration profile sets an edge computing infrastructure primarily targeting cellular access network. This Configuration Profile is designed for a small, secure data processing cluster for high-speed and low latency radio access service. These nodes represent a large portion of the total number of network elements for an operator with the equipment possibly being in an outside plant in harsh, minimally controlled temperature cabinets, making power-management a necessity. The reference architecture uses the Access Edge configuration to set up the infrastructure for virtualized radio access networks (vRAN) and FlexRAN™ software solutions.

This Configuration Profile is designed to meet the following:

- Optimal balance of power and packet performance for the given application
- Platform and network security
- Low Latency with High-speed packet processing
- Automated infrastructure orchestration capabilities are mandatory

2.3.4 Remote Central Office-Forwarding Configuration Profile (Near Edge)

This Configuration Profile is designed for clusters ranging from a half rack to a few racks of servers, typically in a pre-existing, repurposed, unmanned structure. Usage scenarios include running latency-sensitive applications near the user (for example, real-time gaming, stock trading, video conferencing).

This Configuration Profile addresses an installation using Kubernetes cluster hardware, software capabilities, and configurations that help enable high performance for packet forwarding. This category could contain workloads, such as UPF, vBNG, vCMTS, and vCDN. This Configuration Profile is designed to meet the following:

- Power efficiency is a requirement
- Resource usage efficiency and sharing are a priority
- Scaling and redundancy/reliability through scale-out; performance is defined at the cluster level
- Multitenancy support may be required
- Operational automation is important (site is unmanned)

2.3.5 Regional Data Center Configuration Profile (Regional POP)

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 content delivery, media, mobile connectivity, and cloud services. This Configuration Profile is exclusively tailored and defined for Media Visual Processing workloads such as CDN Transcoding. It is designed to meet the following:

- Automation is mandatory due to scale
- High connectivity (in the aggregate - not individual data plane performance)
- Scaling and redundancy/reliability through scale-out
- Multitenancy support may be required

2.3.6 Non-Location-Specific Configuration Profiles

In addition, the following Configuration Profiles are not specific to network location. They are provided to allow maximum configuration flexibility:

- **Basic Configuration Profile:** Minimum hardware and software capabilities required to support a BMRA Kubernetes cluster setup
- **Full Configuration Profile:** Complete hardware and software capabilities offered in a BMRA
- **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.

Note: The BMRA with Storage model is implemented using the Storage Configuration Profile. This version of BMRA is a Kubernetes-native, high performance object storage setup that supports deploying MinIO tenants onto private and public cloud infrastructures ("Hybrid" Cloud).

2.4 Ansible Playbooks

Intel provides a set of Ansible Playbooks using Ansible scripts and Helm charts that enable easy and fast automatic installation[5] of a BMRA and VMRA. The Ansible playbooks enable users to customize multiple parameters to fit their installation requirements. Each of the Ansible playbooks executes the configuration defined by a Configuration Profile.

- Ansible is an agentless configuration management tool that uses playbooks to perform actions on many machines.
- Helm is a package manager tool that runs on top of Kubernetes to automate the installation process of plugins and Kubernetes capabilities.

Installation is done using one of the top-level Ansible playbooks and includes the following phases:

1. **Infrastructure Setup**: Addresses the initial system configuration, including telemetry setup.
   When deploying a VMRA, the infrastructure setup is run on both the host and in the virtual machines.
2. **Kubernetes Setup**: Deploys Kubernetes capabilities and Kubernetes add-ons via Kubespray.
3. **Capability Setup:** Installation and configuration of the system capabilities, with some using Helm charts.

Figure 9 provides a high-level view of the **BMRA Ansible Playbook.**

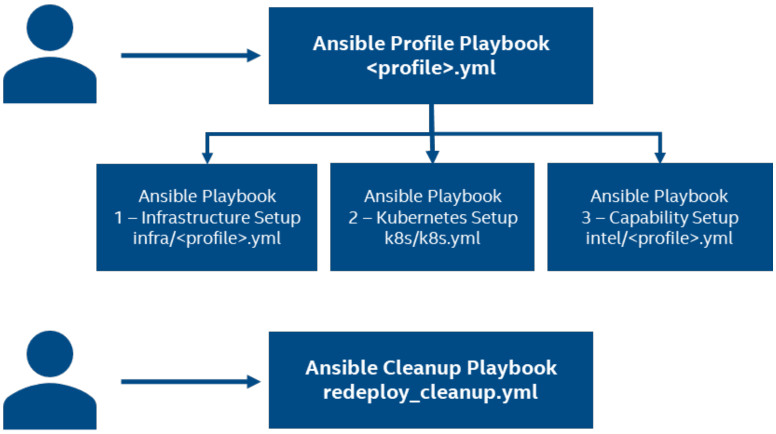


Figure 9. BMRA Ansible Playbook Overview

Figure 5 illustrates the **VMRA Ansible Playbook**. This playbook leverages some of the functionality from BMRA but includes an additional overlay Ansible playbook for setting up the VM infrastructure. Deployment includes the steps outlined in Figure 10. Note that a VMRA can be created without Kubernetes.

[5] Workloads and configurations. Results may vary.

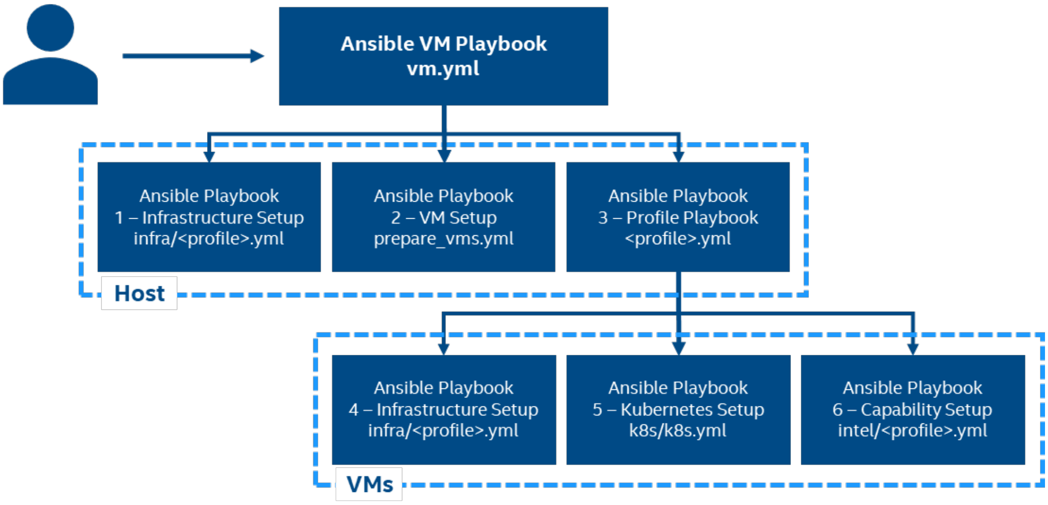


Figure 10. VMRA Ansible Playbook Overview

For more details about a playbook, refer to Section 3 of the appropriate user guide: Network and Cloud Edge Container Bare Metal Reference System Architecture User Guide and Network and Cloud Edge Virtual Machine Reference System Architecture User Guide.

3 Hardware Components

The BMRA and VMRA support a range of hardware that enables the different Configuration Profiles and deployment models. For all Configuration Profiles, possible hardware components for different nodes, as well as the BIOS components available, are listed in the appropriate user guide.

When implementing a Kubernetes cluster, the cluster typically consists of multiple worker nodes managed by one or more Kubernetes control nodes. Table 4, Table 5, and Table 6 show example listings of the hardware options for control and worker nodes. Worker nodes can be in the "base" configuration, or, if your configuration needs improved processing, you may choose to use the "plus" configuration instead.

Note: This release of VMRA does not support the Intel Xeon D processor.

Table 4. Hardware Options for Control Node – 4th Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
4th Gen Intel Xeon Scalable processors	Intel® Xeon® Gold 5418N processor at 2.0 GHz, 24 C/48 T, 165 W, or higher number Intel® Xeon® Gold or Platinum CPU SKU	Required
Memory	DRAM only configuration: 256 GB DRAM (16x 16 GB DDR5)	Required
Network Adapter	Intel® Ethernet Network Adapter E810-CQDA2 or E810-XXVDA2	Required
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)	Recommended
LAN on Motherboard (LOM)	10 Gbps or 25 Gbps port for Preboot Execution Environment (PXE) and Operation, Administration, and Management (OAM)	Required
	1/10 Gbps port for Management Network Adapter	Required
Additional Plug-in cards	N/A	

Table 5. Hardware Components for Worker Node Plus – 4th Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
4th Gen Intel Xeon Scalable processors	Intel® Xeon® Gold 6428N processor at 1.8 GHz, 32 C/64 T, 185 W	Required
Memory	Option 1: DRAM only configuration: 512 GB (16 x 32 GB DDR5)	Required
	Option 2: DRAM only configuration: 512 GB (32 x 16 GB DDR5)	
Intel® QAT	Integrated in the processor	Required

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
Intel® Optane™ Persistent Memory	Option 1: 1 TB (8x 128 GB Intel® Optane™ persistent memory in 8+4 Topology)	Recommended
	Option 2: 2 TB (16x 128 GB Intel® Optane™ persistent memory in 8+8 Topology)	
Network Adapter	Option 1: Intel® Ethernet Network Adapter E810-CQDA2	Required
	Option 2: Intel® Ethernet Network Adapter E810-2CQDA2	
Storage (Boot Drive)	Intel® SATA Solid State Drive D3 S4510 at 480 GB or equivalent boot drive	Required
Storage (Capacity)	Intel® SSD D7-P5510 Series at 4 TB or equivalent drive (recommended NUMA aligned)	Recommended
LAN on Motherboard (LOM)	10 Gbps or 25 Gbps port for Preboot Execution Environment (PXE) and Operation, Administration, and Management (OAM)	Required
	1/10 Gbps port for Management Network Adapter	Required
Additional Plug-in cards	Intel® Server Graphics 1 card	Optional

Table 6. Hardware Components for Worker Node Plus (Access Edge - vRAN) – 4th Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
4th Gen Intel Xeon Scalable processors	Intel® Xeon®-SP Gold 6421N 32 C/ 64 T 1.8 GHz 185 W, or higher number Intel® Xeon® Gold or Platinum CPU SKU	Required
Memory	DRAM only configuration: 128 GB DRAM (8 x 16 GB DDR5)	Required
Intel® Optane™ Persistent Memory	1 TB (8x 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 E810-2CQDA2	
	Option 3: Intel® Ethernet Network Adapter E810-XXVAM-DA4	
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	Intel® vRAN Accelerator ACC100/200 Adapter	Required

Table 7. Hardware Components for Storage Node – 3rd Gen Intel Xeon Scalable Processor

INGREDIENT	REQUIREMENT	REQUIRED/ RECOMMENDED
3rd Gen Intel Xeon Scalable processors	Intel® Xeon® Gold 6338N CPU @ 2.2 GHz 32 C/64 T, 185 W, or higher number Intel® Xeon® Gold or Platinum CPU SKU	Required
Memory	Option 1: DRAM only configuration: 512 GB (16 x 32 GB DDR4, 2666 MHz)	Required
	Option 2: DRAM only configuration: 512 GB (32 x 16 GB DDR4, 2666 MHz)	
Intel® QAT	Intel® C620 Series Chipset integrated on base board Intel® C627/C628 Chipset, integrated with NUMA connectivity to each CPU or minimum 16 Peripheral Component Interconnect Express (PCIe) lane connectivity to one CPU	Required
Intel® Optane™ Persistent Memory	512GB (4 x 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 E810-2CQDA2	
Storage (Boot Drive)	Intel® SATA Solid State Drive D3 S4510 at 480 GB or equivalent boot drive	Required
Storage (Capacity)	Kioxia CM6 3.2 TB NVMePCIe4x4 2.5"15mm SIE 3DWPD - KCM6XVUL3T20	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

3.1 4th Gen Intel Xeon Scalable Processor Capabilities

Table 8 provides a list of the silicon capabilities of the 4th Gen Intel® Xeon® Scalable processor. For more information about the features enabled in the Reference System Architectures, see the appropriate user guide.

Note: Some of the features listed are only available under NDA.

Table 8. Silicon Capabilities of 4th Gen Intel Xeon Scalable Processor

	FEATURE	DESCRIPTION
CPU/Accelerator	IAX	In-Memory Analytics accelerator provides effective bandwidth and fast offload by enhancing deeper compression than software-only techniques. IAX performs computationally demanding scan and filter operations in place of cores.
	QAT	Intel QuickAssist Technology is a set of accelerators for acceleration of bulk crypto, compression, and public key cryptography. Intel QAT provides acceleration for network security protocols such as IPSec and TLS, web and storage compression, and a more secure key protection for encrypted customer keys.
	DLB	Intel® Dynamic Load Balancer (Intel® DLB) offloads queue management from IA cores and provides dynamic, flow-aware balancing and reordering. It is primarily designed to help ensure balanced workload distribution, provide ultra-fast throughput and resource allocation, and reduce I/O and memory footprint.
	DSA	Intel® Data Streaming Accelerator (Intel® DSA) helps to enhance performance for workloads reliant on data movement by offloading data movement instructions to dedicated engines, allowing core cycles to be prioritized for other operations.
	CLDEMOTE	In Intel® architecture, L3 cache is shared across all cores and is non-exclusive to L2 and L1 cache. The CLDEMOTE instruction allows cache to be demoted from the L1 and L2 cache structure into the L3 shared cache. This allows data to be accessible by other cores and reduces the latency in snooping lower-level caches.
	MOVDIRI	MOVDIRI is an instruction for higher-throughput writes to memory. It is a more proficient mechanism for software to issue 4B/8B writes to MMIO-mapped register devices and is designed to improve streaming accelerator usages.
Power Management	Intel SST-PP /Intel SST-TF	The 4th Gen Intel Xeon Scalable processor introduces an enhanced set of Intel Speed Select technologies. Intel® Speed Select Technology – Performance Profile (Intel® SST-PP) provides a set of performance profiles, while Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF) looks to optimize opportunistic excursions into turbo frequencies.
	Power Instructions – UMONITOR, UMWAIT, TPAUSE	These are a set of instructions designed to transition into a power optimized state, helping enable fast entry and exit latency from power states. UMONITOR instruction is ordered as a load operation with respect to other memory transactions, while UMWAIT instructs the processor to enter an implementation-dependent optimized state while monitoring a range of addresses. The TPAUSE functionality is similar to monitor-less UMWAIT.
Security	SGX	Intel® Software Guard Extensions (Intel® SGX) provides fine-grain data protection via application isolation in memory. Integrity provides more resistance against physical attacks.
	CryptoDev and CryptoNI	The cryptodev library provides a crypto device framework for management and provisioning of hardware and software crypto poll mode drivers, defining generic APIs that support several different crypto operations. The framework currently only supports cipher, authentication, chained cipher/authentication, and AEAD symmetric and asymmetric crypto operations.
RAS	RAS	Reliability, Availability, and Serviceability is a cornerstone of a highly performant network. The 4th Gen Intel Xeon Scalable processor provides a set of features to enhance platform resiliency, debuggability, manageability, and health monitoring. Telemetry further builds on the set of features to provide reporting from multiple layers of hardware and software.
ISA	FP-16 (5G ISA)	5G ISA provides around 40 new floating-point instructions to enhance the existing range of Intel® Advanced Vector Extensions (Intel® AVX) instructions. These instructions are specific to network workloads that require greater precision for RAN-type algorithms and allow for simplified instructions calls.

