White Paper

Service Providers (SPs), Multiple System Operators (MSOs), Cable Operators

Intel-Dell Verified Reference Configuration for vCMTS on Red Hat OCP

Scalable, Cloud-Native vCMTS with Intel® Verified Reference Configuration for Next-Generation Broadband Services

Authors

Timothy Miskell
Cloud Software Architect, Intel

Thuy Nguyen
Head of Cable Segment, Intel

Muhammad Siddiqui
Solutions Architect, Intel

John Pezzulli
Chief Technology Office - MSO, Dell

Table of Contents

Introduction

As cable operators transition to cloud-native architectures, the need for scalable, high-performance systems becomes more critical. Intel's Verified Reference Configuration (VRC) for virtualized Cable Modem Termination Systems (vCMTS) addresses this by providing a robust, pre-validated solution.

This reference design, built on the 4th Gen Intel® Xeon® Scalable processors and deployed on Red Hat OpenShift 4.14, delivers a flexible and efficient platform for managing broadband services. By leveraging Intel's advanced processing technology and Dell PowerEdge R660 servers, operators can achieve faster deployment cycles, streamlined management, and enhanced performance to support modern service demands. This whitepaper outlines the key components, system configuration, and best practices for deploying vCMTS in a cloud-native environment, enabling operators to confidently scale their networks and deliver next-generation connectivity.

Intel VRC for vCMTS

When network operators, service providers, cloud service providers, or enterprise infrastructure companies choose an Intel VRC for vCMTS they will be able to deploy the network function virtualized (NFV) applications more efficiently and securely than ever before. End users spend less time, effort, and expense evaluating hardware and software options. Intel Verified Reference Configurations help end users simplify design choices by bundling hardware and software pieces together while making the high performance more predictable.

Intel Verified Reference Configurations for vCMTS on a multi-node architecture that consists of a master and a set of worker nodes. The Intel Verified Reference Configuration for vCMTS provides all required resources to implement a software-defined infrastructure that resides within each cloud server instance and is controlled by the orchestrator. The master node is intended to be used for control, signaling, and management implementing the Containerized Network Function (CNF) management.

The Intel Verified Reference Configuration for vCMTS is part of the Intel VRC for Network Function Virtualization Infrastructure Forwarding Platform (NFVI-FP) program, which support multiple high throughput workloads such as a virtualized Broadband Network Gateway (vBNG), a 5G User Plane Function (UPF), along with a Virtualized Access Gateway Function (vAGF).

