Dell EMC Ready Stack for Red Hat OpenShift Container Platform 4.6

Enabled by Dell EMC PowerEdge R640 and R740xd Servers; PowerSwitch Networking; PowerMax, PowerScale, PowerStore, Unity XT Storage

April 2021

H18217.6

Design Guide

Abstract

This design guide describes how to design and specify a Dell Technologies server and switch infrastructure for validated hardware configurations, facilitating deployment of Red Hat OpenShift Container Platform 4.6 following a Dell Technologies infrastructure deployment.

Dell Technologies Solutions

Copyright

Contents

Chapter 1 Introduction

This chapter presents the following topics:

Solution overview and key benefits

Ready Stack solution for OpenShift Container Platform 4.6

Dell EMC Ready Stack for Red Hat OpenShift Container Platform 4.6 is a flexible infrastructure that has been designed, optimized, and validated for an OpenShift Container Platform 4.6 on-premises bare-metal deployment. The deployment that this guide describes does not require a hypervisor.

The Dell EMC Ready Stack solution consists of the following documents:

- Dell EMC Ready Stack design guide (this document)
- Dell EMC Ready Stack deployment guide

  (Both documents are available at the Dell Technologies Info Hub for Containers.)

This Ready Stack solution provides:

- A detailed overview of validated OpenShift Container Platform hardware designs
- A scalable hardware platform of up to 210 compute nodes spread across seven racks
- Rapid implementation and time-to-value

The solution includes the following components:

- Red Hat OpenShift Container Platform 4.6 for application development and deployment
- Dell EMC PowerEdge R640 and R740xd servers for compute and storage
- Dell EMC PowerSwitch S5200 series switches for infrastructure network enablement
- Dell EMC PowerSwitch S3048 switch for out-of-band (OOB) management of the cluster

Note: While you can rely on Red Hat Enterprise Linux security and container technologies to prevent intrusions and protect your data, some security vulnerabilities might persist. For information about security vulnerabilities in OpenShift Container Platform, see *OCP Errata*. For a general listing of Red Hat vulnerabilities, see the *RH Security Home Page*.

OpenShift Container Platform and Kubernetes

OpenShift Container Platform 4.6 consists of many open-source components that have been carefully integrated to provide a consistently dependable platform on which you can develop and deploy scalable containerized applications. OpenShift Container Platform provides great flexibility for accommodating platform deployment preferences. For more information, see OpenShift Container Platform 4.6 Documentation.

At the heart of OpenShift Container Platform is Kubernetes container orchestration software. For more information, see What Kubernetes is.

Document purpose

Dell EMC Ready Stack for Red Hat OpenShift Container Platform is a proven design to help organizations accelerate their container deployments and cloud-native adoption. This guide provides information for building an on-premises infrastructure solution to host OpenShift Container Platform 4.6. The guide describes the Dell Technologies design decisions and configurations that enable solution architects to:

- Design and deploy a container platform solution.
- Extend or modify the design as necessary to meet customer requirements.

This guide includes:

- Container ecosystem design overview
- Network infrastructure design guidance
- Container and application storage design guidance
- Server requirements to support OpenShift Container Platform node roles
- Hardware platform configuration recommendations
- Rack-level design and power configuration considerations

A companion deployment guide provides information about automation-assisted deployment of the solution. This guide is available at the Dell Technologies Solutions Info Hub for Containers.

For information about the manual installation and deployment of Red Hat software products, see OpenShift Container Platform 4.6 Documentation.

Note: This guide may contain language from third-party content that is not under Dell's control and is not consistent with Dell's current guidelines for Dell's own content. When this content is updated by the relevant third parties, this guide will be revised accordingly.

Audience

This design guide is for system administrators and system architects. Some experience with Docker, Kubernetes, and OpenShift Container Platform technologies is recommended.

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on the solution and the solution documentation. Contact the Dell Technologies Solutions team by email or provide your comments by completing our documentation survey.

Author: Piyush Tandon

Contributors: John Terpstra, Umesh Sunnapu, Scott Powers, Aighne Kearney

Note: For additional information about this solution, see the Dell Technologies Solutions Info Hub for Containers.

Chapter 2 Technology and Deployment Process Overview

This chapter presents the following topics:

Introduction

OpenShift Container Platform 4.6 can host the development and runtime execution of containerized applications. The platform is continuing to mature and expand rapidly, providing you with access to the tools your team needs so that your business can grow. OpenShift Container Platform is based on Kubernetes, the de facto container automation and life cycle management platform for containerized workloads and services. Ready Stack for OpenShift Container Platform 4.6 includes Dell EMC hardware (servers, switches, and storage) to enable you to develop, validate, and deploy your containerized applications.

This chapter describes the OpenShift Container Platform architecture, infrastructure components, and requirements for a viable Ready Stack for OpenShift Container Platform 4.6 cluster, which can drive the core of modern telecommunications practices, multimedia operations, service provider infrastructure operations, the demands of the gaming industry, and financial transaction workloads.

OpenShift Container Platform

Overview

OpenShift Container Platform is an enterprise-grade declarative state machine that has been designed to automate application workload operations based on the upstream Kubernetes project. In a Kubernetes context, “declarative” means that developers can specify, in code, a configuration for an application or workload without knowing how that application is going to deployed. OpenShift Container Platform uses the enterprise-grade Kubernetes distribution, called the OpenShift Kubernetes Engine, to provide production-oriented container and workload automation. OpenShift Container Platform 4.6 is based on Kubernetes version 1.19, which includes native support for cluster snapshots, enabling cluster backup and recovery. On top of the Kubernetes Engine, OpenShift Container Platform provides administrators and developers with the tools they require to deploy and manage applications and services at scale, as shown in the following figure.

Note: OpenShift Container Platform is a certified Kubernetes distribution.

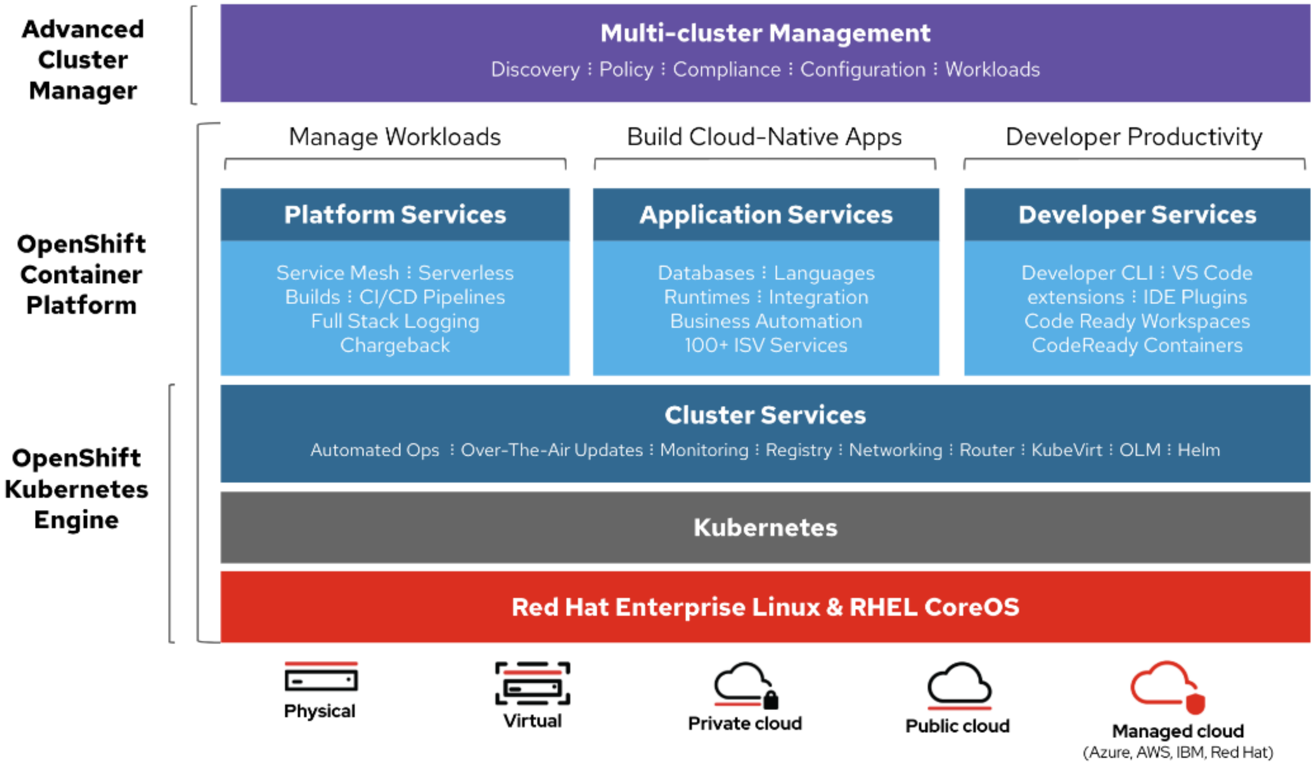


Figure 1. OpenShift Container Platform architecture

What Kubernetes is

Kubernetes provides an abstraction layer for application containers, deployments, and services and automates all container operations. Developers and administrators manipulate Kubernetes object declarations and abstractions to achieve the desired state of operations. Developers and administrators can specify the needs of an application in a declarative manner, and Kubernetes automatically deploys, terminates, or restarts containers to converge on this desired state.

What Kubernetes is not

Kubernetes is not just an “orchestration” platform for containers, which implies imperative, sequential actions. There is no imperative management of containers in Kubernetes. Rather, Kubernetes consists of independent control processes (state transition machines) that move the current state of the cluster towards the desired state. This mechanism has fundamental implications for how cluster operations, application middleware, and more can be managed automatically (see Cluster automation).

Upstream Kubernetes has some fundamental limitations in that it does not build or deploy applications, does not provide logging, monitoring, or alerting mechanisms, and is not a self-healing, self-managing system. As an open-source project, Kubernetes must support a variety of use cases and enable users to use a wide variety of projects that are compatible with Kubernetes.

Why OpenShift?

OpenShift Container Platform fills the gaps that Kubernetes leaves open:

- Platform-level services including building and packaging applications
- Integrated logging and monitoring solutions (Prometheus and Grafana)

- Integrated web console

OpenShift Container Platform is intended as a turnkey solution for production-grade environments. Among other benefits, OpenShift Container Platform:

- Eliminates the complexity of installing Kubernetes and of adding authentication, management, logging, security, and networking.
- Provides additional self-management capabilities that are not found in Kubernetes due to the tightly coupled toolchain: the default containers-first operating system (Red Hat CoreOS), a Kubernetes-first container runtime (CRI-O), and a rigorous testing and certification process for additional Red Hat and vendor middleware.

Kubernetes concepts

In Kubernetes, everything is an object. Every object has a current state, a desired state, and a specification of how a state transition can be achieved. This specification includes everything from applications, deployments, and services to machine configuration and management of specific hardware resources. When a Kubernetes object is created, the cluster uses the object to transition towards the desired state for the cluster. Custom Resource Definitions (CRDs) can be used to specify new resource types, which can then be used to create Custom Resources (CRs). Middleware (typically, operators) can use this extensible mechanism to create resource types that Kubernetes and other middleware with appropriate access can manage and use.

Cluster automation

The Operator Framework gives vendors the ability to manage the life cycle of the middleware they provide—for example, the Dell CSI Operator provides drivers for Dell EMC storage products. Operators attempt to encode the operational knowledge that is required for various stateful applications. Like Helm, an Operator can be used to configure and install middleware; however, depending on the complexity of the Operator, the Operator can fully automate an application’s life cycle management. Operators are application-specific, and therefore an Operator must be installed to manage each middleware application. In contrast, Helm is a universal package manager for Kubernetes.