	FEATURE	DESCRIPTION
	AMX (TMUL)	Advance Matrix Multiply is a scalable grid of computes designed to increase the volume of work done per instruction. Tile Multiply (TMUL) also provides a set of multiply instructions specific to Intel AVX enhancements.
	VP2INTERSECT	VP2INTERSECT is a new instruction for Intel® Advanced Matrix Extensions 512 (Intel® AVX-512) for acceleration of sparse matrices multiplication. It generates controls from SIMD indices that Intel AVX-512 data can efficiently process using the controls.
I/O	CXL 1.1	Compute Express Link is an enhanced IO interface that provides a high-bandwidth bus for inter-device transfer. In addition, it also unlocks sharing of resources between devices.
	PCIe Gen5	PCIe Gen5, introduced in the 4th Gen Intel Xeon Scalable processor spec, provides around double the bandwidth from the previous generation.
Virtualization	S-IOV	Intel® Scalable I/O Virtualization (Intel® Scalable IOV) is an accelerator for communication between VMs and containers and I/O devices such as PCIe. Building on the SR-IOV architecture, it provides incremental scalability and higher performance and allows multiple guest OSes running simultaneously within a single platform to natively share PCIe devices like the DSA and NICs.
	SVM	Shared Virtual Memory enables accelerator devices to operate in CPU virtual address space. It provides a consistent view of memory across host application and offloaded tasks. Offload operations can support data structures with pointers/references to other data structures. SVM avoids memory pinning and copying (marshalling) overheads. It supports I/O page fault through the IOMMU to enable memory over-commit via demand paging for both CPU and device access to memory.

4 Software Components

Intel, with its partners and the hardware and developer communities, continues to enable Intel® architecture advancements for the cloud native ecosystem through support of **Kubernetes features**, extension of **Kubernetes plugins**, and development of **system hardware resources**.

Note: This release of VMRA does not support the Intel Xeon D processor.

4.1 Kubernetes Features

Kubernetes (K8s) is a leading open-source orchestration platform for automating deployment, scaling, and management of containerized applications. To enhance Kubernetes for network functions virtualization (NFV) and networking usage, Intel and its partners are developing capabilities and methodologies that expose Intel® architecture platform features for increased and predictive application and network performance[6]. Such features include the following:

- **Multus** enables support of multiple network interfaces per pod to expand the networking capability of Kubernetes. Supporting multiple network interfaces is a key requirement for many virtual network functions (VNFs), as they require separation of control, management, and data planes. Multiple network interfaces are also used to support different protocols or software stacks and different tuning and configuration requirements.
- **Node Feature Discovery** (NFD) enables generic hardware capability discovery in Kubernetes, including Intel Xeon processor-based hardware.
- **Bond CNI** allows for aggregating multiple network interfaces into one logical interface.
- **Platform Aware Scheduling** (PAS) contains a group of related projects designed to expose platform-specific attributes to the Kubernetes scheduler.
- **Native CPU Manager** provides mechanisms for CPU core pinning and isolation of containerized workloads.
- **Topology Manager**, a native component of Kubernetes (starting with v1.16), enables other kubelet components to make resource allocation decisions using topology-based information.
- **Hugepages** support, added to Kubernetes v1.8, enables the discovery, scheduling, and allocation of hugepages as a native first-class resource. This support addresses low latency and deterministic memory access requirements.
- **Device plugins**, including Intel QuickAssist Technology, DSA, and SR-IOV device plugins, help boost performance and platform efficiency.

[6] Workloads and configurations. Results may vary.

- **Kubernetes Operators** are software extensions to Kubernetes that use custom resources to manage applications and their components. Operators follow Kubernetes principles for resource automation, enabling automated installation.

4.2 Platform System Features

With the Kubernetes capabilities mentioned earlier, Intel is constantly developing new platform-level capabilities for enhanced and predictive application and network performance.

- **Dynamic Device Personalization (DDP)** is one of the key technologies of the Intel® Ethernet 700 and 800 Series. It enables workload-specific optimizations using the programmable packet processing pipeline to support a broader range of traffic types.
- **Cloud-Native Data Plane (CNDP)** provides a set of userspace libraries for accelerating packet processing for cloud applications. CNDP aims to accomplish better performance than that of Linux standard network interfaces by taking advantage of Intel technologies.
- **Single Root Input/Output Virtualization (SR-IOV)** provides I/O virtualization that makes a single PCIe device (typically a NIC) appear as many network devices in the Linux kernel. In Kubernetes, this results in network connections that can be separately managed and assigned to different pods.
- **Intel® Advanced Vector Extensions 512 (Intel® AVX-512)** promotes 512-bit SIMD instruction extensions. They include extensions of the Intel AVX family of SIMD instructions but are encoded using a new encoding scheme. This scheme supports 512-bit vector registers (up to 32 vector registers in 64-bit mode) and conditional processing using opmask registers.
- **Intel® Speed Select Technology – Base Frequency (Intel® SST-BF)** offers a deterministic higher-frequency pool of processing cores to execute high priority workloads and a pool of cores running at lower frequency for non-critical workloads.
- **Intel® Speed Select Technology – Core Power (Intel® SST-CP)** allows an OS/VMM to control which cores can take advantage of any power headroom that is available in the system to help enable greater system performance.
- **Intel® Speed Select Technology – Performance Profile (Intel® SST-PP)** unlocks greater flexibility for defining core throughput by allowing greater granularity of core performance and a more effective balance between core count, thermals, and frequency.
- **Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF)** provides an ability to select a pool of cores to opportunistically access additional Turbo Mode frequencies (P0) for high throughput workloads.
- **Secure Cloud Native Network Platforms** – Using Intel® Security Libraries for Data Center (Intel® SecL – DC), Reference System Architectures can help build end-to-end platform security. Intel SecL - DC integrates platform attestation into the cloud-native architecture and uses Kubernetes to orchestrate and run workloads only on trusted pods.
- **Key Management Reference Application (KMRA) with Intel® Software Guard Extensions (Intel® SGX)** demonstrates the integration of the asymmetric key capability of Intel SGX with a third-party hardware security model (HSM) on a centralized key server.
- **Intel® QAT Engine for OpenSSL** is an option in the libcrypto general purpose cryptographic library. This implementation supports secure applications by directing the requested cryptographic operations to the hardware or software capability present on the underlying platform.
- **Media** processing on the Intel® Server GPU card. The Intel Server GPU is the first discrete graphics processing unit for data centers based on the Intel Xe architecture with support for MPEG-2, AVC, HEVC, and VP9 transcoding, plus AV1 decode.

4.3 Container Runtimes

Containers are increasingly important in cloud computing and fundamental to cloud native adoption. A container is lightweight, agile, and portable, and it can be quickly created, updated, and removed. Kubernetes is a leading open-source system for automating deployment, scaling, and management of containerized applications.

A container runtime is software that allows Kubernetes to create and manage containers on a node (see Figure 11). The runtime usually consists of two parts: a high-level runtime that implements the container runtime interface (CRI) specification used by Kubernetes, and a low-level runtime that creates and runs the container processes on request through the Open Container Initiative (OCI) specification.

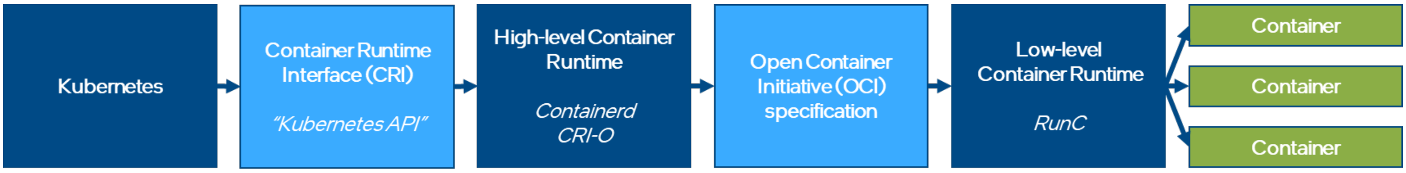

Kubernetes
Container Runtime Interface (CRI)
"Kubernetes API"
High-level Container Runtime
Containerd
CRI-O
Open Container Initiative (OCI) specification
Low-level Container Runtime
RunC
Container
Container
Container

Figure 11. Generic Kubernetes CRI stack

One exception is Docker, which is more of a container platform or engine that also works as a stand-alone tool for running and managing containers (see Figure 12).

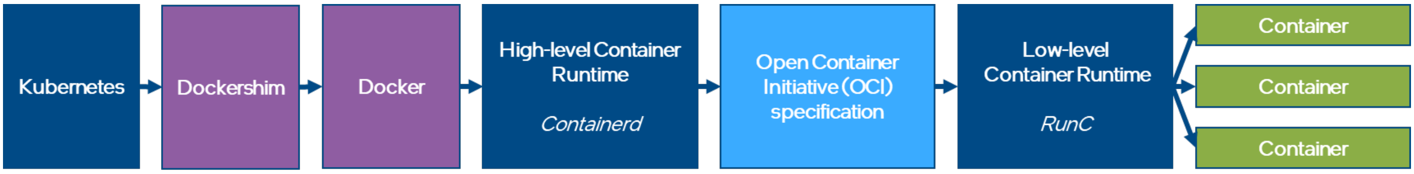

Kubernetes
Dockershim
Docker
High-level Container Runtime
Containerd
Open Container Initiative (OCI) specification
Low-level Container Runtime
RunC
Container
Container
Container

Figure 12. Deprecated Kubernetes Runtime Stack Using Docker

Note: Docker was deprecated as a runtime for Kubernetes after v1.20[7].

4.4 Kubernetes Plugins

The following device plugins are used to advertise Intel® architecture system hardware resources to Kubernetes. Several of the plugins are container network interfaces (CNIs), which are explained here: CNI.

4.4.1 Multus CNI

Kubernetes natively supports only a single network interface. Multus is a CNI plugin specifically designed to support multiple networking interfaces in a Kubernetes environment. Supporting multiple network interfaces is a key requirement for many network functions (NFs), as they require separation of control, management, and data planes. Multiple network interfaces are also used to support different protocols or software stacks and different tuning and configuration requirements. Operationally, Multus behaves as a broker and arbiter of other CNI plugins, meaning it invokes other CNI plugins (such as Flannel, Calico, SR-IOV, or Userspace CNI) to do the actual work of creating the network interfaces. Note that the plugins SR-IOV and Userspace and Bond CNIs can be installed as part of the reference architecture deployment using an Ansible playbook. Multus v3.3 has been integrated with KubeVirt, officially recognized as a CNCF project, and officially released starting with Kubespray v2.12.

For more details, see: Multus CNI.

4.4.2 SR-IOV Network Device Plugin

Single Root Input/Output Virtualization (SR-IOV) provides I/O virtualization that makes a single PCIe device (typically a network adapter) appear as many network devices, also known as virtual functions (VFs) in the Linux kernel. In Kubernetes, this results in network connections that can be separately managed and assigned to different pods.

The Intel SR-IOV network device plugin discovers and exposes SR-IOV network resources as consumable extended resources in Kubernetes. It works with SR-IOV VFs with both kernel drivers and DPDK drivers. When a VF is attached with a kernel driver, then the SR-IOV CNI plugin can be used to configure this VF in the pod. When using the DPDK driver, the containerized application configures this VF as required.

The SR-IOV network device plugin provides a way to filter the available VFs and make them available as endpoints in Kubernetes. Using a list of selectors, a subset of VFs can be associated with a resource name that can be used when assigning resources to pods.

For more details, see: SR-IOV Network Device Plugin.

4.4.3 SR-IOV CNI

The SR-IOV device plugin enables partition of a single physical network adapter (PCI) resource into virtual PCI functions that can be attached to Kubernetes pods. To attach an SR-IOV device resource to the Kubernetes pod network, the SR-IOV CNI can be used. The SR-IOV CNI plugin enables the Kubernetes pod to be attached directly to an SR-IOV virtual function (VF) using the standard SR-IOV VF driver in the container host's kernel. The SR-IOV CNI can be omitted if the VF is attached to a containerized application through a userspace VF driver.

For more details, see: SR-IOV CNI,

4.4.4 Userspace CNI

Userspace CNI provides a high-performance container networking solution and data plane acceleration for containers. It is designed to implement userspace networking (as opposed to kernel space networking), such as DPDK-based applications. It is designed to run with either OVS-DPDK or VPP along with the Multus CNI plugin in Kubernetes deployments.

4.4.5 Bond CNI

Bond CNI allows for aggregation of multiple network interfaces into one logical interface. Interface bonding is used to provide additional network capacity and/or redundancy. When used as stand-alone plugin, interfaces are obtained from the host's network namespace. A bonded interface is created in the container network namespace. When used with Multus, interfaces that were previously passed to the container can be bonded.

[7] "The Istio Service Mesh." Istio. Accessed October 8, 2021. https://istio.io/latest/about/service-mesh/.

4.4.6 Intel® QuickAssist Device Plugin

Intel® QuickAssist Adapters integrate hardware acceleration for compute-intensive workloads, such as bulk cryptography, public key exchange, and compression, on Intel® architecture platforms. The Intel QAT device plugin for Kubernetes supports Intel QuickAssist Adapters.