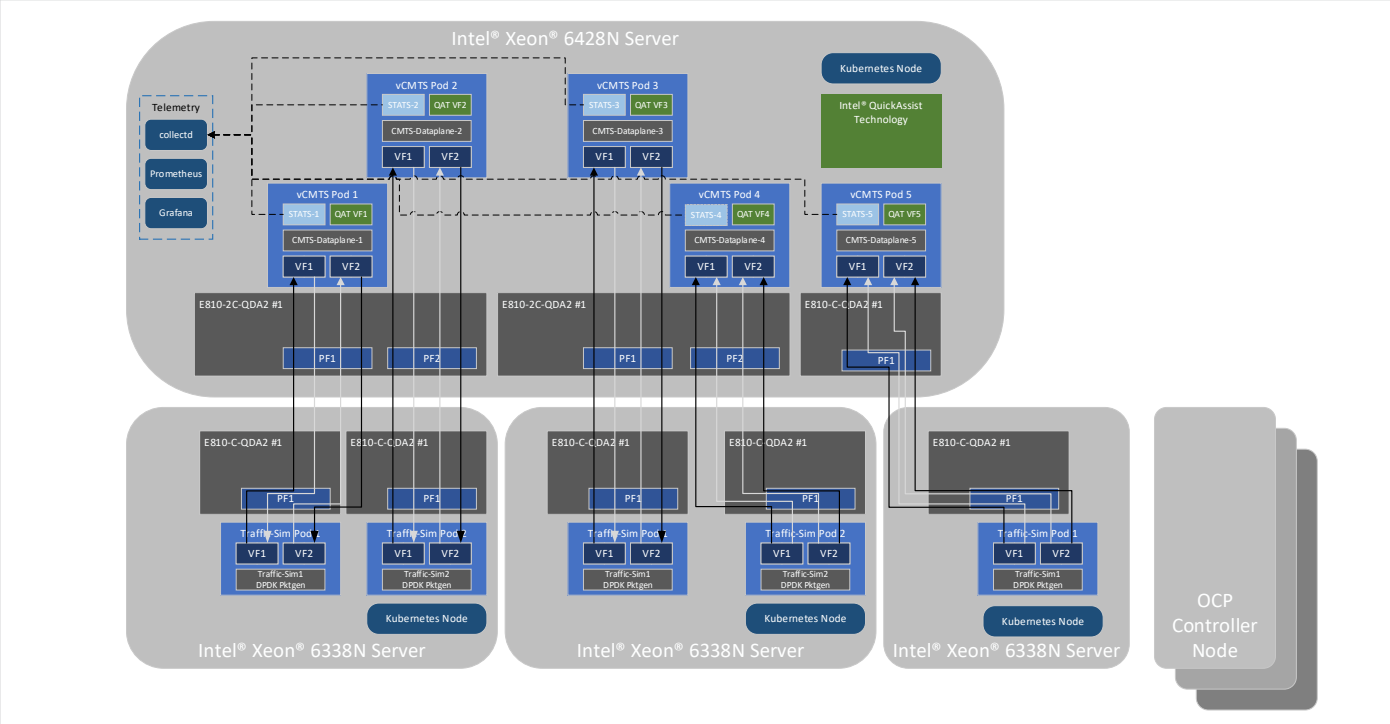

Figure 1. Platform Configuration for the Intel® vCMTS Deployed on a Dell PowerEdge R660 Server

System Overview

The Intel® vCMTS reference data plane environment consists of a vCMTS data plane node and a traffic generator node with the following specifications:

Figure 2. Dell PowerEdge R660 Server

vCMTS Data Plane Node:

- 4th Generation Intel® Xeon® 6428N Scalable Processor
- Intel® E810 100 GbE and E810 200 GbE ethernet adapters
- Intel® QuickAssist Technology (Intel QAT)

vCMTS Traffic Generator Node:

- Dell PowerEdge R660 2u Server
- 4th Generation Intel® Xeon® 6428N Scalable Processor
- Intel® E810 100 GbE and E810 200 GbE ethernet adapters

Note: This whitepaper describes a sample system configuration for baseline performance. The Intel® vCMTS reference data plane application supports additional processor and NIC configurations not covered here. For this VRC, the system is deployed on Red Hat OpenShift Container Platform, though other deployment models, including Intel® Container Bare Metal Reference Architecture and bare metal Linux, are also supported.

Supported Environments

Red Hat OpenShift Container Platform

Red Hat OpenShift Container Platform (OCP) is an enterprise-grade stack designed for on-premises and cloud-based deployments. Built on Red Hat Enterprise Linux and Kubernetes, OCP provides a secure and scalable multi-tenant operating system for enterprise-grade applications, while delivering integrated application runtimes and libraries.

Red Hat OCP comes with a streamlined, automatic install so users can be up and running with Kubernetes as quickly as possible. Once installed, OCP uses operators for push-button, automatic platform updates for the container host, cluster, and application services running on the cluster. The Intel® vCMTS reference data plane has been adapted to deploy seamlessly on a pre-installed OpenShift Cluster.

Kubernetes Orchestrated Container Based Deployments

Under an Under an OCP deployment, multiple containerd containers run DPDK-based DOCSIS MAC upstream and downstream data plane instances as Kubernetes pods, each representing a cable service group. On the vCMTS traffic generator, containerd containers run DPDK pktgen instances to simulate corresponding DOCSIS traffic. DOCSIS control-plane details reside in a JSON file with subscriber cable-modem information, and optional upstream schedulers can be configured to manage upstream bandwidth. Telemetry functions run in containerd containers under Kubernetes, with collectd gathering data plane statistics and KPIs into a Prometheus time-series database. A Grafana dashboard provides visualizations of these metrics via Prometheus queries.

Sample Platform Configuration

The sample platform configuration shown in the Figure below supports vCMTS data plane processing for a configurable number of service-groups on a BMRA or OCP Kubernetes orchestrated platform. Various mappings of cores and VFs to service-group pods are supported.

Please note that if Upstream Scheduling is enabled in external mode, additional pods are deployed for this function. Each Upstream Scheduling pod allocates upstream bandwidth for multiple vCMTS pods.

In a bare metal Linux deployment the applications are run as processes directly on the host OS rather than in containers and there are no Kubernetes controller nodes required. However, all other components on the diagram remain true.

vCMTS Reference Data Plane Pipeline

The Intel® vCMTS reference data plane includes a DOCSIS MAC data plane compliant with DOCSIS 3.1 specifications (MULPI, DEPI, UEPI, SEC) and built on the DPDK framework. Its primary purpose is to characterize packet-processing performance and power consumption on Intel® Xeon® platforms. Figure 2 shows a high-level view of the upstream and downstream packet-processing pipelines, with the downstream pipeline implemented as two stages (upper and lower MAC). The DPDK API used for each DOCSIS MAC function is also indicated. Each vCMTS instance, deployed as a Kubernetes pod, handles all subscriber traffic for a specific cable service group.