The following figure shows the benefits that Operators can provide, depending on the complexity of the Operator:

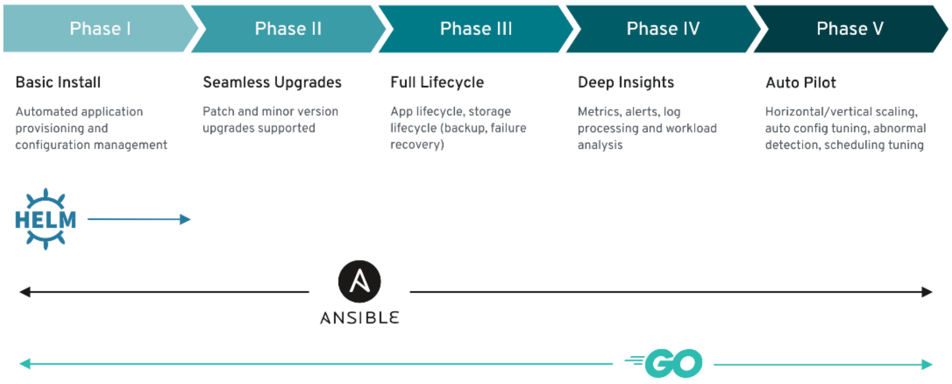

Phase I
Phase II
Phase III
Phase IV
Phase V
Basic Install
Automated application provisioning and configuration management
Seamless Upgrades
Patch and minor version upgrades supported
Full Lifecycle
App lifecycle, storage lifecycle (backup, failure recovery)
Deep Insights
Metrics, alerts, log processing and workload analysis
Auto Pilot
Horizontal/vertical scaling, auto config tuning, abnormal detection, scheduling tuning
HELM
ANSIBLE
GO

Figure 2. Operator maturity

Operators are designed to simplify Day-2 operations by automatically deploying, updating, and maintaining specific application deployments. This simplification is achieved through the creation of CRDs that are managed through a control loop that is embedded in the Operator. More complex Operators can be used to fully automate the life cycle management of various applications and middleware, scaling, and handling abnormalities gracefully.

Cloud-native infrastructure

A cloud-native infrastructure must accommodate a large, scalable mix of service-oriented applications and their dependent components. These applications and components are generally microservice-based. The The key to sustaining their operation is to have the right platform infrastructure and a sustainable management and control plane. This reference design helps you specify infrastructure requirements for building an on-premises OpenShift Container Platform 4.6 solution.

The following figure shows the solution design:

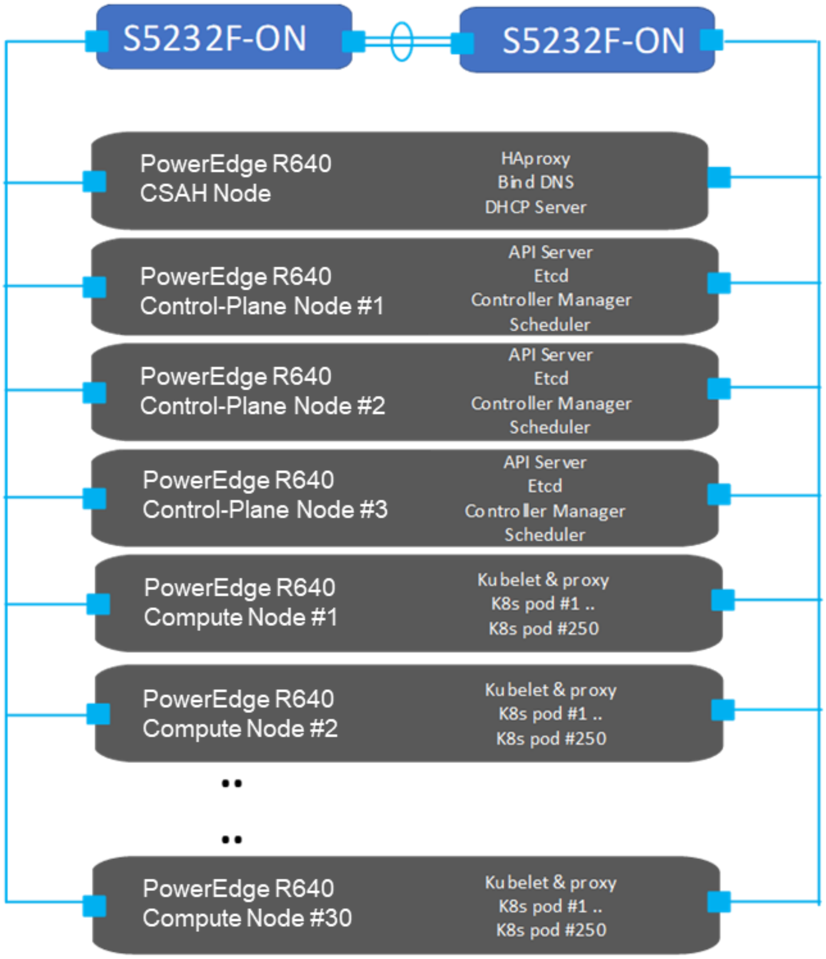


Figure 3. OpenShift Container Platform 4.6 cluster design

Terminology

This Ready Stack design recognizes four host types that make up every OpenShift Container Platform cluster: the bootstrap node, control-plane nodes, compute nodes, and storage nodes.

The deployment process also requires a node called the Cluster System Admin Host (CSAH). A description of the process is available in the *Ready Stack for Red Hat OpenShift Container Platform 4.6 Deployment Guide* at the Dell Technologies Solutions Info Hub for Containers.

Note: Red Hat official documentation does not refer to a CSAH node in the deployment process.

CSAH node

The CSAH node is not part of the cluster, but it is required for OpenShift cluster administration. Dell Technologies strongly discourages logging in to a control plane node to manage the cluster. The OpenShift CLI administration tools are deployed onto the control plane nodes, while the authentication tokens that are required to administer the OpenShift cluster are installed on the CSAH node only as part of the deployment process.

Note: Control-plane nodes are deployed using immutable infrastructure, further driving the preference for an administration host that is external to the cluster.

Bootstrap node (VM)

The CSAH node manages the operation and installation of the container ecosystem cluster. Installation of the cluster begins with the creation of a bootstrap VM on the CSAH node, which is used to install control-plane components on the controller nodes. Delete the bootstrap VM after the control plane is deployed. Dell Technologies recommends provisioning a dedicated host for administration of the OpenShift Container cluster. The initial minimum cluster can consist of three nodes running both the control plane and applications, or three control-plane nodes and at least two compute nodes. OpenShift Container Platform requires three control-plane nodes in both scenarios.

Basic node configuration

Node components are installed and run on every node within the cluster; that is, on controller nodes and compute nodes. The components are responsible for all node runtime operations. Key components consist of:

- **Kubelet:** An agent that runs on each node to perform declarations or actions that are provided to the cluster-API. *Kubelet* performs node service functions to ensure that running pods are compliant with *PodSpecs* and remain healthy. Kubelet does not manage containers or pods that were not created by Kubernetes.
- **Kube-proxy:** An instance of *kube-proxy* runs on every node of the cluster. It implements Kubernetes network services that run on the node. It also manages network connectivity and traffic route management based on host operating system packet filtering.
- **Container Runtime:** The chosen container runtime engine must be deployed on each node in a Kubernetes cluster. The Container Runtime Engine must comply with the Kubernetes Container Runtime Interface (CRI) specifications. OpenShift Container Platform defaults to the CRI-O container runtime and cannot be changed.

Control plane

Nodes that implement control plane infrastructure management are called controller nodes. Three controller nodes establish the control plane for the operation of an OpenShift cluster. The control plane operates outside the application container workloads and is responsible for ensuring the overall continued viability, health, availability, and integrity of the container ecosystem. Removing controller nodes is not allowed. OpenShift Container Platform also deploys additional control-plane infrastructure to manage OpenShift-specific cluster components.

The control plane provides the following functions:

- **API Server:** The API server exposes the Kubernetes control plane API for other platform services (such as a web console) to consume and has API endpoints to manage cluster resources.
- **Etcd:** Highly available and consistent key-value store used to maintain Kubernetes cluster data. The etcd daemon is run on each control plane node and requires at least two running daemons to achieve quorum. For production clusters, at least three control-plane nodes are therefore required, each running an etcd daemon.
- **Scheduler:** The Kubernetes scheduler assigns new pods to a node based on the resource requirements (for CPU, RAM, and GPU, for example), and the affinity and anti-affinity mechanisms.
- **Controller manager:** The controller managers run all controller processes. While each controller process is independent, the processes are run as a single process to reduce complexity. The controllers include the node, replication, endpoints, service, and token controllers.
- **OpenShift API server:** The OpenShift API server validates and configures the data for OpenShift resources such as projects, routes, and templates. The OpenShift API server is managed by the OpenShift API Server Operator.
- **OpenShift controller manager:** The OpenShift controller manager watches etcd for changes to OpenShift objects such as project, route, and template controller objects, and then uses the API to enforce the specified state. The OpenShift controller manager is managed by the OpenShift Controller Manager Operator.
- **OpenShift OAuth API server:** The OpenShift OAuth API server validates and configures the data to authenticate to OpenShift Container Platform, such as users, groups, and OAuth tokens. The OpenShift OAuth API server is managed by the Cluster Authentication Operator.
- **OpenShift OAuth server:** Users request tokens from the OpenShift OAuth server to authenticate themselves to the API. The OpenShift OAuth server is managed by the Cluster Authentication Operator.

Compute plane

In an OpenShift cluster, application containers are deployed to run on compute nodes, by default. The term “compute node” is arbitrary; nothing specific is required to run compute nodes and, therefore, applications can be run on control plane nodes. Cluster nodes advertise their resources and resource utilization so that the scheduler can allocate containers and pods to these nodes and maintain a reasonable workload distribution. The Kubelet service runs on each compute node. This service receives container deployment requests and ensures that the requests are instantiated and put into operation. The

Kubelet service also starts and stops container workloads and manages a service proxy that handles communication between pods that are running across compute nodes.

Logical constructs called MachineSets define compute node resources. MachineSets can be used to match requirements for a pod deployment to a matching compute node. OpenShift Container Platform supports defining multiple machine types, each of which defines a compute node target type.

Compute nodes can be added to or deleted from a cluster if doing so does not compromise the viability of the cluster. If the control plane nodes are not designated as schedulable, at least two viable compute nodes must always be operating. Further, enough compute platform resources must be available to sustain the overall cluster application container workload.

Storage nodes

Storage can be either provisioned from dedicated nodes or shared with compute services. Provisioning occurs on disk drives that are locally attached to servers that have been added to the cluster as compute nodes.

OpenShift Container Storage (OCS), which is deployed after the cluster deployment, simplifies and automates the deployment of storage for cloud-native container use. To integrate Ceph OCS storage into the container ecosystem infrastructure, administrators must provision appropriate storage nodes. It is also possible to use existing compute nodes if they meet OpenShift Container Storage hardware requirements.

You can initiate the deployment of OCS from the embedded OperatorHub when you are logged into OpenShift Container Platform as the cluster administrator. For more information, see OpenShift Container Platform 4.6 Documentation.

Deployment process

Dell Technologies has simplified the process of bootstrapping the OpenShift Container Platform 4.6 cluster. To use the simplified process, ensure that:

- The cluster is provisioned with network switches and servers.
- Network cabling is complete.
- Internet connectivity has been provided to the cluster. Internet connectivity is necessary to install OpenShift Container Platform 4.6.

The deployment procedure begins with initial switch provisioning. This step enables preparation and installation of the CSAH node, involving:

- Installing Red Hat Enterprise Linux 7
- Subscribing to the necessary repositories
- Creating an Ansible user account
- Cloning a GitHub Ansible playbook repository from the Dell ESG container repository
- Running an Ansible playbook to initiate the installation process

Dell Technologies has generated Ansible playbooks that fully prepare the CSAH node. Before the installation of the OpenShift Container Platform 4.6 cluster begins, the Ansible playbook sets up a PXE server, DHCP server, DNS server, HAProxy, and HTTP server. The playbook also creates ignition files to drive installation of the bootstrap, control plane, and compute nodes. It also starts the bootstrap VM to initialize control plane components. The playbook presents a list of node types that must be deployed in top-down order.