For more details, see: Intel Device Plugins for Kubernetes

4.4.7 Intel® Software Guard Extensions (Intel® SGX) Device Plugin

The Intel SGX device plugin allows Kubernetes workloads to use Intel SGX on 3rd and 4th Gen Intel® Xeon® Scalable processors. The plugin is used with an SGX admission webhook and EPC memory registration to isolate specific application code and data in memory via enclaves that are designed to be protected from processes running at higher privilege levels. An additional remote attestation server is required to verify that software is running inside an SGX enclave on a trusted computing node.

For more details, see: Intel Device Plugins for Kubernetes.

4.4.8 Node Feature Discovery

In a standard deployment, Kubernetes reveals very few details about the underlying platform to the user. This may be a good strategy for general data center use, but, in many cases workload behavior or performance may improve by leveraging the platform features (hardware and/or software). Node Feature Discovery (NFD) is a Kubernetes add-on that detects and advertises platform hardware and software capabilities that can be used to facilitate intelligent scheduling of a workload. NFD detects the following features:

- **CPUID**: NFD advertises CPU features such as Intel® Advanced Vector Extensions (Intel® AVX). Certain workloads, such as machine learning, may gain a significant performance improvement from these extensions.
- **SR-IOV networking**: NFD detects the presence of SR-IOV-enabled NICs, allowing optimized scheduling of network-intensive workloads.
- **Intel® Resource Director Technology (Intel® RDT)**: Intel RDT allows visibility and control over the use of last-level cache (LLC) and memory bandwidth between co-running workloads. By allowing allocation and isolation of these shared resources, and thus reducing contention, Intel RDT helps to mitigate the effects of noisy neighbors. NFD detects the different Intel RDT technologies supported by the underlying hardware platform.
- **Intel® Turbo Boost Technology**: NFD detects the state of Intel Turbo Boost Technology, allowing optimal scheduling of workloads that have a well-understood dependency on this technology.
- **IOMMU**: An input/output memory management unit (IOMMU), such as Intel® Virtualization Technology (Intel® VT) for Directed I/O (Intel® VT-d), allows isolation and restriction of device accesses. This enables direct hardware access in virtualized environments, highly accelerating I/O performance by removing the need for device emulation and bounce buffers.
- **SSD storage**: NFD detects the presence of non-rotational block storage on the node, making it possible to accelerate workloads requiring fast local disk access.
- **NUMA topology**: NFD detects the presence of NUMA topology, making it possible to optimize scheduling of applications based on their NUMA awareness.
- **Linux kernel**: NFD detects the kernel version and advertises it through multiple labels, allowing the deployment of workloads with different granularity of kernel version dependency.
- **PCI**: NFD detects PCI devices, allowing optimized scheduling of workloads dependent on certain PCI devices.

For more information, see Node Feature Discovery Feature sources and Node Feature Discovery.

4.4.9 Topology Manager

Today's systems use a combination of CPUs and hardware accelerators to support latency-critical execution and high-throughput parallel computation. To help extract the optimal performance from such systems, optimizations related to CPU isolation, memory, and device locality must be made. In Kubernetes however, these optimizations are handled by a disjointed set of components. Topology Manager is a feature of Kubernetes distribution. It is a kubelet component that coordinates the set of components that are responsible for making topology-aligned resource allocations.

For details see: Topology Management - Implementation in Kubernetes Technology Guide

4.4.10 Kubernetes Native CPU Manager

Native CPU Manager for Kubernetes provides mechanisms for allocation of CPU cores to workloads in situations where pods contend for resources of the CPU. When contention happens, workloads may be moved to other CPUs and workload performance may be impacted. To avoid this, CPU Manager offers an option to allocate exclusive cores to a workload (pod) by specifying "guaranteed QoS" and integer CPU requests.

Note: The CPU Manager for Kubernetes was deprecated in Kubernetes 1.22. Use the Kubernetes Native CPU Manager.

See: Feature Highlight: CPU Manager

4.4.11 Platform Aware Scheduling

Platform Aware Scheduling (PAS) contains a group of related projects designed to expose platform-specific attributes to the Kubernetes scheduler using a modular policy-driven approach. PAS contains a core library and information for building custom scheduler extensions as well as specific implementations that can be used in a working cluster or leveraged as a reference for creating new Kubernetes scheduler extensions.

Telemetry Aware Scheduling (TAS) is the initial reference implementation of PAS. It can expose any platform-level metric to the Kubernetes Scheduler for policy-driven filtering and prioritization of workloads. You can read more about TAS here: Telemetry Aware Scheduling.

GPU Aware Scheduling is the implementation of the GPU resource-aware Kubernetes scheduler extension.

Telemetry Aware Scheduling is made up of two components deployed in a single pod on a Kubernetes cluster.

- **Telemetry Aware Scheduler Extender** is contacted by the generic Kubernetes scheduler every time it needs to make a scheduling decision on a pod that calls for telemetry scheduling. The extender checks if there is a telemetry policy associated with the workload. If so, it inspects the strategies associated with the policy and returns opinions on pod placement to the generic scheduler. The scheduler extender has two strategies that it acts on – `scheduleonmetric` and `dontschedule` – and is implemented and configured as a Kubernetes Scheduler Extender. See: Extending Kubernetes.
- **Telemetry Policy Controller** consumes TAS policies. The Telemetry Policy Controller parses TAS policies and places them in a cache to make them locally available to all TAS components. The controller consumes new telemetry policies as they are created, removes them when deleted, and updates them as they are changed. The policy controller also monitors the current state of policies to see if they are violated

TAS acts on two strategy types.

- **`scheduleonmetric`** has only one rule. It is consumed by the Telemetry Aware Scheduler Extender and prioritizes nodes based on a comparator and an up-to-date metric value. For example:
  ```
  scheduleonmetric when health_metric is LessThan
  ```
- **`deschedule`** is consumed by the Telemetry Policy Controller and can have multiple rules. If a pod is running on a node that violates this policy, it can be descheduled with the Kubernetes descheduler. For example:
  ```
  deschedule if health_metric Equals 2
  ```

TAS allows arbitrary, user-defined rules to be put in place to impact scheduling in a Kubernetes cluster. Using the Kubernetes descheduler, it can evict a workload that is breaking rules to have the workload placed on a more suitable node.

In a modern cloud computing cluster, there is a torrent of data that only certain subject matter experts know how to interpret and act upon. In scheduling workloads, operators know on which compute nodes a workload may perform better based on up-to-date utilization metrics. Likewise, certain telemetry values, or combinations of values, can be recognized as signs that a node has a serious problem that is interfering with workload operation. The TAS policy system allows these insights to influence the scheduling and lifecycle placement process, turning implicit personal knowledge into formal, actionable information.

For details, refer to: Telemetry Aware Scheduling (TAS) - Automated Workload Optimization with Kubernetes (K8s) Technology Guide

4.5 Istio Service Mesh

Istio is an open-source service mesh that layers transparently onto existing distributed applications.

> “A service mesh is a dedicated infrastructure layer that you can add to your applications. It allows you to transparently add capabilities like observability, traffic management, and security, without adding them to your own code. The term ‘service mesh’ describes both the type of software you use to implement this pattern, and the security or network domain that is created when you use that software.” [8]

Istio’s features provide a uniform and efficient way to help secure, connect, and monitor services. Istio is the path to load balancing, service-to-service authentication, and monitoring – with few or no service code changes. Its powerful control plane enables vital features, including:

- More secure service-to-service communication in a cluster with TLS encryption, strong identity-based authentication, and authorization
- Automatic load balancing for HTTP, gRPC, WebSocket, and TCP traffic
- Fine-grained control of traffic behavior with rich routing rules, retries, failovers, and fault injection
- A pluggable policy layer and configuration API supporting access controls, rate limits, and quotas
- Automatic metrics, logs, and traces for all traffic within a cluster, including cluster ingress and egress

[8] “The Istio Service Mesh.” Istio. Accessed October 8, 2021. https://istio.io/latest/about/service-mesh/.

For more details, refer to: The Istio service mesh.

The Reference System Architectures are shipped with a predefined set of Istio configuration profiles. The following Istio configuration profiles are available: default, minimal, external, empty, none. If no Istio configuration profile is defined by the variable `service_mesh.profile` in the `group_vars/all.yml` file, then the default Istio configuration profile is applied. If no Istio configuration profile is set, then no specific Istio configuration profile is deployed.

It is also possible to deploy a custom Istio profile by placing a profile configuration yaml file to the service mesh role directory: `roles/service_mesh_install/files/profiles/`. The name of the file must correspond to the profile name.

For more details on Istio configuration profiles, see Istio Installation Configuration Profiles.

4.5.1 Bypass TCP/IP Stack Using eBPF

This option provides acceleration of the Istio TCP/IP traffic in the following scenarios:

- Service to service communication when the services are in the same pod
- Service to service communication when the services are in the same node

Acceleration is accomplished using an eBPF program loaded on every node participating in the service mesh communication. To enable this feature, set `group_vars/all.yml` variable `service_mesh.tcpip_bypass_ebpf` to `true`.

For more details, visit Istio TCP/IP Bypass.

4.5.2 TLS Splicing

In TLS splicing, after a TCP connection is established between the client and Envoy, any subsequent TLS traffic is forwarded to an upstream server as raw TCP data. This is required for the client to access some external services where connection data should not be decoded by Envoy.

4.5.3 Istio Acceleration with Intel AVX-512 Crypto (Examples Available Under NDA)

Intel acceleration features enabled:

- AVX-512 crypto acceleration for TLS connections
- AVX-512 vector AES for symmetric data encryption

4.5.4 Istio Acceleration and Compression with QAT 2.0 (Examples Available Under NDA)

Intel acceleration features enabled:

- QAT 2.0 crypto acceleration for TLS handshakes
- QAT 2.0 compression acceleration for HTTPs connections

4.5.5 Istio mTLS Private Key Protection with SGX

The security of the private keys of mTLS communication between Service Mesh workloads by the Intel® SGX. The private keys are stored and used inside the SGX enclaves and are never stored anywhere else in the system. Istio proxies can use the private key in the enclave by the key handle provided by SGX to communicate with other counterparts.

4.5.6 Istio Multi-tenancy

The Kubernetes cert-manager add-on automates the management and issuance of TLS certificates from various issuing sources. A user can use cert-manager to approve and sign Certificate Signing Request (CSR) for workloads that are deployed in service mesh. This means that different workloads owned by a particular tenant can hold different Certificate Authorities (CAs).

4.6 Kubernetes Operators

Kubernetes operators are subject-specific controllers that extend the functionality of the Kubernetes API to create, configure, and manage complex entities on behalf of a Kubernetes user. Kubernetes is designed from the ground up for automation as it includes native mechanisms for deploying, running, and scaling workloads. However, some workloads and services require deeper knowledge of how the system should behave outside the bounds of core Kubernetes. Operators are purpose-built with operational intelligence to address the individuality of such constructs. Tools for operator creation help developers build an automation experience for cluster administrators and end users. By extending a common set of Kubernetes APIs and tools, Kubernetes operators can help provide ease of deployment and streamlined Day 1 and Day 2 operations. The Kubernetes Operator Pattern has emerged as a solution to help ensure that the automation of these function-specific applications is possible in a Kubernetes cluster.

4.6.1 SR-IOV Network Operator

The SR-IOV Network Operator is designed to help the user provision and configure the SR-IOV CNI plugin and device plugin in the Kubernetes cluster.

When the SR-IOV Network Operator is enabled by setting `sriov_network_operator_enabled: true` in the `group_vars/all.yml` file, it is mutually exclusive with SR-IOV CNI and device plugins. The plugins must be disabled by setting the respective options in `group_vars/all.yml` and `host_vars/<hostname>.yml`. The SR-IOV Network Operator is enabled by default, and SR-IOV CNI and device plugins are disabled by default.

To configure SR-IOV device custom resource definition (CRD), provide the `SriovNetworkNodePolicy`. The Reference System Architecture handles creation of such CRDs and creates and populates them automatically according to the provided configuration in the `host_vars/node1.yml` file, namely in the section `dataplane_interfaces`.

For more details, refer to: SR-IOV Network Operator.

4.6.2 Intel Device Plugins Operator

Intel device plugins operator is a Kubernetes custom controller. Its goal is to serve the installation and lifecycle management of Intel device plugins for Kubernetes. It provides cluster administrators with a single point of control for GPU, QAT, SGX, DSA, and FPGA devices. Device plugins are deployed by applying custom resources, and each device plugin has its own custom resource definition (CRD). The corresponding controller watches CRUD operations to those custom resources. Currently, the Reference System Architecture uses the Intel device plugins operator to deploy the Intel SGX device plugin and Intel GPU device plugin.

For more details, refer to: Intel Device Plugins for Kubernetes.

4.6.3 MinIO Operator

Figure 7 shows the software architectural view for the storage solution. MinIO provides the central process for servicing storage. However, to meet redundancy and scalability requirements, a storage node never operates in isolation. Rather, it is deployed in concert with a number of peers. Figure 13 represents this figuratively.

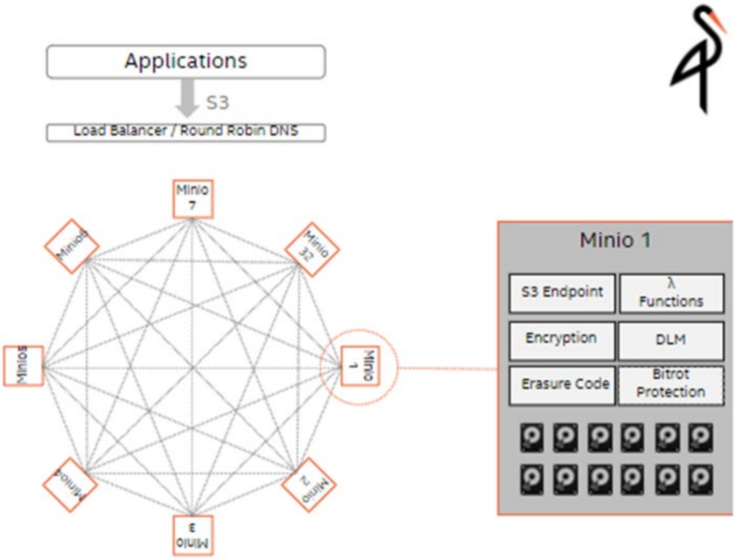


Figure 13. Storage Node Deployment Example

In the Reference System Architecture, four MinIO instances are selected for the cluster – the minimum requirement to ensure data integrity. However, this does not limit the user to expand beyond this number to address specific deployment and capacity needs.