Red Hat OpenShift Container Platform Software Stack

The Intel® vCMTS reference data plane can run on the Red Hat OpenShift Container Platform cloud-native software stack as shown in the Figure below. The vCMTS upstream and downstream data plane processing for individual cable service-groups run in crio-o containers on the RHEL OS, allowing them to be instantiated and scaled independently.

The entire system including applications, telemetry, power-management and infrastructure management is orchestrated by Kubernetes, with both Red hat and community operators being used for resource management functions such as assignment of SR-IOV interfaces for Intel® NIC and QAT devices as well as CPU isolation from the Linux kernel.

The python-based vCMTS platform management tool vcmts-pm, which runs as a container on the OCP deployment, provides commands to abstract and simplify the configuration of vCMTS components before handing over management to Kubernetes.

Figure 3. RHOCP Software Stack

CPU Core Management

Kubernetes Deployments (BMRA/OCP)

In a Kubernetes deployment, CPU core allocation is managed by the Kubernetes Native CPU Manager, which manages a shared pool of CPU cores from which cores may be allocated for exclusive use by upstream and downstream data plane containers.

Initially the CPU core pool contains all of the available CPU cores in the compute node.

The vCMTS data plane pods are assigned the QoS class of "Guaranteed" so that when containers are created for these pods by the kubelet process, CPU cores are removed from the shared pool and allocated exclusively for the lifetime of the container. Any existing containers using these cores are migrated to another core.