Note: For enterprise sites, consider deploying appropriately hardened DHCP and DNS servers. Similarly, consider using resilient multiple-node HAProxy configuration. The Ansible playbook for this design deploys a single HAProxy instance. This guide provides CSAH Ansible playbooks for reference only at the implementation stage.

The Ansible playbook creates an `install-config.yaml` file that is used to control deployment of the ***bootstrap*** node. For more information, see the *Dell EMC Ready Stack: Red Hat OpenShift Container Platform 4.6 Deployment Guide* at the Dell Technologies Solutions Info Hub for Containers. An ignition configuration control file starts the bootstrap node, as shown in the following figure:

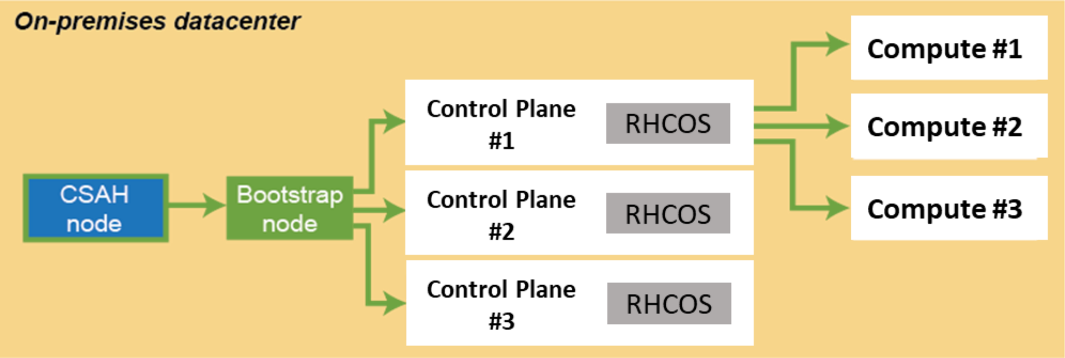


Figure 4. Installation workflow: Creating the bootstrap, control-plane, and compute nodes

Note: An installation that is driven by ignition configuration generates security certificates that expire after 24 hours. You must install the cluster before the certificates expire, and the cluster must operate in a viable (nondegraded) state so that the first certificate rotation can be completed.

The cluster bootstrapping process consists of the following phases:

1. After startup, the bootstrap VM creates the resources that are required to start the control-plane nodes. Do not interrupt this process.
2. The control-plane nodes pull resource information from the bootstrap VM to bring them up into a viable state. This resource information is used to form the `etcd` control plane cluster.
3. The bootstrap VM instantiates a temporary Kubernetes control plane that is under `etcd` control.
4. A temporary control plane loads the application workload control plane to the control-plane nodes.
5. The temporary control plane is shut down, handing control over to the now viable control-plane nodes.

6. OpenShift Container Platform components are pulled into the control of the control-plane nodes.
7. The bootstrap VM is shut down.

   The control-plane nodes now drive creation and instantiation of the compute nodes.
8. The control plane adds operator-based services to complete the deployment of the OpenShift Container Platform ecosystem.

The cluster is now viable and can be placed into service in readiness for Day-2 operations. You can expand the cluster by adding compute nodes.

Infrastructure requirements

Basic guidance

The following table provides basic cluster infrastructure guidance. For detailed configuration information, see Cluster Hardware Design. Administrators can build a container cluster to be deployed quickly and reliably when each node is within the validated design guidelines.

Table 1. Hardware infrastructure for OpenShift Container Platform 4.6 cluster deployment

Type	Description	Count	Notes
CSAH node	Dell EMC PowerEdge R640 server	1	Creates a bootstrap VM. CSAH runs a single instance of HAProxy. For enterprise high availability (HA) deployment of OpenShift Container Platform 4.6, Dell Technologies recommends using a commercially supported L4 load-balancer or proxy service or system. Options include commercial HAProxy, Nginx, and F5.
Controller nodes	Dell EMC PowerEdge R640 server	3	Deployed using the bootstrap node.
Compute nodes	Dell EMC PowerEdge R640 or R740xd server	Minimum 2,* maximum 30 per rack	No compute nodes are required for a three-node cluster. A standard deployment requires a minimum of two compute nodes (and three controller nodes). To expand a three-node cluster, you must add two compute nodes at the same time. After the cluster is operational, you can add more compute nodes to the cluster through the Cluster Management Service.
Data switches	Either of the following switches: • Dell EMC PowerSwitch S5248-ON • Dell EMC PowerSwitch S5232-ON	2 per rack	Autoconfigured at installation time. **Note:** • HA network configuration requires two data path switches per rack. • Multirack clusters require network topology planning. Leaf-spine network switch configuration may be necessary.
iDRAC network	Dell EMC PowerSwitch S3048-ON	1 per rack	Used for OOB management.
Rack	Selected according to site standards	1–3 racks	For multirack configurations, consult your Dell Technologies or Red Hat representative regarding custom engineering design.

*A three-node cluster does not require any compute nodes. To expand a three-node cluster with additional compute machines, you must first expand the cluster to a five-node cluster using two additional compute nodes.

Minimum viable solution requirements

Installing OpenShift Container Platform requires, at a minimum, the following nodes:

- One CSAH node, which is used to run the bootstrap VM. The CSAH node is used later to manage the cluster while the cluster is in production use.
- Three nodes running both the control plane and data plane, enabling customers to develop OpenShift 4.6 POCs using only four nodes. The cluster can be expanded with additional compute nodes as needed. However, an initial expansion beyond three nodes requires two compute nodes. A four-node cluster (three controllers, one compute) is not supported. The minimum viable solution options are a three-node cluster (three control-compute nodes) or a five-node cluster (three controller nodes, two compute nodes) plus the CSAH node for cluster administration with either option.

HA of the key services that make up the OpenShift Container Platform cluster is necessary to ensure run-time integrity. Redundancy of physical nodes for each cluster node type is an important aspect of HA for the bare-metal cluster.

In this design guide, HA includes the provisioning of at least two network interface controllers (NICs) and two network switches that are configured to provide redundant pathing. The redundant pathing provides for network continuity if a NIC or a network switch fails.

OpenShift Container Platform 4.6 must use Red Hat Enterprise Linux CoreOS (RHCOS) for the control-plane nodes and can use either RHCOS or Red Hat Enterprise Linux 7.6 for compute nodes. Using Red Hat Enterprise Linux 7 on the compute nodes is now deprecated, and the ability to use Red Hat Enterprise Linux 7 compute nodes in OpenShift will be removed in a future release of OpenShift. The bootstrap and control-plane nodes must use RHCOS as their operating system. Each of these nodes must be immutable.

The following table shows the minimum resource requirements:

Table 2. Minimum resource requirements for OpenShift Container Platform 4.6 nodes

Node type	Operating system	Minimum CPU cores	RAM	Storage
CSAH	Red Hat Enterprise Linux 7.6+	4	32 GB	200 GB
Bootstrap	RHCOS 4.6	4	16 GB	120 GB
Controller	RHCOS 4.6	4	16 GB	120 GB
Compute	RHCOS 4.6 or Red Hat Enterprise Linux 7.6 (deprecated)	2	8 GB	120 GB

Network connectivity requirements

The RHCOS nodes must fetch ignition files from the Machine Config server. This operation uses an `initramfs-based-node` startup for the initial network configuration. The startup requires a DHCP server to provide a network connection giving access to the ignition files for that node. Subsequent operations can use static IP addresses.

Chapter 3 Networking Infrastructure and Configuration

This chapter presents the following topics:

Introduction

The components and operations that make up the container ecosystem each require network connectivity, plus the ability to communicate with all the others and respond to incoming network requests. This Ready Stack for OpenShift Container Platform 4.6 reference design uses Dell EMC PowerSwitch networking infrastructure.

OpenShift network operations

Operating components

Applications run on compute nodes. Each compute node is equipped with resources such as CPU cores, memory, storage, NICs, and add-in host adapters (GPUs, SmartNICs, FPGAs, and so on). Kubernetes provides a mechanism to enable orchestration of network resources through the Container Network Interface (CNI) API.

The CNI API uses the Multus CNI plug-in to enable attachment of multiple adapter interfaces on each pod. Container Resource Definitions (CRD) objects are responsible for configuring Multus CNI plug-ins.

Container communications

A pod, a basic unit of application deployment, consists of one or more containers that are deployed together on the same compute node. A pod shares the compute node network infrastructure with the other network resources that make up the cluster. As service demand expands, more identical pods are often deployed to the same or other compute nodes.

Networking is critical to the operation of an OpenShift Container cluster. Four basic network communication flows arise within every cluster:

- Container-to-container connections (also called highly coupled communication)
- Pod communication over the local host network (127.0.0.1)
- Pod-to-pod connections, as described in this design guide
- Pod-to-service and ingress-to-service connections, which are handled by services

Containers that communicate within their pod use the local host network address. Containers that communicate with any external pod originate their traffic based on the IP address of the pod.

Application containers use shared storage volumes (configured as part of the pod resource) that are mounted as part of the shared storage for each pod. Network traffic that might be associated with nonlocal storage must be able to route across node network infrastructure.

Services networking

Services are used to abstract access to Kubernetes pods. Every node in a Kubernetes cluster runs a kube-proxy and is responsible for implementing virtual IP (VIP) for service. Kubernetes supports two primary modes of finding (or resolving) a service:

- **Using environment variables**—This method requires a reboot of the pods when the IP address of the service changes.

- **Using DNS**—OpenShift Container Platform 4.6 uses CoreDNS to resolve service IP addresses.

Some part of the application (for example, front-ends) might want to expose a service outside the application. If the service uses HTTP, HTTPS, or any other TLS-encrypted protocol, use an ingress controller; otherwise, use a load balancer, external service IP address, or node port.

A node port exposes the service on a static port on the node IP address. A service with `NodePort-type` as a resource exposes it on a specific port on all nodes in the cluster. Ensure that external IP addresses are routed to the nodes.

Ingress controller

OpenShift Container Platform uses an ingress controller to provide external access. The ingress controller defaults to running on two compute nodes, but it can be scaled up as required. Dell Technologies recommends creating a wildcard DNS entry and then setting up an ingress controller. This method enables you to work only within the context of an ingress controller. An ingress controller accepts external HTTP, HTTPS, and TLS requests using SNI and then proxies them based on the routes that are provisioned.

You can expose a service by creating a route and using the cluster IP. Cluster IP routes are created in the OpenShift Container Platform project, and a set of routes is admitted into ingress controllers.

You can perform sharding (horizontal partitioning of data) on route labels or name spaces. Sharding ingress controllers enables you to:

- Load-balance the incoming traffic.
- Segregate the required traffic to a single ingress controller.

Networking operators

The following operators are available for network administration:

- **Cluster Network Operator (CNO)**—Deploys the OpenShift SDN plug-in during cluster installation and manages kube-proxy running on each node
- **DNS operator**—Deploys and manages CoreDNS and instructs pods to use the CoreDNS IP address for name resolution
- **Ingress operator**—Enables external access to OpenShift Cluster Platform cluster services and deploys and manages one or more HAProxy-based ingress controllers to handle routing

Container Networking Interface

The Container Networking Interface (CNI) specification serves to make the networking layer of containerized applications pluggable and extensible across container runtimes. The CNI specification is used in both upstream Kubernetes and OpenShift in the pod network. This use is not implemented by Kubernetes, but by various CNI plug-ins. The most commonly used plug-ins are:

- **Multus:** CNI plug-in that supports the multinetwork function in Kubernetes. Typically, Kubernetes pods have only one networking interface, but the use of Multus means that pods can be configured to support multiple interfaces. Multus acts as a “meta plug-in,” a plug-in which calls other CNI plug-ins. In addition to other CNI plug-ins, Multus supports SR-IOV and DPDK workloads.