MinIO itself is configured into a mesh – each knowing and operating in concert with its peers. It is the MinIO operator that in a Kubernetes environment orchestrates the rollout.

For more information, see MinIO Operator.

4.6.4 Intel® Ethernet Operator

The role of the Intel® Ethernet Operator is to orchestrate and manage the configuration of the capabilities exposed by the Intel E810 Series NICs. The operator is a state machine that configures certain functions, monitors the status, and acts autonomously based on user interaction. The Intel Ethernet Operator provides functionality for:

- Update of the devices' FW via NVM Update tool
- Update of the devices' DDP profile
- Flow configuration of traffic handling for the devices, based on supported DDP profile

Figure 14 shows an overview of the architecture of the Intel Ethernet Operator, which includes the following:

- The operator provides APIs, controllers, and daemons for the management and execution of supported features.

- You interact with the operator by providing custom resources (CR) to Kubernetes that are constantly monitored to detect changes and act based on what is detected.
- Separate CRs are provided for the FW/DDP update functionality and the flow configuration functionality.
- After the CR is applied or updated, the operator checks if the configuration is already applied, and if it is not, it applies the configuration.

For more information, see Intel Ethernet Operator.

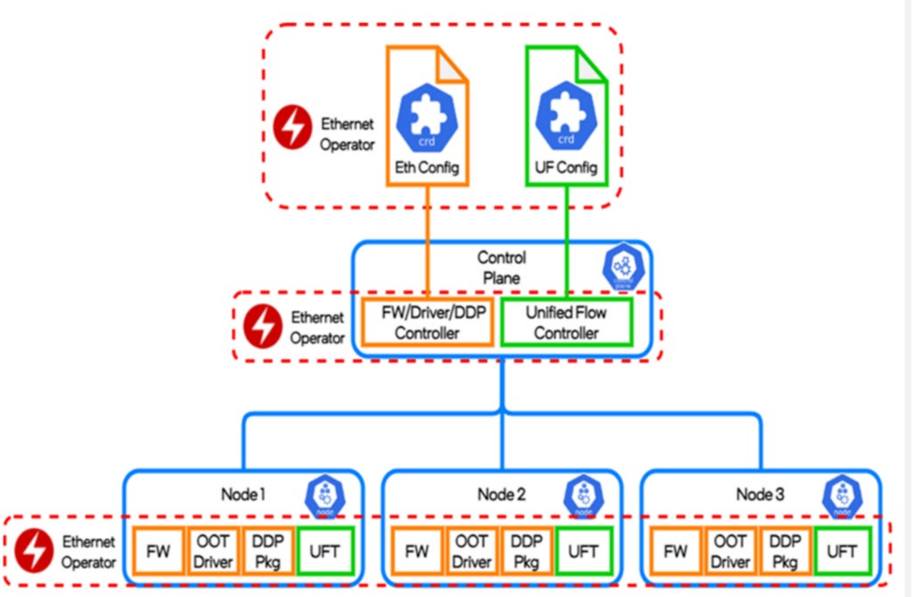


Figure 14. Intel Ethernet Operator Architecture Overview

4.6.5 Forward Error Correction (FEC) Operator

The role of the FEC Operator for Intel Wireless FEC Accelerator is to orchestrate and manage the resources and devices exposed by a range of Intel's vRAN FEC acceleration devices and hardware within the Kubernetes cluster. The operator is a state machine that configures the resources and then monitors them; it acts autonomously based on user interaction.

This operator handles the management of the FEC devices used to accelerate the FEC process in vRAN L1 applications - the FEC devices are provided by a designated hardware (for example, FPGA or eASIC PCI extension cards). It provides functionality to create desired VFs (virtual functions) for the FEC device, binds them to appropriate drivers, and configures the VF's queues for desired functionality in 4G or 5G deployments. It also deploys an instance of the SR-IOV network device plugin that manages the FEC VFs as a Kubernetes cluster resource and configures this device plugin to detect the resources. The user interacts with the operator by providing a CR (Custom Resource). The operator constantly monitors the state of the CR to detect any changes and acts based on the changes detected. The CR is provided per cluster configuration. The components for individual nodes can be configured by specifying appropriate values for each component per "nodeName". After the CR is applied or updated, the operator/daemon checks if the configuration is applied already, and, if not it binds the PFs to the driver, creates the desired number of VFs, binds them to the driver, and runs the `pf-bb-config` utility to configure the VF queues to the desired configuration.

Additional information about the FEC Operator can be found at SEO Operator for Wireless FEC Accelerators

4.7 Dynamic Device Personalization (DDP)

One of the key technologies of the Intel® Ethernet 700 and 800 Series Network Adapters is Dynamic Device Personalization (DDP). This technology enables workload-specific optimizations using the programmable packet-processing pipeline. Protocols also help improve packet processing efficiency, which can result in enhanced performance in specific use cases.[9]

Additional information about DDP can be found at Pipeline Programmability with Dynamic Device Personalization (DDP) Feature Brief

4.7.1 DDP on Intel Ethernet 700 Series Network Adapters

Intel Ethernet 700 Series Network Adapters support only one DDP image loaded on a network card. An image must be loaded to the first physical function of the device (PF0), but the configuration is applied to all ports of the adapter.

For Intel Ethernet 700 Series Network Adapters, the Reference System Architectures provide the following DDP images:

[9] Workloads and configurations. Results may vary.

- `ecpri.pkg (eCPRI)`
- `esp-ah.pkg (ESP, AH)`
- `ppp-oe-ol2tpv2.pkg (PPPoE)`
- `mplsogreudp.pkg (MPLSoGRE/MPLSoUDP)`
- `gtp.pkgo (GTPv1)`

For more information, see the following:

- Dynamic Device Personalization for Intel® Ethernet 700 Series
- Intel® Ethernet Controller 700 Series GTPv1 - Dynamic Device Personalization Application Note

4.7.2 DDP on Intel® Ethernet 800 Series Network Adapters

Intel® Ethernet 800 Series Network Adapters support a broad range of protocols in DDP profile images. You can choose the default image or the telecommunications (comms) image that supports certain market-specific protocols in addition to protocols in the OS-default package. The OS-default DDP package supports the profiles listed in Table 9.

Table 9. DDP Default Package Profiles

DEFAULT PACKAGE
MAC
EtherType
VLAN
IPv4
IPv6
TCP
ARP
UDP
SCTP
ICMP
ICMPV6
CTRL
LLDP
VXLAN-GPE
VxLAN (non-GPE)
Geneve
GRE
NVGRE
RoCEv2
COMMS PACKAGE
Extends default profile package with the following protocols:
GTP
PPPOE
L2TPv3
IPsec
PFCP

4.7.2.1 Kubernetes Support for DDP in Intel Ethernet 800-Series Network Adapters

SR-IOV devices (VFs included) are exposed to the Kubernetes pods via the SR-IOV device plugin (SR-IOV DP). In the Reference System Architectures, the plugin cannot discover loaded DDP plugins on Intel Ethernet 800-Series Network Adapters (SR-IOV Network Device Plugin Configurations).

As a result, a workload that requires a DDP profile cannot be orchestrated on such network adapters. A workaround can be accomplished in the following ways:

1. Deploy a `comms` profile on every node with Intel Ethernet 800 Series Network Adapter, or

2. Deploy a `comms` profile on a subset of Intel Ethernet 800 Series Network Adapters and label the nodes accordingly using Kubernetes labels. Then deploy the workload that requires a specific DDP profile with specific Node Selector.

4.8 Kubernetes Power Manager

The Kubernetes Power Manager is a Kubernetes operator designed to expose and utilize Intel-specific power management technologies in a Kubernetes environment. The Kubernetes Power Manager allows control of the following Intel SST features on the Kubernetes level:

- Intel SST-BF - (Intel Speed Select Technology - Base Frequency)
- Intel SST-CP - (Intel Speed Select Technology - Core Power)
- Frequency Tuning

For more details, refer to Kubernetes Power Manager.

4.9 Intel® Speed Select Technology

Intel® Speed Select Technology (Intel® SST) is a collection of features that give more granular control over CPU performance.

4.9.1 Intel Speed Select Technology – Base Frequency

Intel® Speed Select Technology - Base Frequency (Intel® SST-BF) is offered in select SKUs of 2nd Gen Intel Xeon Scalable processors (5218N, 6230N, and 6252N), 3rd Gen Intel Xeon Scalable processors (6318N and 6338N), and 4th Gen Intel Xeon Scalable processors. Intel SST-BF controls the core frequency model (Figure 15). When enabled, some cores run at higher frequency and the remaining cores run at lower frequency.

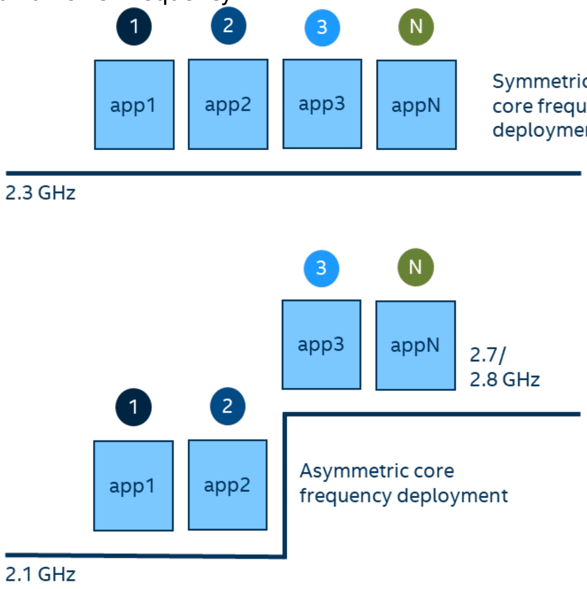


Figure 15. CPU Core Frequency Deployment Methods

Applications that consist of virtualized switches or load distribution functions in front of worker threads can benefit from the Intel SST-BF feature. With high frequencies, load distribution threads can pass data extremely fast to workers.[10]

Intel SST-BF enabled on a node is advertised by the Node Feature Discovery with the `sst_bf.enabled` label.

For more information about scheduling workloads on Kubernetes with Intel SST-BF support, visit Intel® Speed Select Technology - Base Frequency (Intel® SST-BF) with Kubernetes Application Note

On the 3rd Gen Intel Xeon Scalable processor, it is possible to prioritize power flow to the high priority cores in the event there is a power constraint scenario, for example, TDP ceiling hit or power supply malfunction. Power prioritization helps high frequency cores maintain their frequency.

In BMRA deployments, the core priority of high frequency cores is enabled by default.

For more information about Intel SST-BF technology, refer to Intel® Speed Select Technology - Base Frequency (Intel® SST-BF)

[10] Workloads and configurations. Results may vary.

4.9.2 Intel® Speed Select Technology – Core Power

Intel® Speed Select Technology – Core Power (Intel® SST-CP), available on 3rd and 4th Gen Intel Xeon Scalable processors, allows users to define priorities in core power flow in the event there is a power constraint scenario, for example TDP ceiling hit or power supply malfunction. Power prioritization helps cores maintain their frequency, while power is drawn from cores with lower priority.

Reference System Architectures allow users to define four Intel SST-CP Classes of Service that contain:

1. Priority
2. Affected Cores
3. Minimum CPU Frequency
4. Maximum CPU Frequency

Intel SST-CP configuration can be found in `host_vars`. Intel SST-CP enabled on a node is advertised by the Node Feature Discovery with the `sst_cp.enabled` label.

For more information about Intel® SST-CP technology, refer to: Intel® Speed Select Technology - Core Power (Intel® SST-CP)

4.9.3 Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF)

Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF), available on 3rd and 4th Gen Intel Xeon Scalable processors, allows users to assign specific cores to achieve a higher turbo frequency. This feature enables the ability to set different "All core turbo ratio limits" to cores based on priority. By using this feature, some cores can be configured for a higher turbo frequency by designating them as high priority at the cost of lower or no turbo frequency on the low priority cores. For this reason, this feature is only useful when the system is busy using all CPUs, but the user wants a configurable option to get high performance on some CPUs.

The support of Intel SST-TF depends on Intel SST-PP performance level configuration. It is possible that only a certain performance level supports Intel SST-TF. It is also possible that only the base performance level (level = 0) has the support of Intel SST-TF. Hence, first select the desired performance level to enable this feature.

For more information about Intel SST-TF technology, refer Intel® Speed Select Technology - Turbo Frequency (Intel® SST-TF) and Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF) Overview User Guide.

4.9.4 Intel® Speed Select Technology – Performance Profile (Intel® SST-PP)

Intel® Speed Select Technology – Performance Profile (Intel® SST-PP), available on 3rd and 4th Gen Intel Xeon Scalable processors, permits dynamic configuration of a server based on workload performance necessities. The working nature of Intel SST-PP is to set up CPUs that need to be online and others that need to remain offline to sustain a guaranteed frequency performance level. Intel SST-PP provides the ability to monitor changes in frequency levels dynamically.

Reference System Architectures provide two options for Intel SST-PP deployment:

- Configure Intel SST-PP with all features (Intel SST-BF, Intel SST-CP, and Intel SST-TF) in auto mode, which enables turbo frequency for all available online CPUs automatically.
- Configure Intel SST-PP with all features (Intel SST-BF, Intel SST-CP, and Intel SST-TF) with user-defined specific CPU ranges such as "2,3,5" to prioritize those CPUs over others.

Intel SST-PP configuration can be found in `host_vars`.

For more information about Intel SST-PP technology, refer to Intel® Speed Select Technology - Performance Profile (Intel® SST-PP)

4.10 Security

The following sections describe some of the security features in the Network and Cloud Edge Reference System Architectures Portfolio.