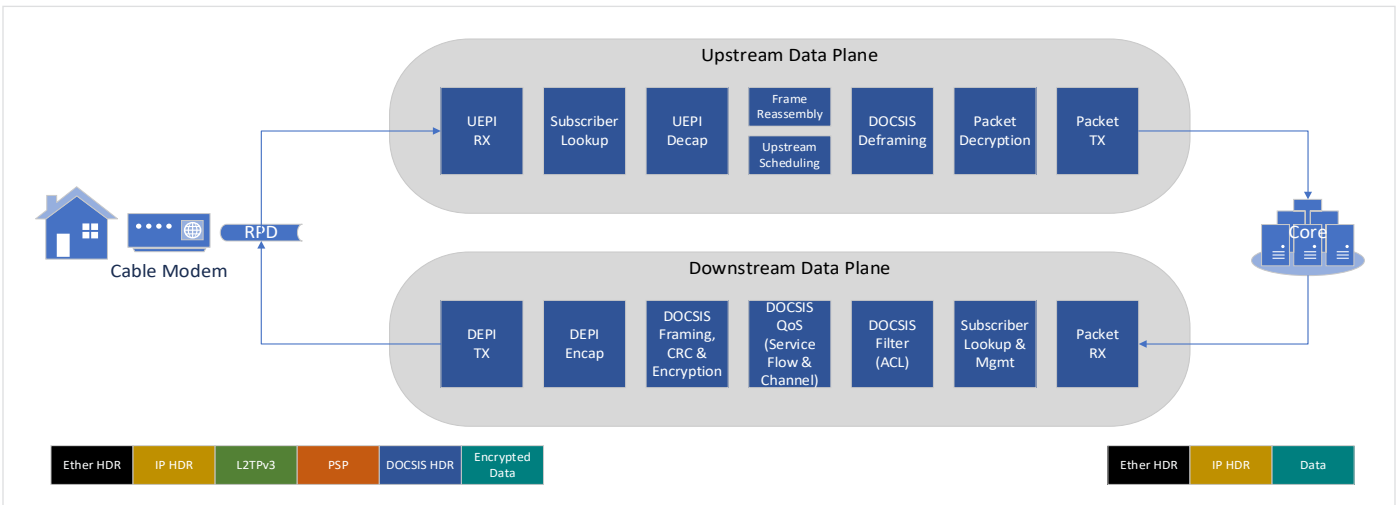

Figure 4. Intel vCMTS Data plane Packet Processing Pipeline

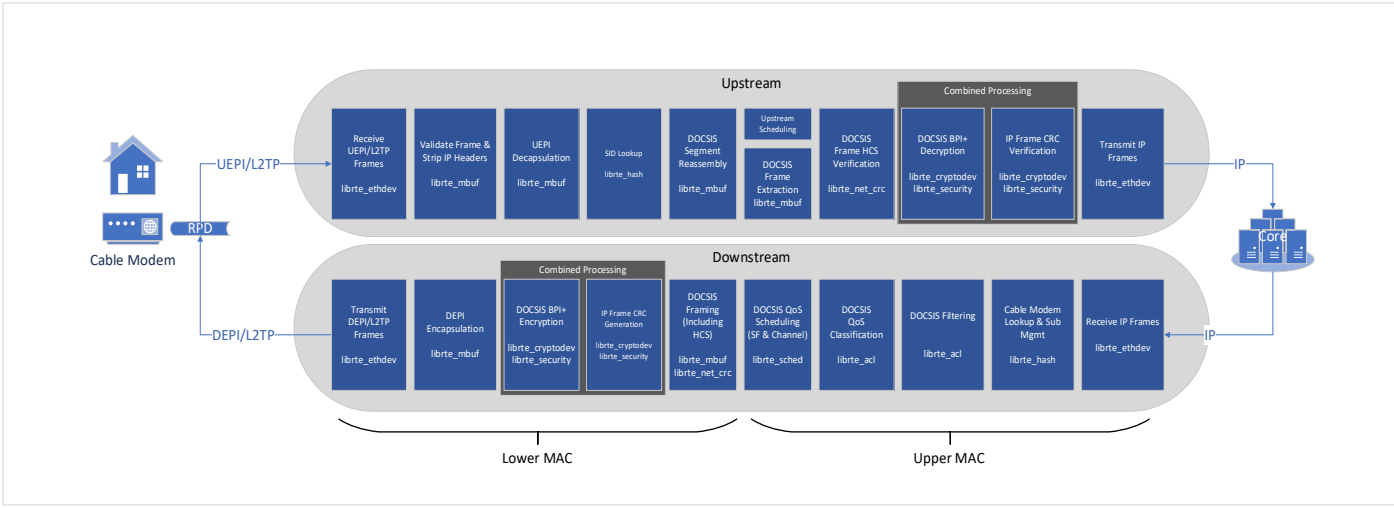

Figure 5. Intel vCMTS Data plane Packet Processing Pipeline (Detailed)

In this way each vCMTS data plane pod can request exclusive cores for its upstream and downstream data plane threads, as well as for Upstream Scheduling if enabled. The native CPU manager provides CPU core isolation which is critical for data-plane performance.

To ensure CPU Manager is working correctly for the vCMTS use case, ensure that in the kubelet configuration file that the cpuManagerPolicy is set to "static" additionaly check that the cpuManagerReconcilePeriod is set to "5s". If necessary restart the kubelet service or the node itself. Refer to the following guide for further information: https://docs.openshift.com/container-platform/4.14/scalability_and_performance/using-cpu-manager.html .

Sample CPU Core Layout

The Figure below shows an example of a CPU core layout on a 32-core dual-processor CPU. In this example CPU cores are allocated for upstream and downstream data plane processing and for Upstream Scheduling (all of which represents a Cable service-group) within a vCMTS Data plane pod. Two sibling hyper-threads of a core are utilized for downstream data plane processing per vCMTS Data plane pod. The upstream data plane processing utilizes one sibling hyper-thread of a core per pod. In this example, the Upstream Scheduler is deployed in "internal" mode, where it occupies the sibling of the upstream data plane hyper-thread and provides upstream scheduling functionality for the associated data plane. The Upstream Scheduler also shares the hyper-thread with management processing such as statistics collection.

An alternative "external" Upstream Scheduler mode is also available, where specific cores are assigned for Upstream Scheduler pods, and a single hyper-thread sibling handles the upstream scheduling for 2 Cable service-groups. In this case, the upstream data plane processing within the vCMTS Data plane pod shares a core with management processing only.

In this case there are 13x vCMTS Service-Groups deployed on 32 CPU Cores with Upstream Scheduling:

- 2 Cores for OS, Telemetry, Orchestration
- 4 Cores reserved for Control Plane
- 13 Cores for 13x DOCSIS MAC Upstream Data-plane instances, Upstream Scheduler & Mgmt
- 13 Cores for 13x DOCSIS MAC Downstream Data-plane instances

Figure 6. Sample CPU Core Layout

Network Interface Configuration

The network interface ports of the vCMTS data plane and traffic generator nodes should be interconnected by DAC cables via an Ethernet switch. Traffic routes between corresponding Pktgen and vCMTS data plane instances by MAC learning in the switch, since their MAC addresses are derived from service-group IDs. Specific switch configuration details are beyond the scope of this document.

If an Ethernet switch is not available, you may connect the NIC ports of the vCMTS data plane and traffic generator nodes directly via DAC cables, but care must be taken to ensure the proper physical alignment of Pktgen and vCMTS data plane NIC ports. A helper script is provided in the release package to facilitate this process, and instructions are included later in this guide once the required environment settings are configured. When vCMTS data plane traffic starts, each traffic generator instance sends an ARP request to establish a link with its corresponding vCMTS data plane instance.