- **DANM:** Developed by Nokia, DANM is a CNI plug-in for Telco-oriented workloads. DANM supports the provisioning of advanced IPVLAN interfaces, acts like Multus in that it is also a meta plug-in, can control VxLAN and VLAN interfaces for all Kubernetes hosts, and more. The DANM CNI plug-in creates a network management API to give administrators greater control of the physical networking stack through the standard Kubernetes API.

OpenShift SDN

OpenShift SDN creates an overlay network that is based on Open Virtual Switch (OVS). The overlay network enables communication between pods across the cluster. OVS operates in one of the following modes:

- Network policy mode (the default), which allows custom isolation policies
- Multitenant mode, which provides project-level isolation for pods and services
- Subnet mode, which provides a flat network

OpenShift Container Platform 4.6 also supports using Open Virtual Network (OVN)-Kubernetes as the CNI network provider. OVN-Kubernetes will become the default CNI network provider in a future release of OpenShift. OpenShift Container Platform 4.6 supports additional SDN orchestration and management plugins that comply with the CNI specification. See Use cases for examples.

Service Mesh

Distributed microservices work together to make up an application. Service Mesh provides a uniform method to connect, manage, and observe microservices-based applications. The Red Hat OpenShift Service Mesh implementation is based on Istio, an open-source project. OpenShift Service Mesh is not installed automatically as part of a default installation; instead, the user must install Service Mesh by using operators from the OperatorHub.

Service Mesh has key functional components that belong to either the data plane or the control plane:

- **Envoy proxy**—Intercepts all traffic for all services in Service Mesh. Envoy proxy is deployed as a sidecar.
- **Mixer**—Enforces access control and collects telemetry data.
- **Pilot**—Provides service discovery for the envoy sidecars.
- **Citadel**—Provides strong service-to-service and end-user authentication with integrated identity and credential management.

Users define the granularity of Service Mesh deployment, enabling them to meet their specific deployment and application needs. Service Mesh can be employed at the cluster level or at the project level. For more information, see the OpenShift Service Mesh documentation.

SR-IOV and multiple networks

Single Root Input/Output Virtualization (SR-IOV) enables the creation of multiple virtual functions from one physical function for a PCIe device (such as NICs). In the network, this capability can be used to create many virtual functions from a single NIC, where each virtual function can be attached to a pod. Latency is reduced because of the reduced I/O overhead from the software switching layer. Also, SR-IOV can be used to configure multiple networks by attaching multiple virtual functions with different networks to a single pod. SR-IOV can be configured in OpenShift by using the SR-IOV Operator, which can create virtual functions and provision additional networks. For more information, see the *Ready Stack for OpenShift Container Platform 4.6 Deployment Guide* at the Dell Technologies Solutions Info Hub for Containers.

Multinetwork support

OpenShift Container Platform 4.6 also supports software-defined multiple networks. OpenShift Container Platform comes with a default network. The cluster administrator defines additional networks using the Multus CNI plug-in and then chains the plug-ins. These additional networks are useful for increasing the networking capacity of the pods and meeting traffic separation requirements.

The following CNI plug-ins are available for creating additional networks:

- **Bridge**—The same host pods can communicate over a bridge-based additional network.
- **Host-device**—Pods can access the host's physical Ethernet network device.
- **Macvlan**—Pods attached to a macvlan-based additional network have a unique MAC address and communicate using a physical network interface.
- **Ipvlan**—Pods communicate over an ipvlan-based additional network.

Leaf-switch considerations

When pods are provisioned with additional network interfaces that are based on macvlan or ipvlan, corresponding leaf-switch ports must match the VLAN configuration of the host. Failure to properly configure them results in a loss of traffic.

Physical network design

Design principles

Dell EMC networking products are designed for ease of use and to enable resilient network creation. OpenShift Container Platform 4.6 introduces various advanced networking features to enable containers for high performance and monitoring. The Dell EMC-recommended design applies the following principles:

- Meet network capacity and the segregation requirements of the container pod.
- Configure dual-homing of the OpenShift Container Platform node to two Virtual Link Trunked (VLT) switches.
- Create a scalable and resilient network fabric to increase cluster size.
- Provide the ability to monitor and trace container communications.

Container network capacity and segregation

Container networking takes advantage of the high speed (25/100 GbE) network interfaces of the Dell Technologies server portfolio. Also, to meet network capacity requirements, pods can attach to more networks using available CNI plug-ins.

Additional networks are useful when network traffic isolation is required. Networking applications such as Container Network Functions (CNFs) have control traffic and data traffic. These different traffic types have different processing, security, and performance requirements.

Pods can be attached to the SR-IOV virtual function (VF) interface on the host system for traffic isolation and to increase I/O performance.

Dual-homing

Dual-homing means that each node that makes up the OpenShift cluster has at least two NICs, each connected to at least two switches. The switches require VLT connections so that they operate together as a single unit of connectivity to provide redundant data paths for all network traffic. The NICs at each node and the ports they connect to on each of the switches can be aggregated using bonding to assure HA operation.

Network fabric

A nonblocking fabric is required to meet the needs of the microservices data traffic. Dell Technologies recommends deploying a leaf-spine network.

Monitoring and tracing

OpenShift Container Platform 4.6 supports Service Mesh. Users can monitor container traffic by using Kiali and perform end-to-end tracing of applications by using Jaeger.

Resilient networking

Each server that has many NIC options in the rack is connected to:

- Two leaf switches with a network interface of choice: 10 GbE, 25 GbE, or100 GbE
- A management switch (typically, 1 GbE) for iDRAC connectivity

Our network design employs a VLT connection between the two leaf switches. In a VLT environment, all paths are active; therefore, it is possible to achieve high throughput while still protecting against hardware failures.

VLT technology allows a server to uplink multiple physical trunks into more than one Dell EMC PowerSwitch switch by treating the uplinks as one logical trunk. A VLT-connected pair of switches acts as a single switch to a connecting server. Both links from the bridge network can forward and receive traffic. VLT provides a replacement for Spanning Tree Protocol (STP)-based networks by providing both redundancy and full bandwidth utilization using multiple active paths.

The major benefits of VLT technology are:

- Dual control plane for highly available, resilient network services
- Full utilization of the active link aggregation (LAG) interfaces
- Active/active design for seamless operations during maintenance events

The VLTi configuration in this design uses two 100 GbE ports between each ToR switch. The remainder of the 100 GbE ports can be used for high-speed connectivity to spine switches or directly to the data center core network infrastructure.

Scale out with leaf-spine fabric

You can scale container solutions by adding multiple compute nodes and storage nodes. A container cluster can have multiple racks of servers. To create a nonblocking fabric that meets the needs of the microservices data traffic, we used a leaf-spine network.

Leaf-spine overview

Layer 2 and Layer 3 leaf-spine topologies employ the following concepts:

- Each leaf switch connects to every spine switch in the topology.
- Servers, storage arrays, edge routers, and similar devices connect to leaf switches, but never to spines.

Our design used dual leaf switches at the top of each rack. We employed VLT in the spine layer, which allows all connections to be active while also providing fault tolerance. As administrators add racks to the data center, leaf switches are added to each new rack.

The total number of leaf-spine connections is equal to the number of leaf switches multiplied by the number of spine switches. Administrators can increase the bandwidth of the fabric by adding connections between leaves and spines if the spine layer has capacity for the additional connections.

Layer 3 leaf-spine network

In a Layer 3 leaf-spine network, traffic is routed between leaves and spines. The Layer 3-Layer 2 boundary is at the leaf switches. Spine switches are never connected to each other in a Layer 3 topology. Equal cost multipath routing (ECMP) is used to load-balance traffic across the Layer 3 network. Connections within racks from hosts to leaf switches are Layer 2. Connections to external networks are made from a pair of edge or border leaves, as shown in the following figure:

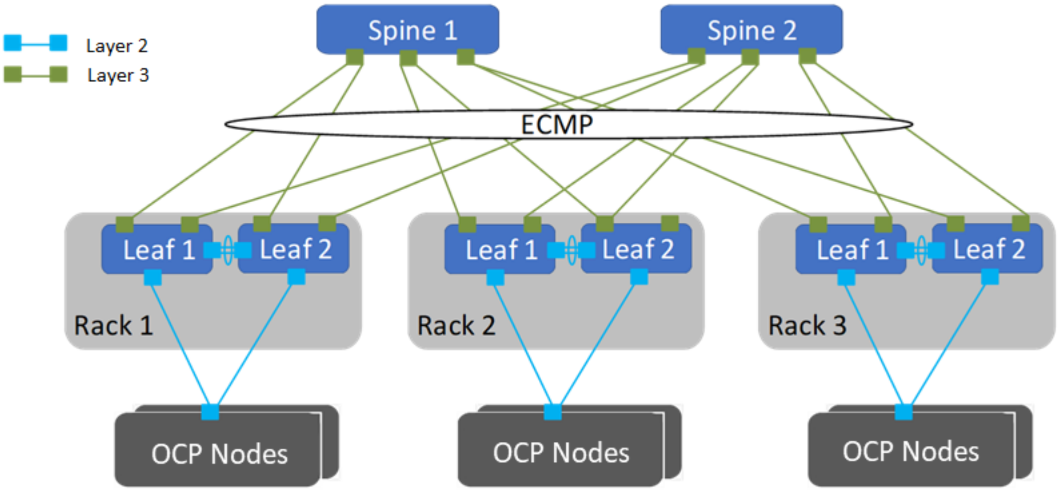


Figure 5. Leaf-spine network configuration

Dell EMC PowerSwitch configuration

Dell's high-capacity network switches are cost-effective and easy to deploy. The switches provide a clear path to a software-defined data center and offer:

- High density for 25, 40, 50, or 100 GbE deployments in top-of-rack (ToR), middle-of-row, and end-of-row deployments
- A choice of S5048F-ON, S5148F-ON, S5212F-ON, S5224F-ON, S5248F-ON, S5296F-ON, S5232F-ON 25 GbE and 100 GbE switches, and the S6100-ON 10 GbE, 25 GbE, 40 GbE, 50 GbE, or 100 GbE modular switch

- S6100-ON modules that include: 16-port 40 GbE QSFP+; eight-port 100 GbE QSFP28; combo module with four 100 GbE CXP ports and four 100 GbE QSFP28 ports

We used Dell EMC Network Operating System OS10 for our solution design. OS10 allows multilayered disaggregation of network functions that are layered on an open-source Linux-based operating system. The following section describes a high-level configuration of the PowerSwitch switches that are used for an OpenShift Container Platform deployment at various scales.

Configuring VLT

At a high level, the VLT configuration consists of the following steps:

1. Enable Spanning Tree, the default, on the VLT peer switches. Spanning Tree is recommended to prevent loops in a VLT domain. RPVST+ (the default) and RSTP modes are supported on VLT ports.
2. Create a VLT domain and configure the VLT interconnect (VLTi).
3. Configure the VLT Priority, VLT MAC Address, and VLT Backup Link.
4. Configure the LAG for the connected device.
5. Verify and monitor the status of VLT by using OS10 show commands.

Installation with Ansible

Dell EMC Networking modules are supported in Ansible core from Ansible 2.3 on. You can use these modules to manage and automate Dell EMC switches running OS10. The modules are run in local connection mode using CLI and SSH transport.

For an example of CLOS fabric deployment based on the Border Gateway Protocol (BGP), see Provision CLOS fabric using Dell EMC Networking Ansible modules example.

High availability and load balancing

This solution uses the following HA features:

- **Red Hat OpenShift Container Platform 4.6**—Multiple control-plane nodes and infrastructure nodes
- **Dell EMC cloud-native infrastructure**—PowerEdge servers with dual NICs
- **Dell PowerSwitch**—Spine-leaf fabric with VLT

Always make external traffic paths highly available to create a complete solution. The cluster administrator can use an external L4 load-balancer in a highly available manner or deploy HAProxy in resilient mode. Deploying HAProxy requires one additional server. As shown in the following figure, the components of a highly available load-balancer design using HAProxy are:

- Keepalived and HAProxy running on CSAH—Configure VIP on a suitable network interface.
- Keepalived and HAProxy running on an additional server—Configure VIP on a suitable network interface.