4.10.1 Native Kubernetes Cluster Security Features

Security features for the native Kubernetes cluster include the following:

- Kubernetes audit trail and log backups by default
- Certs for API server access and timeouts
- Checksums for all downloaded artifacts
- RBAC enabled by default
- TLS cipher suites to help secure cluster communication
- Help secure the container registry by:
  - Limited access with TLS and basic authentication
  - Required server keys and certs signed with Kubernetes Certificate Authority (CA)
- Limited etcd permissions

- Restricted open firewall ports and subnets
- Pod Security Policies (PSP) admission controller enabled with minimal set of rules (defines a set of conditions pods must use to run within the system):
  - EventRateLimit – mitigates DoS attacks against API server
  - AlwaysPullImages – forces credential checks every time a pod image is accessed
  - NodeRestriction – limits objects that a kubelet can modify
  - PodSecurityPolicy
- Security policies on each Helm chart for deployment (collectd, NFD, TAS, and the like)

For additional security considerations, refer to:

- Securing a Cluster
- Pod Security Policies
- NodeRestriction

4.10.2 Intel® Security Libraries for Data Center (Intel® SecL – DC)

Security for cloud-native applications is an increasing challenge for developers. Hardware and software suppliers in the industry are cooperatively developing the cloud architecture to deliver 5G network and edge infrastructure, which makes security an important requirement as workloads and suppliers converge.

By using Intel SecL – DC, Hardware Root of Trust, and Secure Boot, you can create an end-to-end platform security solution for network and edge platforms. Key components of Intel® SecL – DC include the following:

- **Verification service**: Installed in the central control node, it gathers the secure boot signatures and maintains the platform's "trusted" or "untrusted" evaluation results. It maintains the platform trust database with established "known good" values or expected measurements. If a certain firmware or Linux kernel is found to be compromised, the policy can change the platform trust status.
- **Trust agent**: Installed in every physical node that needs to be monitored by the platform security, it takes the TPM ownership on the platform and reports the platform security status to the remote management module.
- **Integration hub**: Connects with orchestration software, such as Kubernetes, and contains an extension to work with Kubernetes. It retrieves the information from the verification service and shares with the orchestration software at a specified interval.

For details on how to deploy, refer to: Secure the Network Infrastructure - Secure Cloud Native Network Platforms User Guide

4.10.3 Intel® Software Guard Extensions

Intel® Software Guard Extensions (Intel® SGX) is an Intel technology to protect select code and data from disclosure or modification. It operates by allocating hardware-protected memory where code and data reside. The protected memory area is called an *enclave*. Data within the enclave memory can only be invoked via special instructions. This feature is only available on 3rd and 4th Gen Intel Xeon Scalable processors.

4.10.4 Key Management Reference Application with Intel® SGX

Key Management Reference Application (KMRA) is a proof-of-concept software created to demonstrate the integration of Intel SGX asymmetric key capability with a hardware security model (HSM) on a centralized key server (Figure 16). The Network and Cloud Edge Reference System Architectures Portfolio highlights KMRA support on 3rd and 4th Gen Intel Xeon Scalable processors only.

The following description outlines the setup of KMRA infrastructure with Intel SGX for private key provisioning to an Intel SGX enclave on a 3rd or 4th Gen Intel Xeon Scalable processor. This method uses the Public-Key Cryptography Standard (PKCS) #11 interface and OpenSSL.

The BMRA Ansible scripts automatically set up the KMRA infrastructure. The process includes the setup of a centralized key server and multiple compute nodes enabled with Intel SGX. The compute nodes are provisioned with private keys into Intel SGX enclaves used by workloads. This general solution can work with any application or workload.

In the Network and Cloud Edge Reference System Architectures Portfolio, a sample NGINX workload/application Ansible script is provided as an example of how the KMRA infrastructure can be used. The NGINX workload uses the private keys from the Intel SGX enclave to establish TLS connections. The private key is never in the clear; thus, certificate signing happens more securely inside the enclave.

Step-by-step instructions for deploying KMRA with BMRA are detailed in the Network and Cloud Edge Container Bare Metal Reference System Architecture User Guide. The instructions also provide information on how to deploy the NGINX workload and configure it to use the private key more securely from inside the enclave to establish TLS connections.

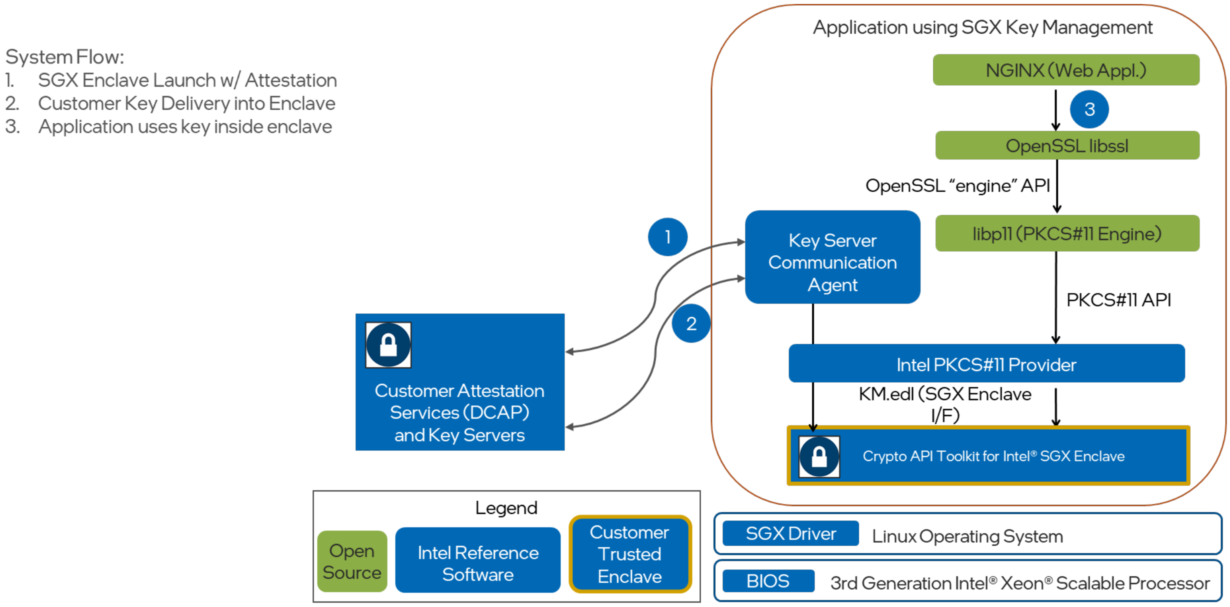

System Flow:
1. SGX Enclave Launch w/ Attestation
2. Customer Key Delivery into Enclave
3. Application uses key inside enclave
Application using SGX Key Management
NGINX (Web Appl.)
3
OpenSSL libssl
OpenSSL "engine" API
1
Key Server Communication Agent
libp11 (PKCS#11 Engine)
PKCS#11 API
2
Customer Attestation Services (DCAP) and Key Servers
Intel PKCS#11 Provider
KM.edl (SGX Enclave I/F)
Crypto API Toolkit for Intel® SGX Enclave
Legend
Open Source
Intel Reference Software
Customer Trusted Enclave
SGX Driver
Linux Operating System
BIOS
3rd Generation Intel® Xeon® Scalable Processor

Figure 16. Key Management Reference Application Infrastructure with Intel® SGX

For more information, see Intel® Software Guard Extensions (Intel® SGX) – Key Management Reference Application (KMRA) on Intel® Xeon® Processors Technology Guide and Intel® Software Guard Extensions (Intel® SGX) - Key Management Reference Application (KMRA) on Intel® Xeon® Processors User Guide.

4.10.5 OpenSSL and QAT Engine

Network security vendors can rely on the OpenSSL project, and specifically the `libcrypto` general purpose cryptographic library, when executing in cloud compute environments. To address the need to have both portability and performance, Intel offers the Intel® QuickAssist Technology Engine for OpenSSL (Intel® QAT Engine for OpenSSL) as an additional option to the default library.

The Intel QAT Engine for OpenSSL 1.1.x provides support for secure applications by directing the requested computation of cryptographic operations to the available hardware acceleration or instruction acceleration present on the platform. The engine supports both the traditional synchronous mode for compatibility with existing applications and the asynchronous mode introduced in OpenSSL 1.1.0 to help achieve maximum performance.

4.10.6 Service Mesh

For information about Service Mesh, see Service Mesh.

4.10.7 Trusted Certificate Service

Trusted Certificate Service (TCS) is a Kubernetes certificate signing solution that uses the security capabilities provided by Intel® SGX. The signing key is stored and used inside the SGX enclaves and is never stored in clear anywhere in the system. TCS is implemented as a cert-manager external issuer by supporting both cert-manager and Kubernetes certificate signing APIs.

For detailed information and use cases, see Trusted Certificate Issuer.

4.10.8 Trusted Attestation Controller

The Trusted Attestation Controller is a Kubernetes controller for reconciling the QuoteAttestation requests initiated by the Trusted Certificate Service (TCS). It mediates between the TCS and the key servers that support attestation services. The key servers can plug into the controller by implementing the API provided by the controller.

For detailed information and use cases, see Trusted Attestation Controller.

4.11 Observability

This solution deploys a broad range of telemetry collectors. Platform collectors capture metrics from CPU performance monitoring, which includes Intel hardware features, such as RAS, power, PMU, RDT, NVMe storage, network interfaces, disks, platform power, memory, and thermal. In addition to hardware telemetry, the Reference System Architectures capture DPDK and OVS metrics.

Metrics can be published to Prometheus and other interfaces such as SNMP, Kafka, and VES.

The telemetry software stack consists of the following components:

- **collectd** – UNIX daemon that collects a wide range of platform telemetry, including RDT and PMU metrics. It has a modular architecture, and data acquisition depends on loaded plugins. Plugins that are loaded with collectd depend on an installed profile. For detailed descriptions of the plugins and metrics available, visit collectd and Barometer Home.
- **Telegraf** – Cloud native server telemetry collection agent that collects a wide array of platform and application telemetry, including RDT, PMU, and DPDK metrics. Telegraf is an open-source plugin-based server agent that you can use to collect custom application metrics, logs, network performance data, platform metrics, and more. Developed by InfluxData, Telegraf's pluggable architecture uses input plugins for collecting metrics and output plugins for sending metrics to various endpoints, such as time series databases like Prometheus and Influx DB or data streaming services like Kafka. For more information on Telegraf and its available plugins, visit Telegraf.
- **Node Exporter** – Platform telemetry monitoring agent; a Prometheus exporter for hardware and OS metrics exposed by NIX kernels, written in Go with pluggable metric collectors. Node Exporter is run with its default configuration on every node.
- **Prometheus** – Prometheus is an open-source telemetry systems monitoring server that scrapes and stores time series data.
- **Grafana** – Grafana is an open-source telemetry visualization tool that is available under the HTTPS endpoint `https://localhost:30000` on any of the cluster nodes.
- **Prometheus Adapter** - Prometheus Adapter is an implementation of the Kubernetes resource metrics API and custom metrics API. It provides Prometheus metrics to the Telemetry Aware Scheduler.
- **Telemetry Aware Scheduler** - Makes telemetry data available to scheduling and descheduling decisions in Kubernetes.
- **Intel Telemetry Insight Reports** - Telemetry reports are available under NDA at the following link: Telemetry Insight Reports. Contact your Intel representative for information.

4.11.1 Telemetry Insight Reports

Telemetry Insight Reports software, available from Intel, translates raw platform metrics into up-leveled networking and operational insights (Figure 17). The software provides insights on platform reliability, utilization, congestion, and configuration issues. These insights can be used to notify NetOps and provide key inputs for remediation actions by automated control systems as part of an observability solution in closed loop systems.

To offer meaningful insights from this information, Intel has created a portfolio of telemetry reports that provides actionable data about the current operational status of the server, whether it is overloaded/congested, misconfigured, or in an unhealthy state.

Included as part of the Telemetry Insight Reports package is the Power Assessment Kit. The Power Assessment Kit provides a detailed view of power consumption over a fleet of servers, including server-level and CPU-level insights into power consumption, as well as the carbon emissions and environmental impact of the server fleet.

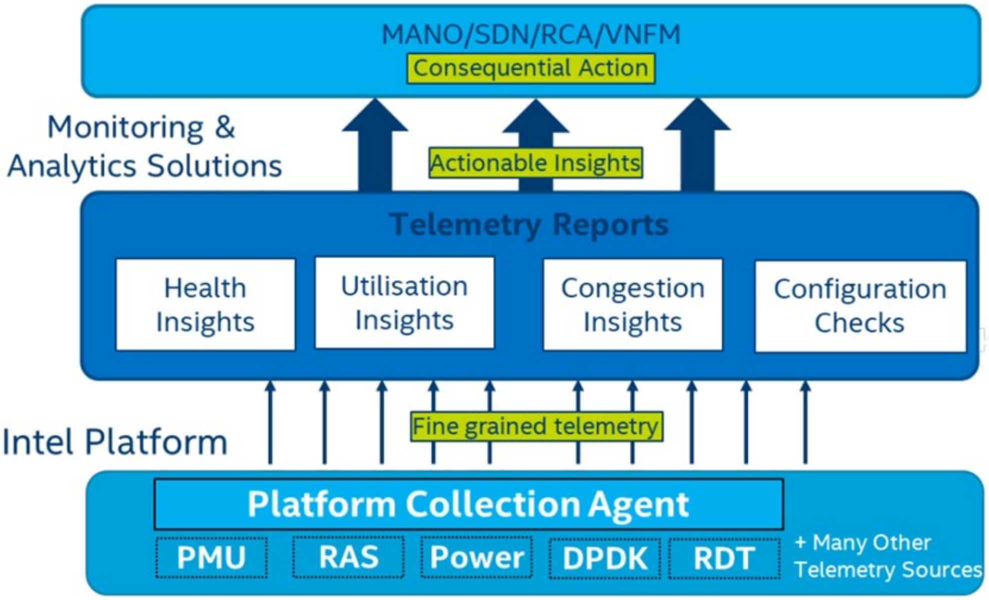


Figure 17. Telemetry Insight Reports High-Level Architecture

4.12 Object Storage

Data can be broken into three generic types of storage – block, file, object. The BMRA with Storage Flavor focuses exclusively on object storage using MinIO. MinIO offers high-performance, S3-compatible object storage and is native to Kubernetes. MinIO has no requirement for telco-specific infrastructure such as DPDK. Hence, MinIO's only architectural requirement is the basic operating system (OS) support for it to service clients and use connected storage devices, such as HDDs and SSDs.