NICs must be installed in the appropriate CPU-affinitized PCIe slots for balanced I/O. You can verify the NIC layout by running the provided command; in the example given, only the 100 GbE E810 NICs handle vCMTS data plane traffic. Device ports whose most significant address bit is unset are affinitized to CPU socket 0, while those with the bit set are affinitized to CPU socket 1, resulting in a balanced I/O configuration.

```
$ lspci | grep -i net
0d:00.0 Ethernet controller: Intel Corporation
Ethernet Controller E810-C for QSFP (rev 02)
0f:00.0 Ethernet controller: Intel Corporation
Ethernet Controller E810-C for QSFP (rev 02)
22:00.0 Ethernet controller: Intel Corporation
Ethernet Controller E810-C for QSFP (rev 02)
22:00.1 Ethernet controller: Intel Corporation
Ethernet Controller E810-C for QSFP (rev 02)
b5:00.0 Ethernet controller: Intel Corporation
Ethernet Controller E810-C for QSFP (rev 02)
b7:00.0 Ethernet controller: Intel Corporation
Ethernet Controller E810-C for QSFP (rev 02)
```

Memory Module Configuration

It is very important to ensure that DRAM modules are installed correctly in the server so that all memory channels are utilized.

For example, for the Intel® Xeon® 6428N scalable processor there are 8 memory channels per CPU socket. In this case a minimum of 16 DRAM modules are required to utilize all memory channels. Furthermore, modules should be installed in the correct color coded slots for optimum channel memory utilization. In general, it is recommended to use high-speed memory modules if possible.

DRAM module layout on the system can be determined by running the command below.

```
$ lshw -c memory
  *-memory:0
       description: System Memory
       physical id: 1000
       slot: System board or motherboard
       size: 1TiB
       capacity: 12TiB
       capabilities: ecc
       configuration: errordetection=multi-bit-
ecc
     *-bank:0
          description: DIMM Synchronous
Registered (Buffered) 4800 MHz (0.2 ns)
          product: HMCG94AEBRA109N
          vendor: 00AD063200AD
          physical id: 0
          serial: 868BE9AB
          slot: A1
          size: 64GiB
          width: 64 bits
          clock: 505MHz (2.0ns)
     *-bank:1
          description: DIMM Synchronous
Registered (Buffered) 4800 MHz (0.2 ns)
          product: HMCG94AEBRA109N
          vendor: 00AD063200AD
          physical id: 1
          serial: 868BEA16
          slot: A2
          size: 64GiB
          width: 64 bits
          clock: 505MHz (2.0ns)
```

System Configuration

The following is a sample system configuration for the Intel® vCMTS reference data plane v22.10.0 application:

vCMTS Dataplane Node

Hardware	Description
Processor	1 x Intel® Xeon® Gold 6428N Processor, 1.8Ghz, 32 Core
Memory	16 x 16GB Dual Rank DDR5
Storage	SSD 480GB
Network Interface Card (NIC)	2x Intel(R) Ethernet Network Adapter E810-C-QDA2 100 GbE (x16 PCIe) or 4x Intel(R) Ethernet Network Adapter E810 XXV710-DA2 25 GbE
OCP Software	**Description**
Host OS	Red Hat Enterprise Linux CoreOS release 4.14, 5.14.0-284.55.1.rt14.340.el9_2.x86_64
OCP	OCP 4.14
Linux Container	cri-o 1.27.4-3.rhaos4.14.git914bcba.el9
Container Orchestrator	Kubernetes v1.27.11+d8e449a
Common Software	**Description**
DPDK	DPDK v22.07, intel-ipsec-mb v1.3
vCMTS	Intel(R) vCMTS Reference Data plane v22.10.0
Statistics	Collectd v5.12.0, Prometheus 2.35.0, Grafana 8.5.3 (version numbers may differ between deployment types

vCMTS Traffic Generator Node

Hardware	Description
Processor	1 x Intel® Xeon® Gold 6428N Processor, 1.8Ghz, 32 Core
Memory	16 x 16GB Dual Rank DDR5
Storage	SSD 480GB
Network Interface Card (NIC)	2x Intel(R) Ethernet Network Adapter E810-C-QDA2 100 GbE (x16 PCIe) or 4x Intel(R) Ethernet Network Adapter E810 XXV710-DA2 25 GbE
OCP Software	**Description**
Host OS	Red Hat Enterprise Linux CoreOS release 4.14, 5.14.0-284.55.1.rt14.340.el9_2.x86_64
OCP	OCP 4.14
Linux Container	cri-o 1.27.4-3.rhaos4.14.git914bcba.el9
Container Orchestrator	Kubernetes v1.27.11+d8e449a
Common Software	**Description**
DPDK	DPDK v22.07, intel-ipsec-mb v1.3
Traffic Generator	DPDK Pktgen v19.11.0

Benchmarks

The following section provides a set of benchmarks that illustrate the scaling possibilities of the Intel® vCMTS data plane application on RHOCP 4.14 with Dell PowerEdge R660. The Key Performance Indicators (KPIs) in this case include, the upstream throughput, the downstream throughput, the packet loss rate, along with the CPU power consumption and the wall power consumption.