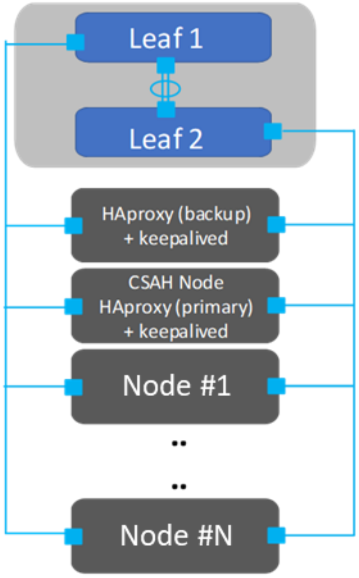


Figure 6. Highly available load-balancing

Keepalived

`Keepalived` is an open-source project that implements routing software using the Virtual Router Redundancy Protocol (VRRP). VRRP allows a switchover to a backup server if the primary server fails. This switchover is achieved by using VIP. To configure `keepalived` on both servers:

- Configure one server as MASTER (the primary server) with high priority.
- Configure the other server as BACKUP with lower priority.

Chapter 4 Storage Overview

This chapter presents the following topics:

OpenShift Container Platform storage

Introduction

Stateful applications create a demand for persistent storage. All storage within OpenShift Container Platform 4.6 is managed separately from compute resources and from all networking and connectivity infrastructure facilities. The CSI API is designed to abstract storage use and enable storage portability.

This solution applies the following Kubernetes storage concepts:

- **Persistent volume (PV)**—The physical LUN or file share on the storage array. PVs are internal objects against which persistent volume claims are created. PVs are unrelated to pods and pod storage life cycles.
- **Persistent volume claim (PVC)**—An entitlement that the user creates for the specific PV.
- **Storage class**—A logical construct defining storage allocation for a given group of users.
- **CSI driver**—The software that orchestrates persistent volume provisioning and deprovisioning on the storage array.

These resources are logical constructs that are used within the Kubernetes container infrastructure to maintain storage for all the components of the container ecosystem that depend on storage. Developers and operators can deploy applications and provision or deprovision persistent storage without having any specific technical knowledge of the underlying storage technology.

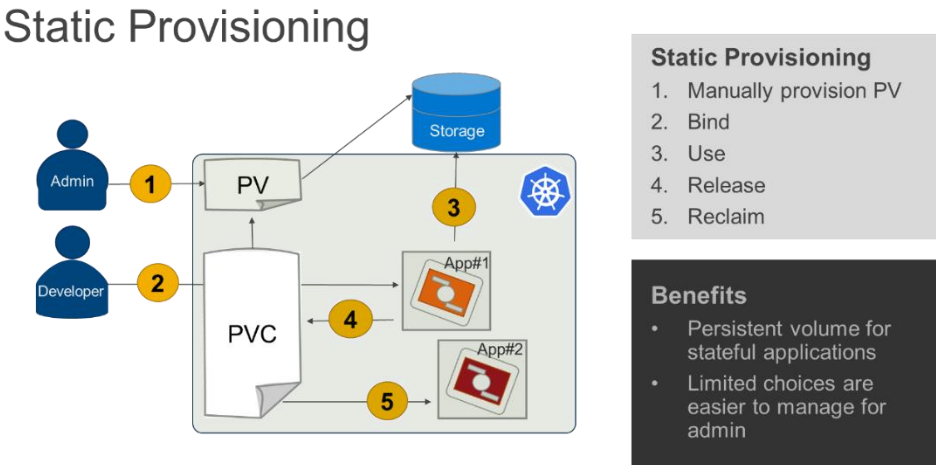
The OpenShift Container Platform administrator is responsible for provisioning storage classes and making them available to the cluster's tenants.

Storage using PVCs is consumed or used in two ways: statically or dynamically. Static storage can be attached to one or more pods by static assignment of a PV to a PVC and then to a specific pod or pods.

Static storage provisioning

With static persistent storage provisioning, an administrator pre-provisions PVs for Kubernetes tenants. When a user makes a persistent storage request by creating a PVC, Kubernetes finds the closest matching available PV. Static provisioning is not the most efficient method for using storage, but it might be preferred when it is necessary to restrict users from PV provisioning.

The following figure illustrates the static storage provisioning workflow in this solution:



Static Provisioning

1. Manually provision PV
2. Bind
3. Use
4. Release
5. Reclaim

Benefits

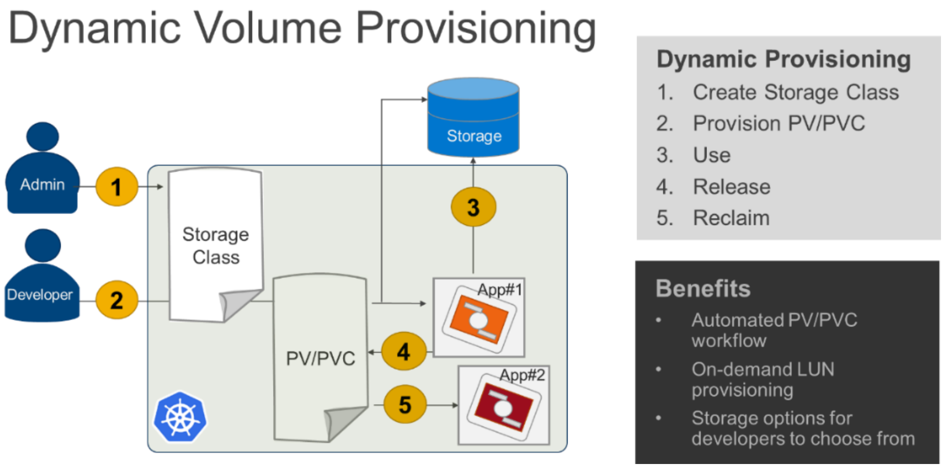
- Persistent volume for stateful applications
- Limited choices are easier to manage for admin

Figure 7. **Static storage provisioning workflow**

Dynamic persistent storage provisioning

Dynamic persistent storage provisioning, the most flexible provisioning method, enables Kubernetes users to secure PV provisioning on demand. Dynamic provisioning has fully automated LUN export provisioning.

The following figure shows the dynamic storage provisioning workflow in this solution:



Dynamic Provisioning

1. Create Storage Class
2. Provision PV/PVC
3. Use
4. Release
5. Reclaim

Benefits

- Automated PV/PVC workflow
- On-demand LUN provisioning
- Storage options for developers to choose from

Figure 8. **Dynamic storage provisioning workflow and benefits**

After a PV is bound to a PVC, that PV cannot be bound to another PVC. This restriction binds the PV to a single namespace, that of the binding project. A PV that has been created for dynamic use is a storage class object that functions as, and is automatically consumed as, a cluster resource.

PV types

OpenShift Container Platform natively supports the following PV types:

- AWS Elastic Block Store (EBS)
- Azure Disk
- Azure File
- Cinder
- Fibre Channel (FC) (can only be assigned and attached to a node)
- GCE Persistent Disk
- HostPath (local disk)
- iSCSI (generic)
- Local volume
- NFS (generic)
- Red Hat OpenShift Container Storage
- VMware vSphere

The CSI API extends the storage types that can be used within an OpenShift Container Platform solution.

PV capacity

Each PV has a predetermined storage capacity that is set in its `capacity` parameter. The storage capacity can be set or requested by a pod that is launched within the container platform. Expect the choice of control parameters to expand as the CSI API is extended and as it matures.

PV access modes

A resource provider can determine how the PV is created and can set the storage control parameters. Access mode support is specific to the type of storage volume that is provisioned as a PV. Provider capabilities determine the PV's access modes, while the capabilities of each PV determine the modes which that volume supports. For example, NFS can support multiple read/write clients, but a specific NFS PV might be configured as read-only.

Pod claims are matched to volumes with compatible access modes based on two matching criteria: access modes and size. A pod claim's access modes represent a request.

Static persistent storage

The use of generic NFS or generic iSCSI is functional and stable. However, NFS and iSCSI do not contain a mechanism to provide service continuity if access to the storage subsystem fails. Generic NFS and iSCSI do not provide the advanced storage protection support that is available using CSI drivers. As described in the following table, the Dell Technologies Storage Engineering team validated the functionality and capability of suitable storage drivers:

Table 3. Generic storage capabilities

Storage type	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
HostPath (local disk)	Yes	N/A	N/A
iSCSI (generic)	Yes	Yes	N/A
NFS (generic)	Yes	Yes	Yes

Container Storage Interface (CSI) external storage

Introduction

OpenShift Container Platform 4.2 introduced support for the CSI operator-framework-driven API. This CSI API manages the control plane (that is, it runs on the control-plane nodes) to orchestrate and manage configuration and tear-down of data-path storage operations. Storage driver plug-in support was available in earlier Kubernetes releases, but it required the integration of volume plug-ins into the core Kubernetes codebase. Kubernetes version 1.19 is integrated into OpenShift Container Platform 4.6.

Why CSI?

The CSI was introduced to GA in Kubernetes v1.13. CSI replaced the volume plug-in system. Volume plug-ins were built “in-tree,” that is, as part of the Kubernetes source code; therefore, changes or fixes to various volume plug-ins provided by storage vendors had to be made in lockstep with the core Kubernetes release schedule. The CSI specification aims to standardize the exposure of block and file storage systems to workloads running on container orchestration systems such as Kubernetes. Kubernetes can now be readily extended to support any storage solution with CSI drivers that the vendor provides. Vendors can manage the life cycle of their drivers directly, using an Operator, without waiting until the next core Kubernetes release.

CSI architecture

Drivers are typically shipped as container images. These images are not platform-aware, and therefore additional components are required to enable interaction between OpenShift Container Platform and the driver image. An external CSI controller running on infrastructure nodes has three containers: attacher, provisioner, and driver container. The attacher and provisioner containers serve as translators; they map OpenShift Container Platform calls to the corresponding calls to the CSI driver. No other communication to the CSI driver is allowed. On each compute node, a CSI Driver Daemon set is created containing the CSI driver, and a CSI Registrar. The Registrar registers the driver with the openshift-node service, which then directly connects to the driver. The following figure shows this architecture:

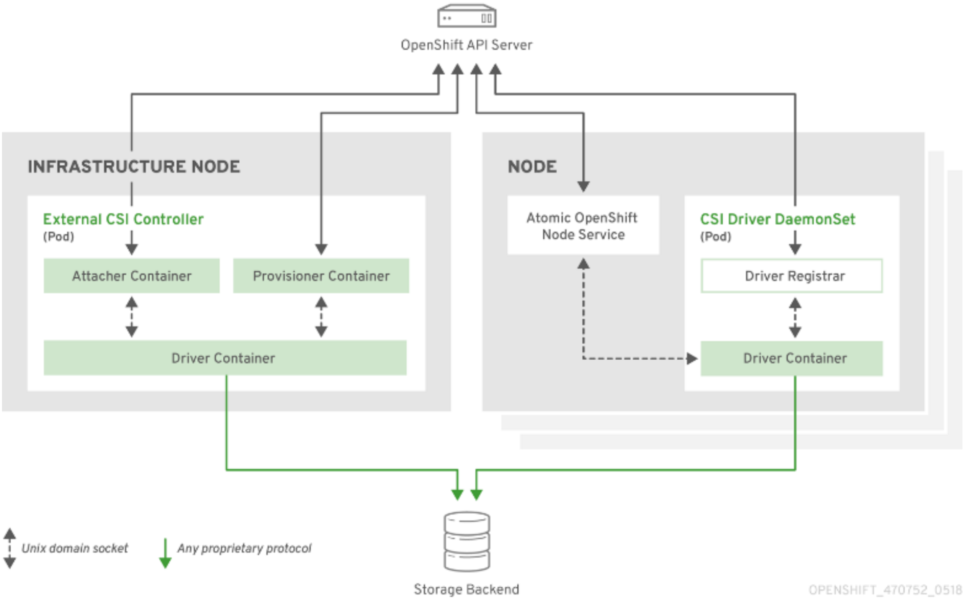


Figure 9. CSI architecture

CSI volume snapshots

Support for snapshots of CSI volumes was added in Kubernetes v1.19 and is available in OpenShift Container Platform 4.6 as a Tech Preview feature. OpenShift provides the CSI Snapshot Controller Operator, which manages snapshot objects. An external snapshot sidecar container must be implemented in the CSI driver to enable snapshot functionality. All Dell Storage CSI drivers support snapshots.