MinIO is a high-throughput, distributed-object storage server, designed for large-scale cloud infrastructure. MinIO's primary targeted uses cases include:

- High throughput media streaming
- Machine learning and analytics
- Hadoop/Big Data storage disaggregation

MinIO delivers the following features:

- Amazon Web Services Simple Storage Service (S3) compatible
- Erasure codes and bit rot protection
- Lambda Compute – event driven notification
- Encryption and tamper proofing
- Pluggable storage backbends
- Distributed

MinIO does not use a proxy layer. All MinIO nodes run the MinIO S3 endpoint. A load balancer uses DNS round robin to distribute client connections across all nodes, servicing those client's S3 API requests.

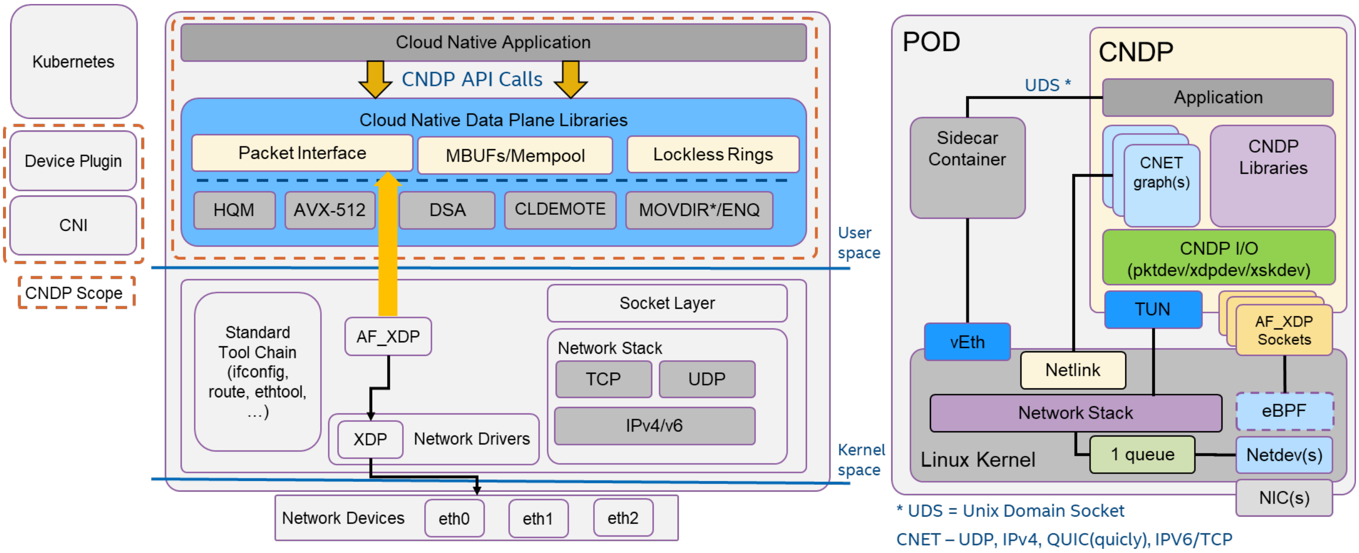
For a more detailed description of MinIO, see Multi-Cloud Object Storage.

4.13 Cloud Native Data Plane

CNDP is a purpose-built flexible data plane that provides a framework for the development of packet processing microservices closely aligned with Kubernetes. It builds on top of AF_XDP, a technology that lets data plane applications execute without knowledge of the underlying network hardware. Using an AF_XDP device plugin, container network interface (CNI), and operators, it is possible to provision, orchestrate, and manage the data plane for cloud native deployments. As CoSPs and enterprises look to deploy their applications in the cloud and at the edge, network workloads may not seamlessly migrate due to hardware incompatibilities. CNDP addresses this challenge.

CNDP makes it easy to efficiently automate and orchestrate packet processing applications on a cloud native platform. It is an open source, community driven project available at CNDP.

Figure 18 shows the CNDP architecture.



* UDS = Unix Domain Socket

CNET – UDP, IPv4, QUIC(quicly), IPV6/TCP

Figure 18. Cloud Native Data Plane Architecture

4.14 TADK

Traffic Analytics Development Kit (TADK) is a set of tools and libraries that provide a starting point for developing network applications. These applications can use acceleration instructions, targeting AI inferencing performance, available in Intel® Xeon® Scalable Processors.

Included with the Reference Architecture is a workload example of a web application firewall that uses a trained machine-learning (ML) model and the acceleration instructions to predict if an attack on the network is occurring. This example is based on the open source ModSecurity Web Application Firewall (WAF), but with changes to the detection libraries from using traditional logic to ML and AI-based prediction.

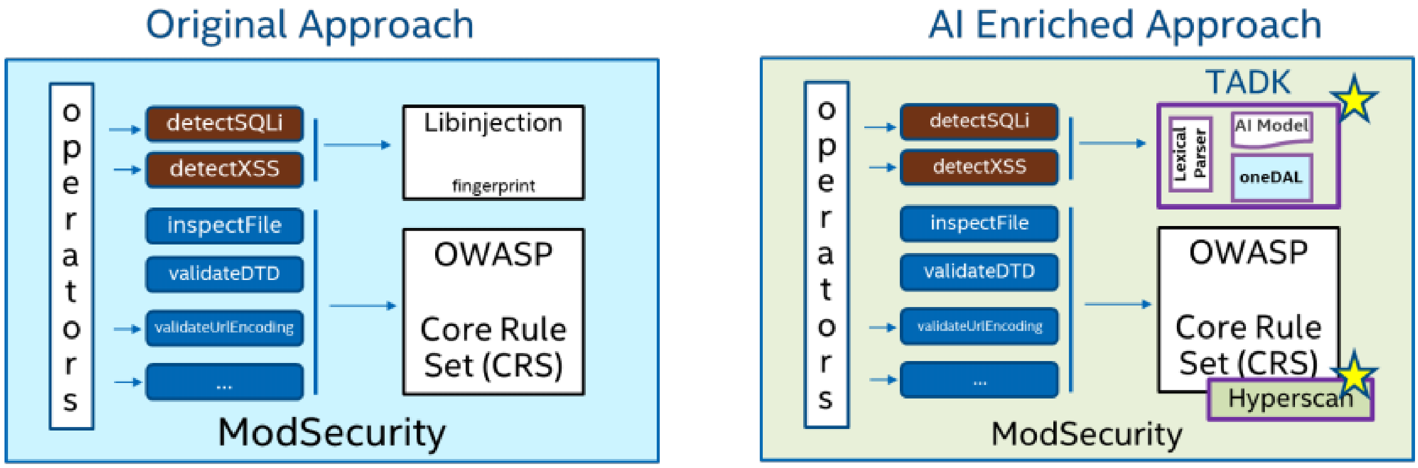


Figure 19. TADK Enhanced Implementation of ModSecurity WAF

TADK is part of a broader suite of AI Technologies that Intel has provided for existing AI frameworks and libraries. More information about these technologies can be found on Intel® Network Builders.

4.15 Access Edge - vRAN

The next generation of 5G network architectures is rapidly evolving to process high-bandwidth real-time data. This means that the core telecom network is transforming from a fixed-function, hardwired appliance architecture to a software-based, open hybrid cloud platform.

The virtualization of the radio access network (vRAN) is an essential step into the 5G future because it simplifies the deployment of new features and algorithms. vRAN also separates network functions from the underlying hardware, making it possible for a flexible and dynamic RAN environment.

4.15.1 Generic vRAN

Using Kubernetes and cloud native capabilities in an Ubuntu environment, a Distributed Unit (DU) can be configured and deployed easily using the 4th Gen Intel Xeon Scalable processor, making use of the latest hardware and software features to accelerate encoding/decoding of physical layer frames and helping to ensure low latency and power consumption.

4.15.2 FlexRAN™ Software Deployment (NDA)

Intel has created a vRAN reference implementation for virtualized cloud-enabled radio access networks that is named FlexRAN™ software.

Reference System Architectures enable FlexRAN software to be configured and deployed easily. Furthermore, power management and the FEC accelerator operators allow for high-resolution configurability and management, helping to result in high throughput and low latency while minimizing power consumption.

Deployment of the “Timer mode” test mode is enabled to provide an experience that showcases FlexRAN software’s stand-alone functionality and benchmarking of the FlexRAN software and the underlying platform combination.

5 Reference Architecture Installation

5.1 Introduction

This section provides an overview of how a particular Reference Architecture Flavor is generated and deployed. The process includes selection of a deployment model (e.g., BMRA, VMRA) and Configuration Profile, followed by installation and setup of hardware, and then system provisioning using Ansible playbooks. Intel provides a set of Ansible Playbooks to install and configure the Reference System Architectures.

Note: Ansible playbooks for 2nd, 3rd, and 4th Gen Intel Xeon Scalable processors and the Intel® Xeon® D processor are open source and available through GitHub. Request access to the NDA Ansible playbooks for the 4th Gen Intel® Xeon® Scalable processor from your Intel representative. The VMRA does not currently support deployments using the Intel Xeon D processor.

The following information describes the general installation process. For a complete guide on deployment, including the required hardware, BIOS configurations, network topology requirements, and more, refer to the appropriate user guide (BMRA or VMRA).

5.2 Installation Overview

Configuration of Reference System Architecture deployments is based on the following:

- Deployment Model: Bare metal cluster (BMRA) or virtual cluster (VMRA).
- The Configuration Profile (a specific network location or generic): On-Premises Edge, Access Edge, Remote Central Office-Forwarding, Regional Data Center, or a generic Basic, Full, Build-Your-Own, or Storage configuration. The choice impacts the hardware required and the Ansible scripts that provision the hardware and software specified.

The Reference System Architectures were designed with Kubernetes in mind for flexibility and simplicity. The installations were validated with Kubernetes on both bare metal and virtualized clusters.

These choices help drive configuration and setup for the following prerequisites needed for the Ansible host to run the scripts and for other nodes. Details for the prerequisites are listed in the appropriate user guide:

- **Hardware BOM** selection and setup
- **Required BIOS/UEFI configuration**, including virtualization and hyper-threading settings
- **Network topology requirements**, listing 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 Ansible host and Kubernetes cluster target servers (if installed)

After you complete these prerequisites, you can download Ansible playbooks for 2nd, 3rd, and 4th Gen Intel Xeon Scalable processors and Intel Xeon D processor directly from the dedicated GitHub releases page or cloned using Git. For Ansible playbooks for NDA 4th Gen Intel Xeon Scalable processor deployments, request access from your Intel Customer Support.

5.2.1 Operating System Selection

The Reference System Architectures Portfolio was validated for the following operating systems for Kubernetes control and worker nodes, VM host, and VMs and their containers:

- [BMRA only] RHEL for x86_64 Version 8 (8.5)
- Rocky Linux 8.5
- Ubuntu 22.04 RT
- Ubuntu 22.04
- Ubuntu 20.04.4 LTS

5.2.2 Network Interface Requirements

The following network requirements must be met before deployment:

- Internet network
  - Accessible by the Ansible host
  - Capable of downloading packages from the internet
  - Can be configured for Dynamic Host Configuration Protocol (DHCP) or with static IP address
- Management network (for all installations)
- Calico pod network interface (for Kubernetes; this can be a shared interface with the internet network)
  - Kubernetes control and worker node inter-node communications
  - Calico pod network runs over this network
  - Configured to use a private static address
- Tenant data networks
  - Dedicated networks for traffic
  - SR-IOV enabled
  - VF can be DPDK bound in pod

5.2.3 Software Prerequisites for Ansible Host, Control Nodes, and Worker Nodes

Before deployment of the Ansible playbooks, the Ansible host must be prepared by logging into it using SSH or your preferred method to access the shell. Install packages on the Ansible host according to the appropriate user guide.

5.3 Ansible Playbook Review

The Reference System Architecture is configured and provisioned automatically using Ansible scripts. Each Configuration Profile has a dedicated Ansible playbook. This section describes how the Ansible playbooks allow for an automated deployment of a fully functional Reference System Architecture, including initial system configuration, Kubernetes or VMs deployment, and set up of capabilities.

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 to allow use case-oriented cluster deployments (Reference Architecture Flavors). Additionally, a special Cluster Removal playbook exists to simplify removal of a deployment. This playbook is useful for restarting an installation from the beginning or removing one completely.

Ansible Playbooks use configuration files that define cluster-wide and host-specific configuration options for each of the Reference Architecture Flavors. With minimal changes they can be used directly with their corresponding Ansible playbooks. Default values for the configuration files of each Reference Architecture Flavor are shown in the appropriate user guide.

Each playbook executes multiple phases – Infrastructure Setup, Kubernetes, VM Deployment (VMRA only), and Capabilities Setup – and assigns Ansible Roles during each phase. For a VMRA deployment, there are individual phases for the Host and the VMs. Figure 20 and Figure 21 illustrate the phases and roles assigned in an example Full Configuration Profile implemented for BMRA and VMRA. The phases are described in more detail below.

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.