The following tables display the specific test conditions used in order to measure the aforementioned KPIs:

Test Conditions for vCMTS Benchmarks	
Total Number of Service Groups	{1, 2, ..., 27, 28}
Total Number of Subscribers	{300, 600, ..., 8,100, 8,400}
Channel Configuration	6 OFDM
AES	Enabled
Downstream QAT Crypto Offload	Disabled
Traffic Profile	imix2: ▪ Upstream: 65%: 84B, 18%: 256B, 17%: 1280B ▪ Downstream: 3%: 68B, 1%: 932B, 96%: 1520B
Power Management	Disabled
DOCSIS Upstream Scheduler	Internal

Test Conditions for vCMTS Benchmarks Continued	
Core Configuration	1us1t_1ds2t
Downstream Cryptodev scheduler	packet-size-distr
Downstream Cryptodev Scheduler Packet Size Threshold (B)	128
Downstream QAT Cipher CRC Offload Check	Disabled
Application Stats Capture	Disabled
CPU Cycle Count Stats Capture	Disabled
Latency Stats Capture	Disabled

Benchmark Results

The below benchmarks evaluated the Intel vCMTS Reference Application deployed on Red Hat OpenShift 4.14, running on Dell PowerEdge R660 servers featuring 4th Generation Intel® Xeon® Scalable processors and Intel® 800 Series Ethernet Adapters. The tests focused on how effectively the system could scale upstream and downstream throughput—measured in millions of packets per second (MPPs)—while maintaining zero packet loss (ZPL) as the number of service groups increased.