Storage feature support with Dell Technologies products

The following table provides an overview of Dell Technologies storage platforms with their corresponding CSI and protocol support. These capabilities reflect what has been implemented in the CSI drivers that are intended for use with OpenShift Container Platform 4.6.

Table 4. Dell Technologies CSI storage products and capabilities

Storage capability	PowerMax	PowerFlex operating system	Unity	PowerScale	PowerStore
Static provisioning	Yes	Yes	Yes	Yes	Yes
Dynamic provisioning	Yes	Yes	Yes	Yes	Yes
Binding	Yes	Yes	Yes	Yes	Yes
Retain Reclaiming	Yes	Yes	Yes	Yes	Yes
Delete Reclaiming	Yes	Yes	Yes	Yes	Yes
Create Snapshot Volume	No	Yes	Yes	Yes	Yes

Storage capability	PowerMax	PowerFlex operating system	Unity	PowerScale	PowerStore
Create Volume from Snapshot	No	Yes	Yes	Yes	Yes
Delete Snapshot	No	Yes	Yes	Yes	Yes
Access Mode	ReadWrite Once	ReadWrite Once	ReadWrite Once	ReadWrite Many	ReadWrite Once
FC	Yes	N/a	Yes	N/a	Yes
iSCSI	Yes	N/a	Yes	N/a	Yes
NFS	N/a	N/a	No	Yes	Yes
Other protocols	N/a	ScaleIO protocol	N/a	N/a	N/a
Red Hat Enterprise Linux node	Yes	Yes	Yes	Yes	Yes
RHCOS node	Yes	No	Yes	Yes	Yes

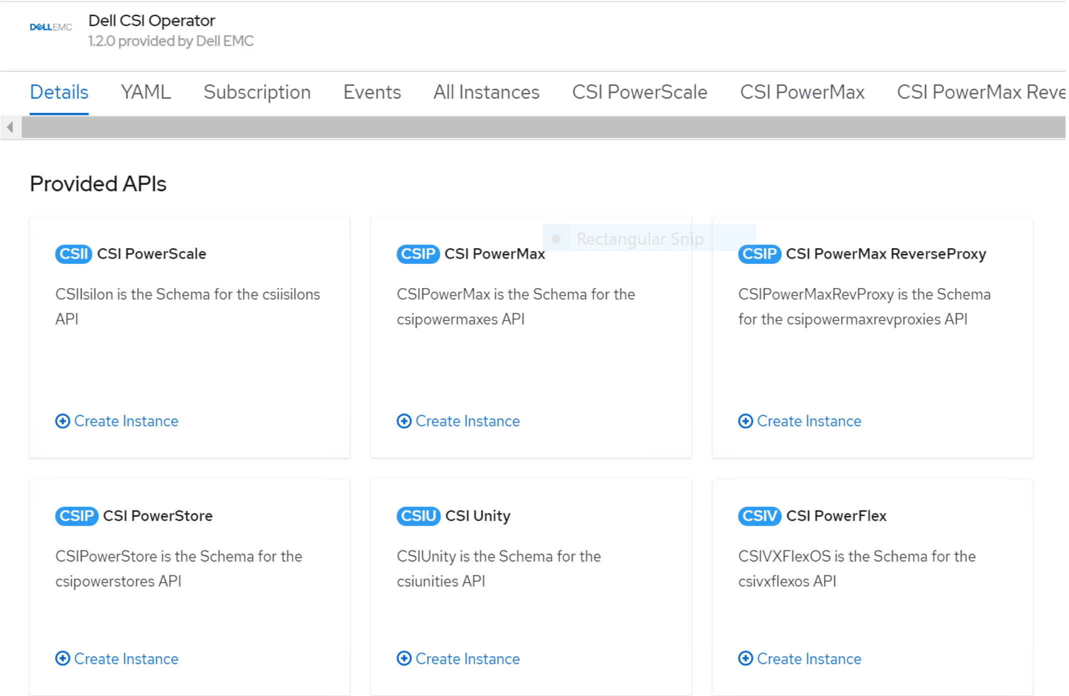
Advanced storage feature support is being added to the CSI driver reference specifications. New to Kubernetes v1.19 is beta support for snapshots, enabling customers to back up and restore application data.

Supported CSI capabilities

Dell Technologies CSI drivers for FC and iSCSI arrays format the volumes with either `xfs` or `ext4` before mounting these volumes to the pods.

Among other factors, consider workload performance and volume access requirements: for example, NFS array is a preferred option for workloads that require concurrent access from multiple clients (such as `Access Mode ReadWriteMany`).

Dell Technologies CSI drivers provide a Red Hat-certified Operator to deploy and manage the life cycle of CSI drivers for OpenShift Container Platform 4.6. Operator deploys and manages the life cycle (installation, upgrade, uninstallation) for all the CSI drivers listed in Table 4, as shown in the following figure:



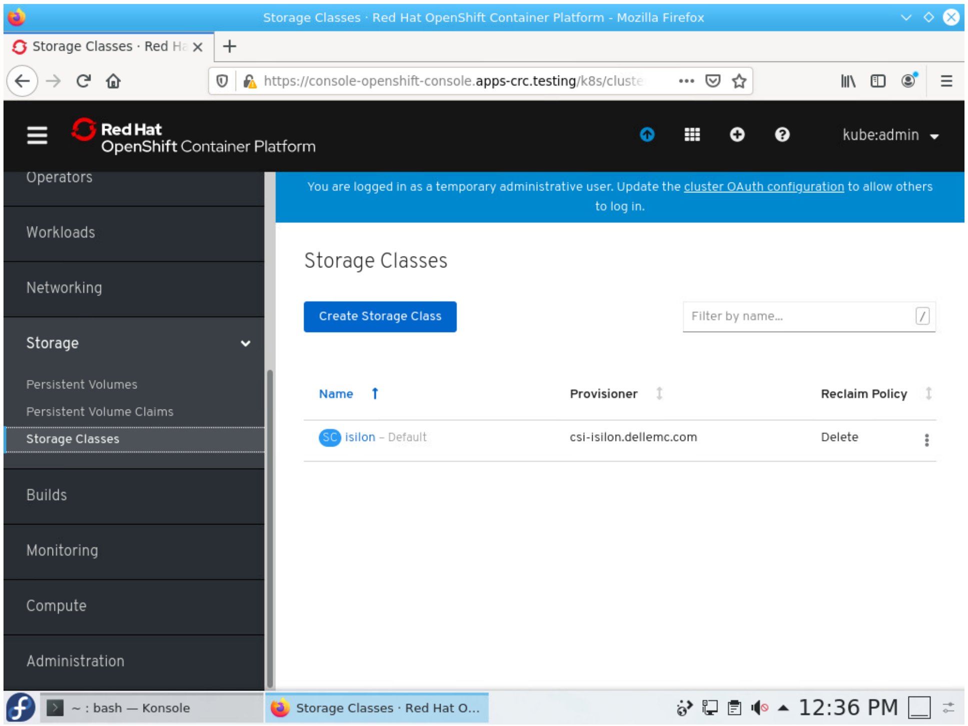
Provided APIs

CSII CSI PowerScale CSIIsilon is the Schema for the csiisilons API Create Instance	CSIP CSI PowerMax CSIPowerMax is the Schema for the csipowermaxes API Create Instance	CSIP CSI PowerMax ReverseProxy CSIPowerMaxRevProxy is the Schema for the csipowermaxrevproxies API Create Instance
CSIP CSI PowerStore CSIPowerStore is the Schema for the csipowerstores API Create Instance	CSIU CSI Unity CSIUnity is the Schema for the csiunities API Create Instance	CSIV CSI PowerFlex CSIVXFlexOS is the Schema for the csivxflexos API Create Instance

Figure 10. Operator-managed drivers

The storage array type dictates specific operator configuration parameters: the API endpoint for the management of the storage platform, protocol, storage pool, and so on.

After the installation is complete, you can access new storage classes directly from the UI and use them as objects with the CLI, as shown in the following figure:



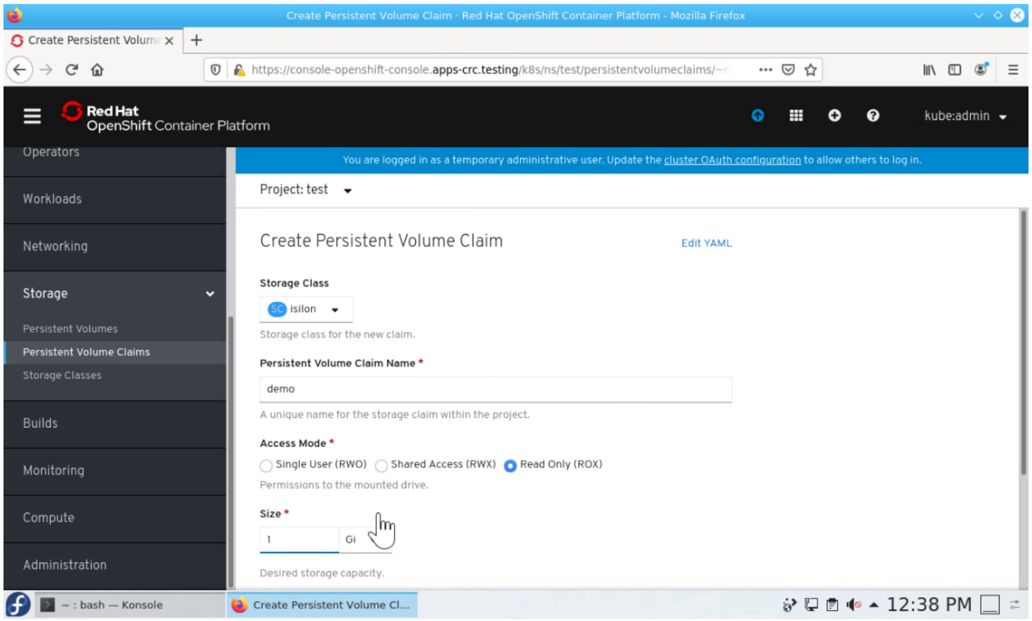
You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Storage Classes

Name	Provisioner	Reclaim Policy
SC isilon – Default	csi-isilon.dellemc.com	Delete

Figure 11. Creating storage classes

You can use the new storage classes in the PV or PVC the same way as the other supported types described in PV types, as shown in the following figure:


Create Persistent Volume Claim · Red Hat OpenShift Container Platform - Mozilla Firefox
Create Persistent Volum
https://console-openshift-console.apps-crc.testing/k8s/ns/test/persistentvolumeclaims/~n
Red Hat
OpenShift Container Platform
kube:admin
Operators
Workloads
Networking
Storage
Persistent Volumes
Persistent Volume Claims
Storage Classes
Builds
Monitoring
Compute
Administration
Edit YAML
~ : bash — Konsole
Create Persistent Volume Cl...
12:38 PM

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: test

Create Persistent Volume Claim

Storage Class

SC isilon

Storage class for the new claim.

Persistent Volume Claim Name *

demo

A unique name for the storage claim within the project.