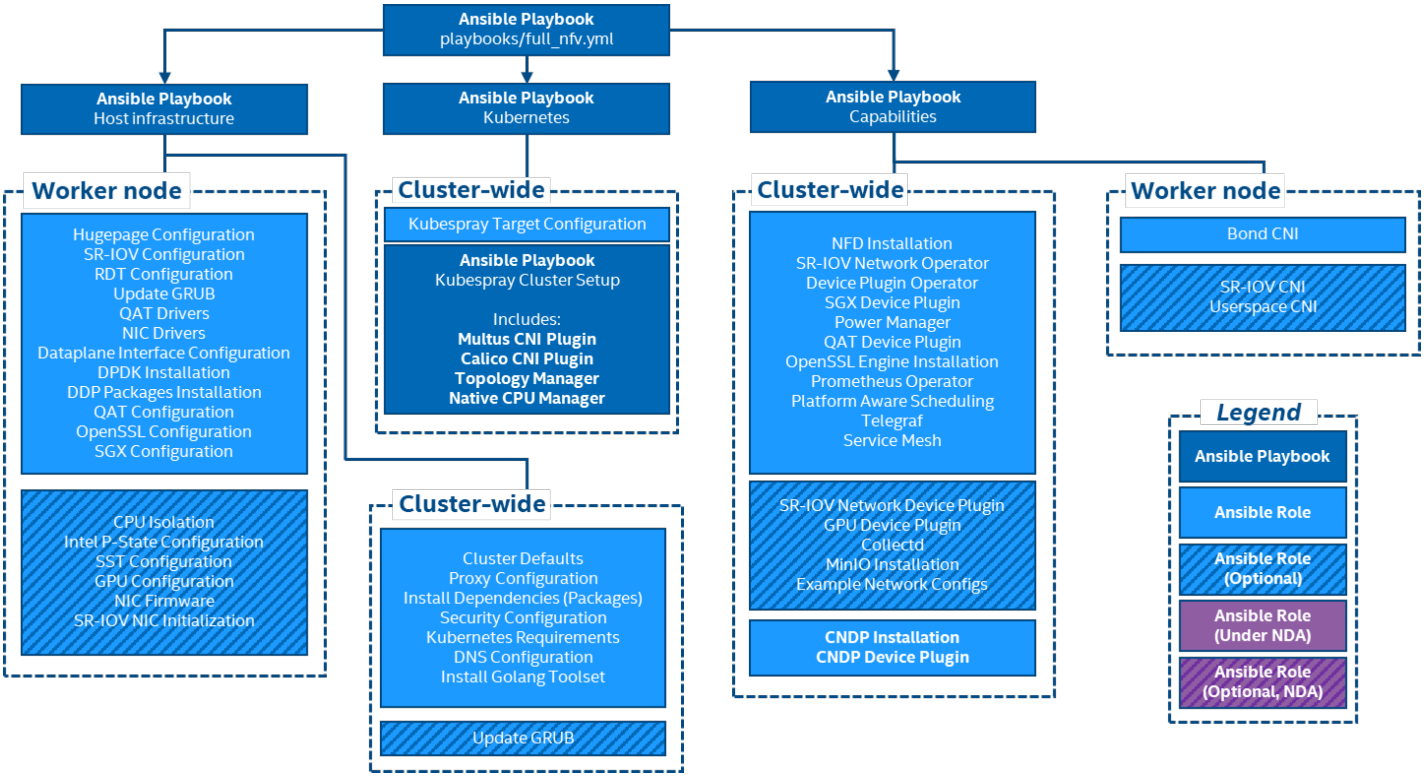


Figure 20. BMRA Ansible Playbooks – Full Configuration Profile Example[11]

[11] Refer to Optimization Notices for more information regarding performance and optimization choices in Intel software products.

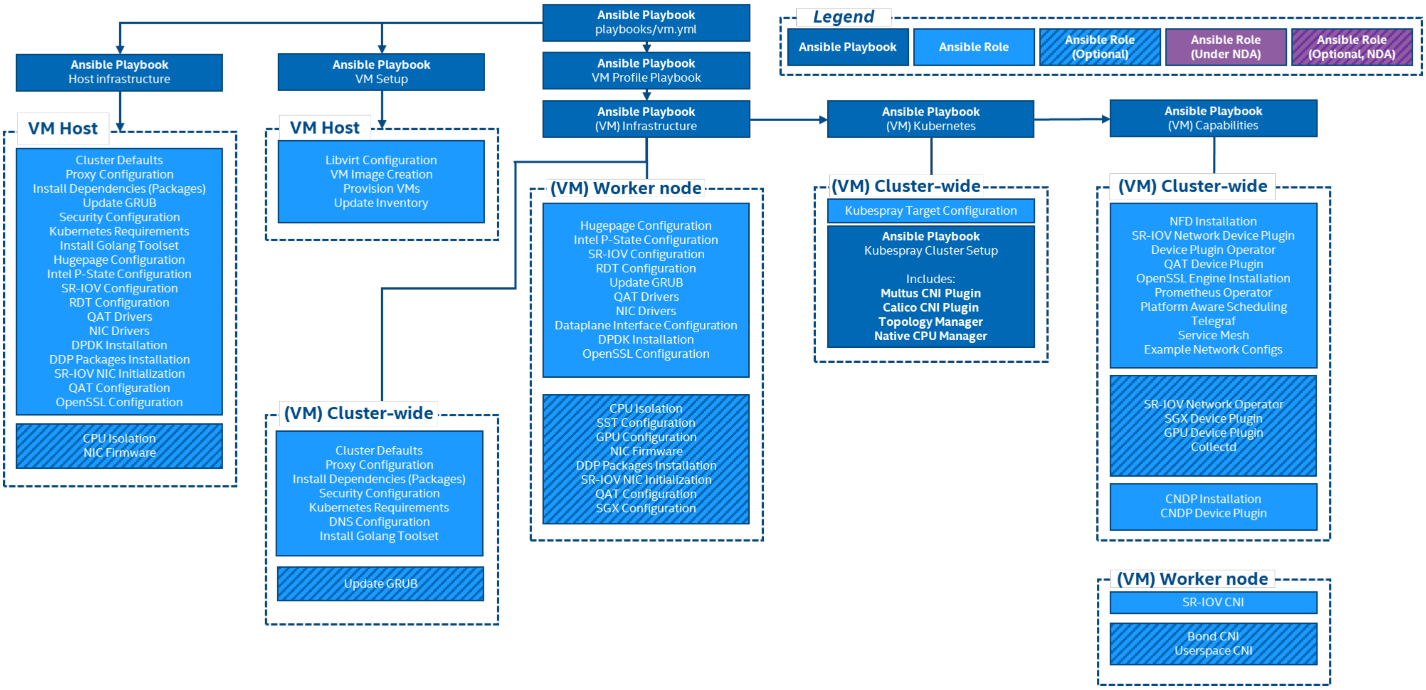


Figure 21. VMRA Ansible Playbook – Full Configuration Profile Example

5.3.1 Ansible Playbook Phases

Regardless of the selected Configuration Profile, the installation process always consists of three main phases. For VMRA, individual setups occur for VM Host and VMs as shown in Figure 21.

1. **Infrastructure Setup** (sub-playbooks located in `playbooks/infra/` directory)
   These playbooks modify kernel boot parameters and apply the initial system configuration for the nodes. Depending on the selected Configuration Profile, this includes:
   - Generic host OS preparation, e.g., 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, 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 VFs and binding to appropriate Linux kernel modules.
   - Network Adapter drivers and firmware updates, which help ensure that all latest capabilities such as DDP profiles are enabled.
   - Intel® Speed Select Technology (Intel® SST) configuration, which provides control over base frequency.
   - 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.
   - Configuration and provisioning of the virtualization layer when deploying the VMRA. Separate Infrastructure Setup procedures are executed for Host and VMs.
2. **Kubernetes Setup** (located in `playbooks/k8s/` directory)
   When a Kubernetes cluster is deployed, 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 support 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 is also a container registry instance deployed to store images of various control-plane Kubernetes applications, such as TAS or device plugins.
3. **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. For a VMRA deployment, system capabilities are installed on the VMs, not the host. The following capabilities are deployed:
   - Device plugins that allow using, for example, SR-IOV, QAT, and GPU devices in workloads running on cluster.
   - SR-IOV CNI plugin, Bond CNI plugin, and Userspace CNI plugin, 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.

  - Telemetry Aware Scheduling, which allows scheduling workloads based on telemetry data.
  - Full Telemetry Stack consisting of collectd, Kube-Prometheus, and Grafana, which give cluster and workload monitoring capabilities and act as a source of metrics that can be used in TAS to orchestrate scheduling decisions.

5.4 Deployment Using Ansible Playbooks

The following steps are required to obtain the Ansible playbook source code, prepare target servers, configure inventory and variable files, and deploy the Kubernetes cluster.

5.4.1 Prepare Target Servers

For each target server that will act as a Kubernetes control or worker node, make sure that it meets the following requirements:

- Python 3 is installed. The version depends on the target distribution.
- SSH is configured and the server is reachable from the Ansible host.
- Internet access on all target servers is mandatory.
- BIOS configuration matching the desired state is applied.

For detailed steps on how to build the Ansible host, refer to the appropriate user guide.

5.4.2 Get Ansible Playbook and Prepare Configuration Templates

On the Ansible host, select the Configuration Profile for deployment and export the environmental variable. For example, for Kubernetes **Basic** Configuration Profile deployment:

```
export PROFILE=basic
```

Install requirements needed by `render.py` script:

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

Generate example profiles for either BMRA or VMRA:

```
[BMRA] make k8s-profile PROFILE=$PROFILE ARCH=<spr,icx,clx> NIC=<cvl,fvl>
[VMRA] make vm-profile PROFILE=$PROFILE ARCH=<spr,icx,clx> NIC=<cvl,fvl>
```

5.4.3 Update Ansible Inventory File

Edit the `inventory.ini` file generated in the previous steps.

1. For a BMRA, specify all target servers by using actual hostnames and management IP addresses and update the `ansible_password` to match the SSH configuration of the target servers.
2. For a VMRA, specify the target VM host server by using actual hostname and management IP address and update the `ansible_user` and `ansible_password` to match the SSH configuration of the target server. In the `[vm_host]` section, update the hostname to match that defined in `[all]`. Do not uncomment any of the hostnames defined under `[kube_control_plane]`, `[etcd]`, and `[kube_node]`, as these are dynamically updated based on the number of virtual machines defined for the target VM host server in `host_vars`.

5.4.4 Update Ansible Host and Group Variables

Each Configuration Profile uses its own set of variables. Refer to the specific Configuration Profile section in the appropriate user guide. The following outlines the procedure:

1. Create `host_vars/<hostname>.yml` files for all worker nodes, matching their hostnames from the inventory file.
2. Edit `host_vars/<hostname>.yml` and `group_vars/all.yml` files to match your desired configuration.

5.4.5 Run Ansible Cluster Deployment Playbook

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

It is recommended to first install a set of OS-specific patches:

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

After completing the patches, the deployment can be started:

```
[BMRA] ansible-playbook -i inventory.ini playbooks/${PROFILE}.yml
[VMRA] ansible-playbook -i inventory.ini playbooks/vm.yml
```

Depending on the selected Configuration Profile, network bandwidth, storage speed, and other similar factors, the execution may take up to 30-40 minutes.

After the playbook finishes without any “Failed” tasks, you can proceed with the deployment validation described in the user guides.

5.4.6 Ansible Cluster Removal Playbook

If you want to clean up the environment to run a new deployment or remove a completed, failed, or incomplete deployment, run the provided Cluster Removal playbook (`redeploy_cleanup.yml`). It removes any previously installed Kubernetes and related plugins.

Appendix A Abbreviations

The following abbreviations are used in this document.

Table 10. Abbreviations

ABBREVIATION	DESCRIPTION
AEAD	Authentication Encryption with Associated Data
AF_XDP	Address Family eXpress Data Path
AMX	Advance Matrix Multiply
API	Application Programming Interface
AV1	AOMedia Video 1 video coding format
AVC	Advanced Video Coding video compression standard
AWS	Amazon Web Services
BIOS	Basic Input/Output System
BKC	Best Known Configuration
BMRA	Bare Metal Reference Architecture
BOM	Bill of Materials
CA	Certificate Authority
CDN	Content Delivery Network
CMTS	Cable Modem Termination System
CNCF	Cloud Native Computing Foundation
CNDP	Cloud Native Data Plane (CNDP)
CNF	Cloud Native Network Function
CNI	Container Network Interface
CO	Central Office
CoSP	Communications Service Provider
CPI	Cycles Per Instruction
CR	Custom Resource
CRD	Custom Resource Definition
CRI	Container Runtime Interface
CRS	Core Rule Set
CRUD	Create, Read, Update, and Delete
CSR	Certificate Signing Request
CU	Central Unit
CXL	Compute Express Link
DCAP	Datacenter Attestation Primitives
DDP	Dynamic Device Personalization (DDP)
DHCP	Dynamic Host Configuration Protocol
DLB	Intel® Dynamic Load Balancer (Intel® DLB)
DNS	Domain Name Service
DoS	Denial of Service
DP	Device Plugin
DPDK	Data Plane Development Kit (see DPDK)
DRAM	Dynamic Random Access Memory
DSA	Intel® Data Streaming Accelerator (Intel® DSA)
DU	Distributed/Distribution Unit
EPC	Electronic Product Code
FEC	Forward Error Correction

ABBREVIATION	DESCRIPTION
FPGA	Field-Programmable Gate Array
FW	Firmware
GPU	Graphics Processor Unit
HA	High Availability
HDD	Hard Disk Drive
HEVC	High Efficiency Video Coding video compression
HQM	Hardware Queue Manager
HSM	Hardware Security Model
HT	Hyper-Threading
IA	Intel® architecture
IAX	In-Memory Analytics
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® Scalable IOV	Intel® Scalable I/O Virtualization (Intel® Scalable IOV)
Intel® SecL – DC	Intel® Security Libraries for Data Center (Intel® SecL – DC)
Intel® SGX	Intel® Software Guard Extensions (Intel® SGX)
Intel® SST	Intel® Speed Select Technology (Intel® SST)
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)
IOMMU	Input/Output Memory Management Unit
IoT	Internet of Things
IPC	Instructions per Cycle
ISA	Instruction Set Architecture
I/O	Input/Output
K8s	Kubernetes
KMRA	Key Management Reference Application (KMRA)
LLC	Last Level Cache
LOM	LAN on Motherboard
LTS	Long Term Support
MANO	Management and Orchestration
MBM	Memory Bandwidth Monitoring (MBM)
MEC	Multi-Access Edge Control
ML	Machine Learning
MMIO	Memory-Mapped Input/Output
MPEG-2	Moving Picture Experts Group standard for digital television and DVD video
MSR	Model-Specific Register
mTLS	Mutual Transport Layer Security
NF	Network Function