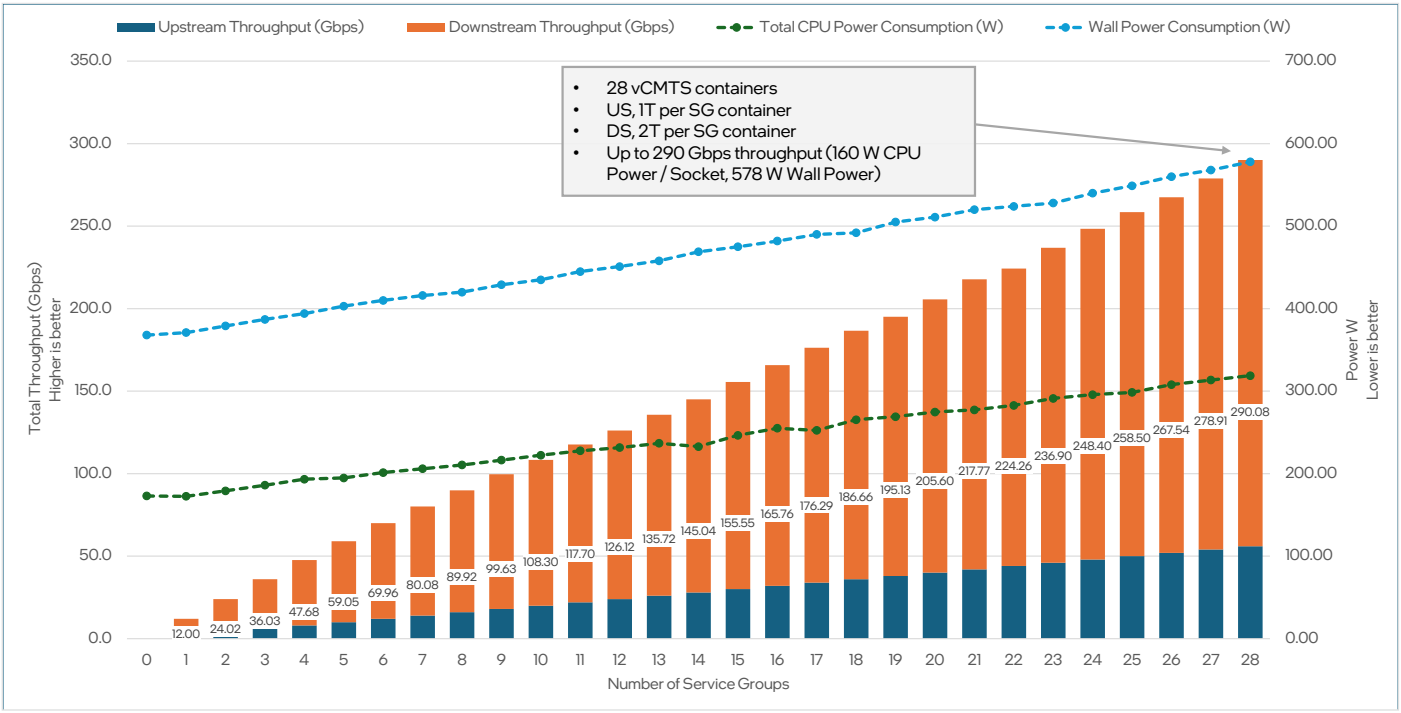

Figure 7. Intel vCMTS Aggregate Bi-Directional Throughput (Gbps) and Power Consumption (W) vs. Number of Service Groups

The scaling tests demonstrated that as more service groups were added, the Intel® vCMTS Reference Application maintained both performance and efficiency. The system delivered line-rate forwarding, preserving data integrity and providing a consistent, high-quality user experience. At full load, CPU and wall power consumption remained under control, showcasing the platform's ability to handle intensive workloads without excessive energy usage.

Key Outcomes:

- Achieved up to 290 Gbps of aggregate bi-directional throughput for 28 service groups at ZPL.
- Demonstrated that increasing service groups does not compromise throughput or system reliability.
- Validated energy efficiency by maintaining stable CPU and wall power consumption at higher loads.
- Confirmed that an Intel-based vCMTS solution can meet next-generation cable service needs with both performance and sustainability in mind.
- Showcased smooth interoperability within a containerized environment on Red Hat OpenShift* 4.14.
- Reinforced the platform's suitability for flexible, future-ready deployments that reduce complexity and operational costs.

Intel vCMTS Cluster Preparation

Common Server Preparation

A number of steps are required to prepare the vCMTS data plane and traffic generator servers prior to the software installation, regardless of the deployment type.

Intel vCMTS Data plane Server BIOS Settings

The Dell PowerEdge R660 server is configured with "System Profile" set to "Custom" and the "Workload Profile" set to "NFVI FP Optimized Turbo Profile". In this case all of the necessary BIOS settings are configured appropriately for the vCMTS data plane server.

The following table provides a list of the specific System BIOS settings that are required on the vCMTS data plane server.

Bios Setup Menu	BIOS Option	Setting
Advanced → Processor Configuration	Intel(R) Hyper-Threading Tech	Enabled
Advanced → Integrated IO Configuration	Intel(R) VT for Directed I/O	Enabled
Advanced → Power & Performance	CPU Power and Performance Profile	Balanced Performance
Advanced → Power & Performance → CPU P State Control	Hardware P-states (HWP)	Disabled
Advanced → Power & Performance → CPU P State Control	Intel(R) Turbo Boost Technology	Disabled
Advanced → Power & Performance → CPU P State Control	Energy Efficient Turbo	Disabled
Advanced → Power & Performance → CPU P State Control	Enhanced Intel Speed Select Technology	Enabled
Advanced → Power & Performance → CPU C State Control	C1E	Enabled

Notes:

- The BIOS setup menu locations above are to give an indication of each option only. The actual locations may differ from server to server.
- PCIe slot bifurcation configuration in the BIOS is required when using Intel® 200 GbE (E810-2C-QDA2) NICs.

Intel vCMTS Traffic Generator Server BIOS Settings

The following System BIOS settings are required on the vCMTS traffic generator server.

Bios Setup Menu	BIOS Option	Setting
Advanced → Processor Configuration	Intel(R) Hyper-Threading Tech	Enabled
Advanced → Integrated IO Configuration	Intel(R) VT for Directed I/O	Enabled
Advanced → Power & Performance	CPU Power and Performance Profile	Balanced Performance
Advanced → Power & Performance → CPU P State Control	Hardware P-states (HWP)	Disabled
Advanced → Power & Performance → CPU P State Control	Intel(R) Turbo Boost Technology	Disabled
Advanced → Power & Performance → CPU P State Control	Energy Efficient Turbo	Disabled
Advanced → Power & Performance → CPU P State Control	Enhanced Intel Speed Select Technology	Enabled

Notes:

- The BIOS setup menu locations above are to give an indication of each option only. The actual locations may differ from server to server.
- PCIe slot bifurcation configuration in the BIOS is required when using Intel® 200 GbE (E810-2C-QDA2) NICs.

Red Hat OCP Installation

Connect NIC Interfaces

If a switch is not used in the setup, the network interfaces of the vCMTS data plane and traffic generator servers will need to be directly connected to allow simulation traffic to flow. In this case, the corresponding links must be connected correctly to one another.