Access Mode *

○ Single User (RWO) ○ Shared Access (RWX) ● Read Only (ROX)

Permissions to the mounted drive.

Size *

1 Gi

Desired storage capacity.

Figure 12. Creating PVCs

OpenShift administrators can control the storage consumption with quotas. The LimitRange and ResourceQuota directives offer quota capability. Set the quota capability, at the namespace level to enforce a minimum and maximum request size as along with the number of volumes and total consumption. This setting prevents a pod from bloating all the storage resources and potentially affecting future claims.

Chapter 5 Cluster Hardware Design

This chapter presents the following topics:

Introduction

This chapter describes node design options that enable you to build a cluster for a wide range of workload-handling capabilities, expanding on information in Technology and Deployment Process Overview. Usually, the platform design process ensures that the OpenShift Container Platform 4.6 cluster can meet initial workloads. The cluster must also be capable of being scaled out as the demand for workload handling grows. With a clear understanding of your workloads, it is easier to approach CPU sizing, memory configuration, network bandwidth capacity specification, and storage needs. Many operational factors can affect how the complexity of a container ecosystem affects operational latencies. A good practice is to add a safety margin to all physical resource estimates. Our goal in providing this information is to help you get Day-2 operations underway as smoothly as possible.

Cluster scaling

The design and architecture of OpenShift Container Platform place the following resource hosting limits on an OpenShift cluster:

- Nodes per cluster: 2,000
- Pods per cluster: 150,000
- Pods per node: 250
- Pods per core: Not specified; limited by maximum pods per node
- Namespaces per cluster: 10,000
- Number of builds: 10,000 (based on 512 MB RAM per image)
- Pods per namespace: 25,000
- Services per cluster: 10,000
- Services per namespace: 5,000
- Back ends per service: 5,000
- Deployments per namespace: 2,000

Red Hat offers support for OpenShift Container Platform 4.6 up to these limits, as described in Planning your environment according to object maximums.

Requirements planning

Workload resource requirements

This section describes how to size an OpenShift-based container ecosystem cluster by using a sample cloud-native application. The following table shows a cloud-native inventory management application with a customized quotation generation system workload. Estimated memory, CPU core, I/O bandwidth, and storage requirements are indicative of resource requirements at times of peak load.

Table 5. Estimated workload resource requirements by application type

Application type	Number of pods	Maximum memory (GB)	CPU cores	Typical IOPS: Kb/s @ block size (KB)	Persistent storage (GB)
Apache web application	150	0.5	0.5	10 @ 0.5	1
Python-based application	50	0.4	0.5	55 @ 0.5	1
JavaScript runtime	220	1	1	80 @ 2.0	1
Database	100	16	2	60 @ 8.0	15
Java-based tools	110	1.2	1	25 @ 1.0	1.5

The overall resource requirements are: 630 pods, 630 CPU cores, 2,047 GB RAM, 1.9 TB storage, and 130 Gbps aggregate network bandwidth.

Our calculations using the workload information from Table 5 take into account the following considerations:

- For each compute node configuration, it is recommended that you reserve four physical CPU cores per node for infrastructure I/O handling systems.
- Memory configuration is constrained to six DIMM modules per CPU socket (a total of 12 DIMM modules per node).
- DIMM module choices based on current trends for 2,933 MHz memory are 16 GB, 32 GB, and 64 GB. The use of 16 GB DIMM modules results in a minimum node memory configuration of 192 GB.
- NIC options are: 2 x 25 GbE, 4 x 25 GbE, and 2 x 100 GbE.
- Overall compute node configuration options take account of the increased overall node workload handling capacity with processor and memory configuration. The configuration assumes that the compute nodes might be used over time for higher-performance workloads and that additional nodes will be installed to meet future growth in compute, storage, and network areas.

Compute node requirements example

Certain cluster design considerations apply to estimating the required number of compute nodes used. This section outlines these considerations.

- The number of pods required to be deployed is 630, which is clearly above the limit of 250 pods per node. The minimum number of nodes based on the limit of 250 pods per compute node is: 630 / 250 = 3 nodes.
- Table 5 provides estimates for the number of nodes that can be used to accommodate the projected workload. The cluster might require 40, 27, or 14 compute nodes, depending on the design of the node. Field experience recommends caution in the use of estimates for production use.

The following table shows the available configurations:

Table 6. Calculated compute node alternate configurations based on Table 5 data

Compute node type (PowerEdge R640)	Required node quantity	Total CPU cores	Total RAM (GB)
Intel Gold 4208 CPU, 192 GB RAM, 2 x 25 GbE NICs	40	640	7,680
Intel Gold 6226 CPU, 384 GB RAM, 4 x 25 GbE NICs	27	648	10,368
Intel Gold 6252 CPU, 768 GB RAM, 2 x 100 GbE NICs	14	672	10,752

Controller node requirements

Our minimum recommended control-plane node configuration is a PowerEdge R640 server with dual Intel Gold 6226 CPUs and 192 GB RAM. As the Red Hat resource requirements show, this node is large enough for a 250-node cluster and higher. Dell Technologies recommends that you do not scale beyond 200 nodes, so the proposed reference design is adequate for nearly all deployments. The following table shows the sizing recommendations:

Table 7. Control-plane node sizing guide

Number of compute nodes	CPU cores*	Memory (GB)
25	4	16
100	8	32
200	16	64

*Does not include provisioning of at least four cores per node for infrastructure I/O handling

Cluster hardware planning

Ready Stack design limits

The Ready Stack for Red Hat OpenShift Container Platform 4.6 design requires a minimum of four servers for a three-node cluster, with each node running as both a controller node and a compute node. A three-node cluster can be expanded to a five-node cluster if required. You can also expand the five-node cluster with more compute nodes at any time. The maximum configuration that the customized Dell Technologies deployment tools support is 210 servers.

Server, switch, and rack configuration

This design guide uses a server-node base configuration for the PowerEdge R640 and PowerEdge R740xd server nodes that can be used in each node role. For compute nodes that require add-in devices such as GPUs, we strongly recommend PowerEdge R740xd servers. Appendix A shows the PowerEdge server baseline configurations that we used in the design. The following table shows the hardware configuration that is required to build the cluster design that we used for our validation work:

Table 8. Cluster configuration: Number of servers

Node name	Quantity	Configuration
CSAH	1	PowerEdge R640 server configuration
Controller	3	PowerEdge R640 server configuration
Compute	2 or more*	PowerEdge R640 or R740xd server configuration

*A three-node cluster does not require any compute nodes; however, to expand a three-node cluster with additional compute machines; you must first expand the cluster to a five-node cluster with two additional compute nodes.

The following table provides additional cluster configuration information:

Table 9. Cluster configuration: Reference information

Quantity*	Description	Dell Technologies reference
1	Rack enclosure: APC AR3300 NetShelter SZ 42U	APC AR3300 NetShelter SZ 42U
1	Management switch: Dell EMC Networking S3048-ON	Dell EMC PowerSwitch S series 1 GbE switches
2	Data switch: Dell EMC Networking S5248F-ON or Dell EMC Networking S5232-ON	Dell EMC PowerSwitch S series 25/40/50/100 GbE switches
7-210	CSAH, Control-plane: Dell EMC PowerEdge R60 Compute nodes: Dell EMC PowerEdge R640 or Dell EMC PowerEdge R740xd	PowerEdge R640 Rack Server or PowerEdge R740xd Rack Server
2-4	Power distribution unit: APC metered rack PDU 17.2 kW	APC metered rack PDU 17.2 kW

*Rack enclosures and power distribution units are site-specific. Review the physical dimensions and power requirements in a site survey.

Validated hardware configuration options

Introduction

For validation test work in our laboratories, we used various server configurations for the Ready Stack for OpenShift Container Platform 4.6. Dell Technologies recommends selecting server configurations that are known to provide a satisfactory deployment experience and to meet or exceed Day-2 operating experience expectations. This chapter provides guidelines for Intel microprocessor selection, memory configuration, local (on-server) disk storage, and network configuration.

Selecting the server processors

The Intel Xeon Gold processor family provides performance, advanced reliability, and hardware-enhanced security for demanding compute, network, and storage workloads.

For clusters of 30 or more nodes, Dell Technologies recommends Intel Xeon Gold series CPUs in the range of the 6226 to 6252 models. This selection is based on experience that we gained from deployment and operation of OpenShift Container Platform 4.6 running on PowerEdge R640 and R740xd servers. The design information in this guide is based on clusters of servers with either Intel Gold 6240 or Intel Gold 6238 processors.

Dell Technologies realizes that many sites prefer to use a single-server configuration for all node types. However, this option is not always cost-effective or practical.

When selecting a processor, consider the following recommendations:

- **Processor core count**—The processor core count must be sufficient to ensure satisfactory performance of the workload operations and base services that are running on each node.
- **Thermal design power (TDP)**—The CPU must be suitable for the amount of heat that is removed from the server through the heat sinks and cooling air flow.
- **Ability to dissipate heat**—During validation work with high-core-count, high-TDP processors, the thermal delta (air discharge temperature minus air intake temperature) across a server was recorded at 65°F. Excessive air discharge (egress) temperature from the server might lead to a premature server-component or system failure.
- **Compute node configurations**—The design of compute nodes for use as part of your OpenShift Container Platform cluster can use many compute node configurations. Compute nodes can use Intel or AMD-based CPU platforms. The processor architecture and core count per node selection can significantly affect the acquisition and operating cost of the cluster that is needed to run your organization’s application workload.

When ordering and configuring your PowerEdge servers, see the *Dell EMC PowerEdge R640 Technical Guide* and *Dell EMC PowerEdge R740 and R740xd Technical Guide*.

For CPU information, see Intel Xeon Gold Processors.

Per-node memory configuration

The Dell Technologies engineering team designated 192 GB, 384 GB, or 768 GB of RAM as the best choice based on memory usage, DIMM module capacity for the current cost, and likely obsolescence over a five-year server life cycle. We chose a midrange memory configuration of 384 GB RAM to ensure that the memory for each CPU has multiples of three banks of DIMM slots that are populated to ensure maximum memory-access cycle speed. Modify the memory configuration to meet your budgetary constraints and operating needs.

Also, consult OpenShift architectural guidance and consider your own observations from running your workloads on OpenShift Container Platform 4.6. For guidance about server memory population (location of DIMM modules in DIMM slots) and, in particular, the use of the firmware setting for “Performance Optimized” mode, see the following Dell Technologies Knowledge Base article: Dell EMC PowerEdge-14G Memory Population Rules updated for certain server's configurations.

Disk drive capacities

The performance of disk drives significantly limits the performance of many aspects of OpenShift cluster deployment and operation. We validated deployment and operation of OpenShift Container Platform using magnetic storage drives (spinners), SATA SSD drives, SAS SSD drives, and NVMe SSD drives.

Our selection of all NVMe SSD drives was based on a comparison of cost per GB of capacity divided by observed performance criteria such as deployment time for the cluster and application deployment characteristics and performance. While there are no universal guidelines, over time users gain insight into the capacities that best enable them to meet their requirements. Optionally, you can deploy the cluster with only hard drive disk drives. This configuration has been shown in testing to have few adverse performance consequences.

Network controllers and switches

When selecting the switches to include in the OpenShift Container Platform cluster infrastructure, consider the overall balance of I/O pathways within server nodes, the network switches, and the NICs for your cluster. When you choose to include high-I/O bandwidth drives as part of your platform, consider your choice of network switches and NICs so that sufficient network I/O is available to support high-speed, low-latency drives:

- **HDD drives**—These drives have lower throughput per drive. You can use 10 GbE for this configuration.
- **SATA/SAS SSD drives**—These drives have high I/O capability. SATA SSD drives operate at approximately four times the I/O level of a spinning HDD. SAS SSDs operate at up to 10 times the I/O level of a spinning HDD. With SSD drives, configure your servers with 25 GbE.
- **NVMe SSD drives**—These drives have high I/O capability, up to three times the I/O rate of SAS SSDs. We populated each node with 4 x 25 GbE NICs, or 2x 100 GbE NICS, to provide optimal I/O bandwidth.