ABBREVIATION	DESCRIPTION
NFD	Node Feature Discovery
NFV	Network Functions Virtualization
NIC	Network Interface Card
NPX	Network Platform Xeon
NTB	Non-Transparent Bridge
NUMA	Non-Uniform Memory Access
NVDIMM	Non-Volatile DIMM
NVM/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
OWASP	Open Web Application Security Project
P GW	Packet Gateway
PAS	Platform Aware Scheduling
PCI	Physical Network Interface
PCIe	Peripheral Component Interconnect Express
PF0	First physical function of the device
PKCS	Public-Key Cryptography Standard
PMD	Poll Mode Driver
PMU	Power Management Unit
POP	Post Office Protocol
PSP	Pod Security Policy
PXE	Preboot Execution Environment
QAT	Intel® QuickAssist Technology
QCT	Quanta Cloud Technology
QoS	Quality of Service
RA	Reference Architecture
RAN	Radio Access Network
RAS	Reliability, Availability, and Serviceability
RBAC	Role-Based Access Control
RDT	Intel® Resource Director Technology (Intel® RDT)
S3	Amazon Web Services Simple Storage Service
S-IOV	Intel® Scalable I/O Virtualization (Intel® Scalable IOV)
SATA	Serial Advanced Technology Attachment
SDN	Software-Defined Networking
SGX	Intel® Software Guard Extensions (Intel® SGX)
SIMD	Single Instruction, Multiple Data
SMF	Session Management Function
SNMP	Simple Network Management Protocol
SR-IOV	Single Root Input/Output Virtualization
SSD	Solid State Drive
SSH	Secure Shell Protocol
SVM	Shared Virtual Memory
TADK	Traffic Analytics Development Kit

ABBREVIATION	DESCRIPTION
TAS	Telemetry Aware Scheduling
TCP	Transmission Control Protocol
TCS	Trusted Certificate Service
TDP	Thermal Design Power
TLS	Transport Layer Security
TMUL	Tile Multiply
TPM	Trusted Platform Module
UDS	UNIX Domain Socket
UEFI	Unified Extensible Firmware Interface
UPF	User Plane Function
UPG	User Profile Gateway
v5GC	Virtual 5G Core
vBNG	Virtual Broadband Network Gateway
vCDN	Virtualized Content Delivery Network
vCMTS	Virtual Cable Modem Termination System
vCPE	Virtual Customer Premises Equipment
VF	Virtual Function
VLAN	Virtual LAN
VM	Virtual Machine
VMM	Virtual Machine Manager
VMRA	Virtual Machine Reference Architecture
VNF	Virtual Network Function
VNFM	Virtual Network Functions Manager
VP9	Video coding format for streaming over the internet
vPCRF	Virtual Policy and Changing Rules Function
VPP	Vector Packet Processing
VXLAN	Virtual Extensible LAN

Appendix B Reference Documentation

Collaterals, including technical guides and solution briefs that explain in detail the technologies enabled in the Network and Cloud Edge Reference System Architectures Portfolio, are available in the following locations: Network Transformation Experience Kits and Container Experience Kits.

Table 11. Reference Documents

REFERENCE	SOURCE
Network and Cloud Edge Container Bare Metal Reference System Architecture User Guide	https://networkbuilders.intel.com/solutionslibrary/network-and-cloud-edge-container-bare-metal-reference-system-architecture-user-guide
Network and Cloud Edge Virtual Machine Reference System Architecture User Guide	https://networkbuilders.intel.com/solutionslibrary/network-and-cloud-edge-virtual-machine-reference-system-architecture-user-guide
Network and Cloud Edge Reference System Architectures Release Overview Solution Brief	https://networkbuilders.intel.com/solutionslibrary/network-and-cloud-edge-reference-system-architectures-release-overview-solution-brief
Network and Cloud Edge Reference System Architectures on a Single Server Quick Start Guide	https://networkbuilders.intel.com/solutionslibrary/network-and-cloud-edge-reference-system-architectures-single-server-quick-start-guide
Network and Cloud Edge Reference System Architecture with FlexRAN™ Software – Setup on a Single Server Quick Start Guide	https://networkbuilders.intel.com/solutionslibrary/network-and-cloud-edge-reference-system-architecture-flexran-software-single-server-quick-start-guide
Intel® Ethernet Operator – Overview Solution Brief	https://networkbuilders.intel.com/solutionslibrary/intel-ethernet-operator-overview-solution-brief
Intel® Xeon® D Processor Technology – Training Video	https://networkbuilders.intel.com/intel-xeon-d-processor-technology-training-video
Intel® Speed Select Technology – Turbo Frequency (Intel® SST-TF) Overview User Guide	https://networkbuilders.intel.com/solutionslibrary/intel-speed-select-technology-turbo-frequency-intel-sst-tf-overview-user-guide
Intel® Deep Learning Boost - Boost Network Security AI Inference Performance in Google Cloud Platform (GCP) Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-deep-learning-boost-boost-network-security-ai-inference-performance-in-google-cloud-platform-gcp-technology-guide
Intel® Speed Select Technology – Performance Profile (Intel® SST-PP) Overview User Guide	https://networkbuilders.intel.com/solutionslibrary/intel-speed-select-technology-performance-profile-intel-sst-pp-overview-user-guide
Intel® AVX-512 and Intel® QAT - Accelerate WireGuard Processing with Intel® Xeon® D-2700 Processor Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-avx-512-and-intel-qat-accelerate-wireguard-processing-with-intel-xeon-d-2700-processor-technology-guide
Intel® AVX-512 - FP16 Instruction Set for Intel® Xeon® Processor Based Products Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-avx-512-fp16-instruction-set-for-intel-xeon-processor-based-products-technology-guide
Node Feature Discovery - Orchestrating Intelligent Workload Scheduling Technology Guide	https://networkbuilders.intel.com/solutionslibrary/node-feature-discovery-orchestrating-intelligent-workload-scheduling
Platform Telemetry Insights – Streaming Telemetry Insights Provider Technology Guide (NDA - For access, contact your Intel representative)	https://dpgresources.intel.com/asset-library/platformtelemetryinsights-streaming-telemetry-insights-provider-techguide/
Sustainability – Power Insights Kit Technology Guide (NDA - For access, contact your Intel representative)	https://dpgresources.intel.com/asset-library/sustainability-power-insights-kit-technology-guide/
Zero Trust – Rethink Zero Trust with Intel Confidential Computing Technology Guide	https://networkbuilders.intel.com/solutionslibrary/zero-trust-rethink-zero-trust-with-intel-confidential-computing-technology-guide
Intel® Ethernet 800 Series - Application Device Queues (ADQ) in a Cloud Native Environment Solution Brief (NDA - For access, contact your Intel representative)	https://dpgresources.intel.com/asset-library/intel_ethernet_800_series_application_device_queues_cloud_native_env_solution_brief/
Galois Field New Instructions (GFNI) Technology Guide	https://networkbuilders.intel.com/solutionslibrary/galois-field-new-instructions-gfni-technology-guide
Galois Field New Instructions - Method for Calculating Toeplitz Hash Using GFNI Technology Guide	https://networkbuilders.intel.com/solutionslibrary/galois-fields-new-instructions-method-for-calculating-toeplitz-hash-using-gfni-technology-guide
Intel® Ethernet Controller - Predictable Load Distribution Using Partial Toeplitz Hash Collections Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-ethernet-controller-predictable-load-distribution-using-partial-toeplitz-hash-collections-technology-guide
Cloud Native Data Plane (CNDP) - Overview Technology Guide	https://networkbuilders.intel.com/solutionslibrary/cloud-native-data-plane-cndp-overview-technology-guide

REFERENCE	SOURCE
Reliability, Availability, and Insights - Reliability, Availability, and Serviceability in Telecommunication Networks Technology Guide	https://networkbuilders.intel.com/solutionslibrary/reliability-availability-and-insights-reliability-availability-and-serviceability-in-telecommunication-networks-technology-guide
Intel® Turbo Boost Technology - Using Intel® Turbo Boost Technology for Communications Workloads Technology Guide (NDA - For access, contact your Intel representative)	https://dpgresources.intel.com/asset-library/intel-turbo-boost-technology-using-intel-turbo-boost-technology-for-communications-workloads-technology-guide/
Advanced Networking Features in Kubernetes and Container Bare Metal Application Note	https://builders.intel.com/docs/networkbuilders/adv-network-features-in-kubernetes-app-note.pdf
CPU Management - CPU Pinning and Isolation in Kubernetes Technology Guide	https://builders.intel.com/docs/networkbuilders/cpu-pin-and-isolation-in-kubernetes-app-note.pdf
Enhanced Utilization Telemetry for Polling Workloads with collectd and the Data Plane Development Kit (DPDK) User Guide	https://networkbuilders.intel.com/solutionslibrary/enhanced-utilization-telemetry-for-polling-workloads-with-collectd-and-the-data-plane-development-kit-dpdk-user-guide
Intel Device Plugins for Kubernetes Application Note	https://builders.intel.com/docs/networkbuilders/intel-device-plugins-for-kubernetes-appnote.pdf
Intel® Ethernet 700/800 Series - Dynamic Device Personalization Support for CNF with Kubernetes Technology Guide	https://builders.intel.com/docs/networkbuilders/intel-ethernet-controller-700-series-dynamic-device-personalization-support-for-cnf-with-kubernetes-technology-guide.pdf
Intel® Ethernet Controller 700 Series GTPv1 - Dynamic Device Personalization Application Note	https://networkbuilders.intel.com/solutionslibrary/intel-ethernet-controller-700-series-gtpv1-dynamic-device-personalization
Intel® Ethernet Controller E810 Dynamic Device Personalization Package (DDP) for Telecommunications Technology Guide	https://www.intel.com/content/www/us/en/products/details/ethernet/800-controllers/e810-controllers/docs.html
Node Feature Discovery Application Note	https://builders.intel.com/docs/networkbuilders/node-feature-discovery-application-note.pdf
QCT Validated Kubernetes Platform with Enhanced Platform Awareness Technical Brief	https://networkbuilders.intel.com/solutionslibrary/qct-validated-kubernetes-platform-with-enhanced-platform-awareness
Telemetry Aware Scheduling (TAS) - Automated Workload Optimization with Kubernetes (K8s) Technology Guide	https://builders.intel.com/docs/networkbuilders/telemetry-aware-scheduling-automated-workload-optimization-with-kubernetes-k8s-technology-guide.pdf
Topology Management - Implementation in Kubernetes Technology Guide	https://builders.intel.com/docs/networkbuilders/topology-management-implementation-in-kubernetes-technology-guide.pdf
Intel® Speed Select Technology – Base Frequency (Intel® SST-BF) with Kubernetes Application Note	https://networkbuilders.intel.com/solutionslibrary/intel-speed-select-technology-base-frequency-with-kubernetes-application-note
Intel® Speed Select Technology Core Power (Intel® SST-CP)	https://www.kernel.org/doc/html/latest/admin-guide/pm/intel-speed-select.html#intel-r-speed-select-technology-core-power-intel-r-sst-cp
Secure the Network Infrastructure - Secure Cloud Native Network Platforms User Guide	https://networkbuilders.intel.com/solutionslibrary/secure-the-network-infrastructure-secure-cloud-native-network-platforms-user-guide
Closed Loop Automation - Telemetry Aware Scheduler for Service Healing and Platform Resilience Demo	https://networkbuilders.intel.com/closed-loop-automation-telemetry-aware-scheduler-for-service-healing-and-platform-resilience-demo
Closed Loop Automation - Telemetry Aware Scheduler for Service Healing and Platform Resilience White Paper	https://builders.intel.com/docs/networkbuilders/closed-loop-platform-automation-service-healing-and-platform-resilience.pdf
Intel Telemetry Insights Reports (For access, contact your Intel Platform Application Engineer)	https://www.intel.com/content/www/us/en/secure/design/internal/content-details.html?DocID=645751
Intel® RDT	https://wiki.opnfv.org/display/fastpath/Intel_RDT
Intel® Server GPU Features	https://www.intel.com/content/www/us/en/products/discrete-gpus/server-graphics-card.html
Intel® Server GPU Data Center Graphics for High-Density Cloud Gaming and Media Streaming	https://www.intel.com/content/www/us/en/products/docs/discrete-gpus/server-graphics-card-product-brief.html
Intel® Server GPUs for Cloud Gaming and Media Delivery	https://www.intel.com/content/www/us/en/products/docs/discrete-gpus/server-graphics-solution-brief.html
Intel® Server GPU Specifications	https://ark.intel.com/content/www/us/en/ark/products/210576/intel-server-gpu.html
Intel® GPU Device Plugin for Kubernetes	https://github.com/intel/intel-device-plugins-for-kubernetes/tree/main/cmd/gpu_plugin

REFERENCE	SOURCE
Intel® AVX-512 - Packet Processing with Intel® AVX-512 Instruction Set Solution Brief	https://networkbuilders.intel.com/solutionslibrary/intel-avx-512-packet-processing-with-intel-avx-512-instruction-set-solution-brief
Intel® AVX-512 - Instruction Set for Packet Processing Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-avx-512-instruction-set-for-packet-processing-technology-guide
Intel® AVX-512 - Writing Packet Processing Software with Intel® AVX-512 Instruction Set Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-avx-512-writing-packet-processing-software-with-intel-avx-512-instruction-set-technology-guide
Ubiquitous Availability of Crypto Technologies Solution Brief	https://networkbuilders.intel.com/solutionslibrary/ubiquitous-availability-of-crypto-technologies-solution-brief
Introduction – Sapphire Rapids Processor: External Design Specification, Volume One: Architecture (For access, contact your Intel Platform Application Engineer)	https://edc.intel.com/content/www/us/en/secure/design/confidential/products-and-solutions/processors-and-chipsets/eagle-stream/sapphire-rapids-server-processor-external-design-specification-volume-one-arch/0.9/
Intel® Architecture Instruction Set Extensions and Future Features Programming Reference	https://wiki.ith.intel.com/download/attachments/1508654814/architecture-instruction-set-extensions-programming-reference.pdf?version=1&modificationDate=1581680607737&api=v2
Intel® Software Guard Extensions (Intel® SGX) – Key Management Reference Application (KMRA) on Intel® Xeon® Processors Technology Guide	https://networkbuilders.intel.com/solutionslibrary/intel-sgx-kmra-on-intel-xeon-processors-technology-guide
Intel® Software Guard Extensions (Intel® SGX) - Key Management Reference Application (KMRA) on Intel® Xeon® Processors User Guide	https://networkbuilders.intel.com/solutionslibrary/intel-sgx-kmra-on-intel-xeon-processors-user-guide
Platform Characteristics by Network Location (For access, contact your Intel Platform Application Engineer)	https://www.intel.com/content/www/us/en/secure/design/internal/content-details.html?DocID=576543

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Intel technologies may require enabled hardware, software or service activation.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.