The corresponding links on both vCMTS data plane and traffic generator servers need to be connected directly to each other. The best method to ensure the correct links are cabled is to open a separate console on each server. Once a command line interface is accessed the following sequence should be followed on each server in parallel to identify the ports to be connected.

1. Identify the ports to be used for vCMTS data plane traffic, i.e. the XL710/E800 series NIC interfaces. This can be achieved by studying the list of interfaces on the OpenShift console under the "Nodes > Node Details > Network Interfaces" submenu or by running the following command on both the vCMTS data plane and traffic generator servers: `dmesg | egrep " ice | i40e " | grep renamed | awk '(print $5)' | sort -n`
2. For each port to be used for the vCMTS data plane traffic run the following command on the vCMTS data plane server: `ethtool --identify <interface_name> 10`
3. Run the following command on the vCMTS traffic generator server: `ethtool --identify <interface_name> 10`
4. A NIC interface on both the vCMTS data plane and traffic generator server will be flashing at this point. Using DAC cables, connect the flashing interface on the vCMTS data plane server to the flashing interface on the vCMTS traffic generator server.
5. Repeat steps 2 to 3 above until all the network interfaces have been cabled back-to-back correctly.

Cluster Installation

Due to the extensive documentation available from Red Hat, this guide does not provide step-by-step instructions for how to install the OpenShift cluster. It is recommended to install a cluster using the Red Hat OpenShift Container Platform Assisted Installer.

- Install the vCMTS reference data plane as a cluster: https://access.redhat.com/documentation/en-us/openshift_container_platform/4.14/html/installing/index .
- Installing the vCMTS reference data plane using the Assisted Installer: https://access.redhat.com/documentation/en-us/openshift_container_platform/4.14/html/installing/installing-on-premise-with-assisted-installer#installing-with-ai .

The subsequent steps in this guide related to OCP deployments assume a working cluster with the vCMTS data plane and traffic generator servers successfully deployed.

Accessing the Installed Cluster

As part of the installation process ensure that the kubeadmin password along with the kubeconfig file have been downloaded from the OpenShift console: https://access.redhat.com/documentation/en-us/openshift_container_platform/4.14/html/installing/installing-on-premise-with-assisted-installer#completing-the-installation_assisted-installer-installing.

In order to load the kubeconfig and additional files to the server it can be accessed by SSH using the private key pair for the public key used to create the installation images: `ssh -i <private_key_file> <ip address of vcmts traffic generator server>`

Configure Persistent Storage

In order for the OpenShift container image registry to function and store images it requires persistent storage to be configured. The corresponding documentation can be found from the following set of links:

- https://docs.openshift.com/container-platform/4.14/storage/persistent_storage/persistent-storage-nfs.html
- https://docs.openshift.com/container-platform/4.14/registry/configuring_registry_storage/configuring-registry-storage-baremetal.html
- https://docs.openshift.com/container-platform/4.14/registry/accessing-the-registry.html

Install the SR-IOV Network Operator

The vCMTS reference data plane requires the SR-IOV Network Operator for management of Network Interfaces required for vCMTS I/O. It should be installed using the OpenShift console by following the documentation available from: https://docs.openshift.com/container-platform/4.14/networking/hardware_networks/installing-sriov-operator.html.

Install the Performance Add On Operator

The vCMTS reference data plane requires the Performance Add On Operator for setup of hugepage memory and isolation of the CPU cores from the Linux scheduler for optimum performance. Performance Add On Operator for OpenShift is available from the following documentation: https://docs.openshift.com/container-platform/4.14/scalability_and_performance/cnf-low-latency-tuning.html.

Conclusion

The Intel® Verified Reference Configuration (VRC) for vCMTS successfully delivers next-generation broadband performance, achieving up to 290 Gbps of aggregate bi-directional throughput for 28 service groups at zero packet loss (ZPL) on Red Hat OpenShift* 4.14 and Dell PowerEdge* R660 servers with 4th Generation Intel® Xeon® Scalable processors.

Through streamlined integration of architectural enhancements, Intel® Ethernet 800 Series Adapters, and the Data Plane Development Kit (DPDK), this cloud-native solution reduces deployment complexity and enhances scalability. As a result, operators can confidently meet evolving subscriber demands with aproven, power-efficient, and future-ready broadband platform.

Learn More

Dell PowerEdge R660 Server

4th Gen Intel® Xeon® Scalable Processors

Intel QuickAssist Technology

100GbE Intel® Ethernet Network Adapter E810

Intel Verified Reference Configurations

Notices & Disclaimers

0801/MC/TM/TN/PDF ♻ Please Recycle 362490-001US