The following table provides information about selecting NICs to ensure adequate I/O bandwidth and to take advantage of available disk I/O:

Table 10. NIC selection to optimize I/O bandwidth

NIC selection	Compute node storage device type
2 x 25 GbE	Spinning magnetic media (HDD)
2 x 25 GbE or 4 x 25 GbE	SATA or SAS SSD drives
4 x 25 GbE or 2 x 100 GbE	NVMe SSD drives

True network HA fail-safe design demands that each NIC is duplicated, permitting a pair of ports to be split across two physically separated switches. A pair of PowerSwitch S5248F-ON switches provides 96 x 25 GbE ports, enough for approximately 20 servers. This switch is cost-effective for a compact cluster. While you could add another pair of S5248F-ON switches to scale the cluster to a full rack, consider using PowerSwitch S5232F-ON switches for a larger cluster.

The PowerSwitch S5232F-ON provides 32 x 100 GbE ports. When used with a four-way QSFP28 to SFP28, a pair of these switches provides up to 256 x 25 GbE endpoints, more than enough for a rackful of servers in the cluster before more complex network topologies are required.

Low latency in an NFV environment

NFV-centric data centers require low latency in all aspects of container ecosystem design for application deployment. This requirement means that you must give attention to selecting low-latency components throughout the OpenShift cluster. We strongly recommend using only NVMe drives, NFV-centric versions of Intel CPUs, and, at a minimum, the PowerSwitch S5232F-ON switch. Consult the Dell Technologies Service Provider support team for specific guidance.

Power configuration

Dell Technologies strongly recommends that all servers are equipped with redundant power supplies and that power cabling provides redundant power to the servers. Configure each rack with pairs of power distribution units (PDUs). For consistency, connect all right-most power supply units (PSUs) to a right-side PDU and all left-most PSUs to a left-side PDU. Use as many PDUs as you need, in pairs. Each PDU must have an independent connection to the data center power bus.

The following figure shows an example of the power configuration that is designed to ensure a redundant power supply for each cluster device:

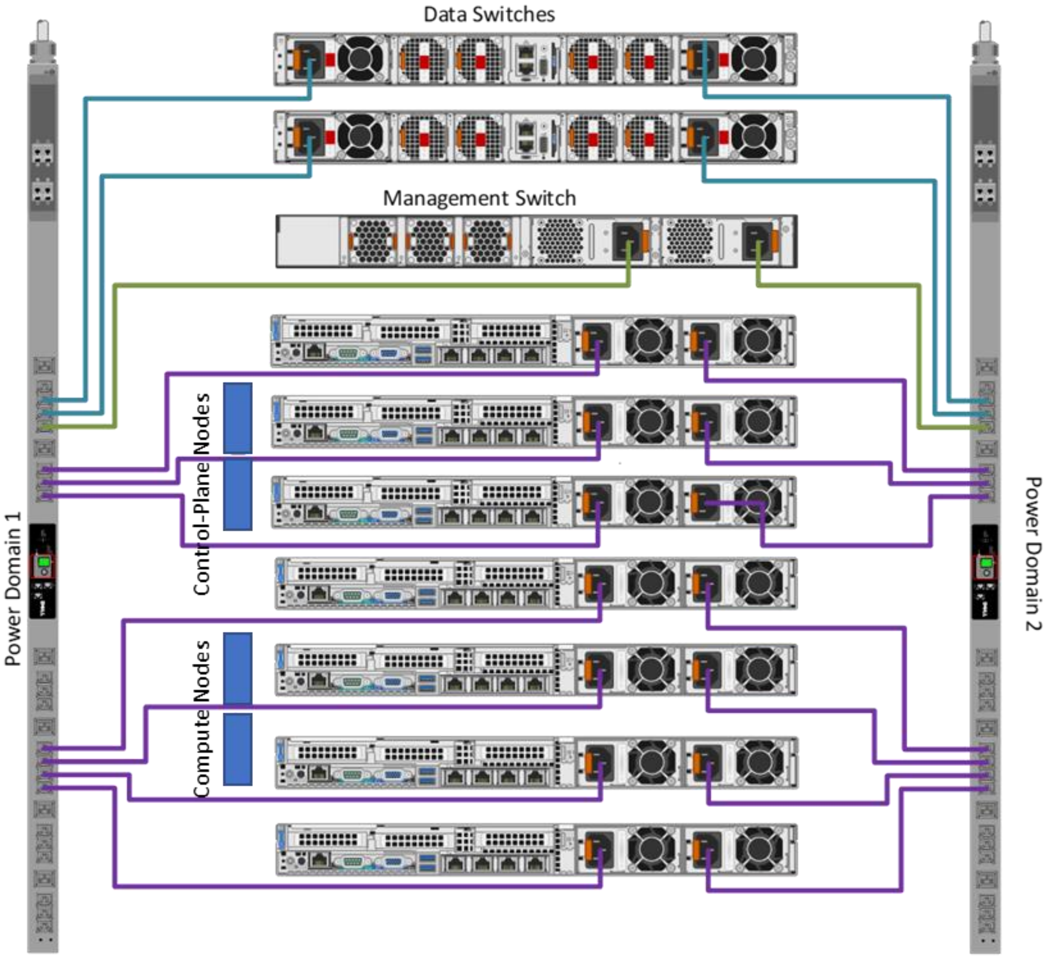


Figure 13. **PSU to PDU power template**

Chapter 6 Use Cases

This chapter presents the following topics:

Introduction

This chapter describes how a Ready Stack for OpenShift Container Platform 4.6 solution supports several different uses cases across both enterprise and service provider markets. The examples in this chapter include enterprise application development and deployment, telecommunications service provider operations, and data analytics and artificial intelligence.

Enterprise applications

Today’s applications are developed using cloud-native principles. These applications offer the agility, speed, and composability of microservices architecture. Enterprise deployment of these applications comes with the following additional requirements:

- Enterprises require geographically distributed deployment of container platforms as a means of *disaster avoidance*.
- Large enterprise deployments might cater to many internal and external partners and customers. This consideration requires multitenancy and user access roles in the container platform deployments. Security and isolation are required between microservices.
- Infrastructure requirements to meet the application needs of network connectivity, storage capacity, and compute.

Site deployment models

Enterprises must deploy a single cluster or multiple highly scalable large clusters in each data center. Dell Technologies strongly discourages spanning a single cluster across multiple geographical sites because Kubernetes has low latency requirements. One of the key decisions we made regarding multisite OpenShift Container Platform 4.6 deployment was to deploy multiple OpenShift clusters across multiple sites. A major advantage of this model of deployment is disaster avoidance. Service can continue even when a disaster occurs at a site.

The following figure shows a multisite deployment of this solution:

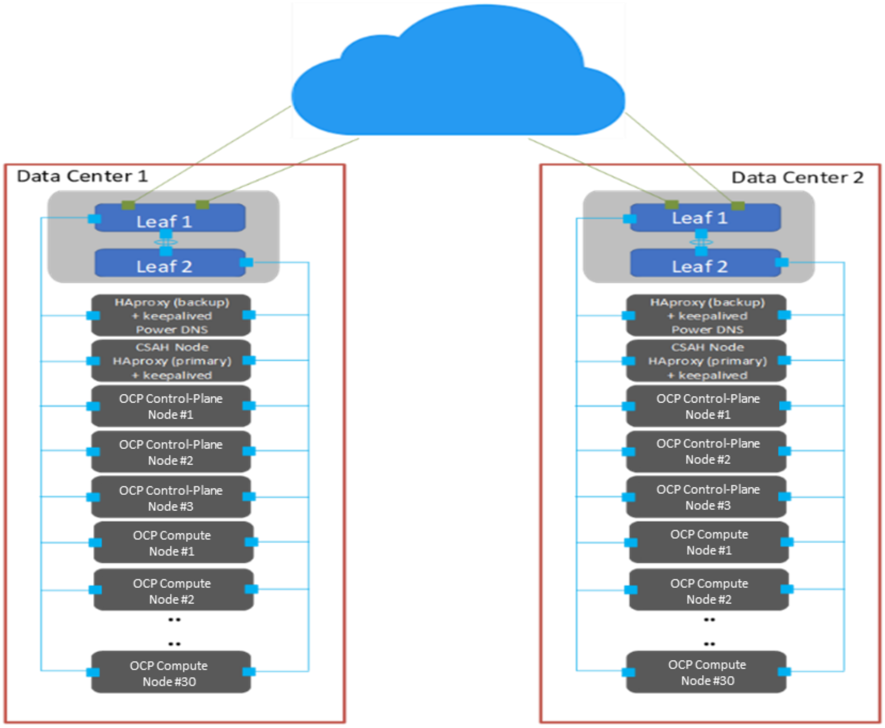

Data Center 1
Leaf 1
Leaf 2
HAproxy (backup) + keepalived Power DNS
CSAH Node HAproxy (primary) + keepalived
OCP Control-Plane Node #1
OCP Control-Plane Node #2
OCP Control-Plane Node #3
OCP Compute Node #1
OCP Compute Node #2
OCP Compute Node #30
Data Center 2
Leaf 1
Leaf 2
HAproxy (backup) + keepalived Power DNS
CSAH Node HAproxy (primary) + keepalived
OCP Control-Plane Node #1
OCP Control-Plane Node #2
OCP Control-Plane Node #3
OCP Compute Node #1
OCP Compute Node #2
OCP Compute Node #30

Figure 14. Multisite OpenShift Container Platform deployment

As shown in Figure 14, key technical components of the solution include:

- RedHat OpenShift Container Platform 4.6
- Load balancer: Global traffic manager (GTM) and local traffic manager (LTM)
- Data center hardware infrastructure

Role-based access control

Access control has implications for what multitenancy means throughout the infrastructure: Portal access and views, logging information, and usage information must be linked to the user role. For example:

- A provider administrator must be able to see usage and metering information for the entire infrastructure.
- A tenant administrator requires access only to the infrastructure that is assigned to that tenant.
- Tenant users require access only to assets and resources that they are permitted to manage.

Role-based access control (RBAC) in OpenShift Container Platform 4.6 can be linked to your Microsoft Active Directory identity management environment or other supported identity managers. This link gives control over user and group access to the container ecosystem infrastructure and services, providing a good foundation for multitenancy support. The following table shows the supported roles:

Table 11. Role-based access control in OpenShift Container Platform 4.6

Role	Description
admin	Project manager
basic-user	User who can get information about projects and users.
cluster-admin	A superuser who can perform any action in any project.
cluster-status	User who can get cluster status information.
edit	User who can modify objects in a project
self-provisioner	User who can create their own projects
view	User who can see most objects in a project.
cluster-reader	User who can read, but not view, objects in the cluster.

Security and isolation

OpenShift Container Platform 4.6 is built on the concept that each project running within a cluster can be isolated from every other project. The project manager must have the administrative privilege to be able to see any other project in the cluster.

Performance monitoring and logging

Cloud service providers typically require the ability to monitor and report on system utilization. OpenShift Container Platform 4.6 includes Prometheus system monitoring and metering and provides capability for extensive data logging. For more information about obtaining cluster resource consumption to drive usage billing through third-party application software, see the following Red Hat documentation:

- About cluster monitoring
- Examples of using metering
- About cluster logging and OpenShift Container Platform

The cluster monitoring operator controls the monitoring components that are deployed and the Prometheus operator controls Prometheus and Alert manager instances. The platform monitors the following stack components:

- Stack component
- CoreDNS
- Elasticsearch
- Etcd
- Fluentd
- HAProxy
- Image registry
- Operator Lifecycle Manager (OLM)
- Telemeter client
- Kubelets
- Kubernetes apiserver
- Kubernetes controller manager

- Kubernetes scheduler
- Metering
- OpenShift apiserver
- OpenShift controller manager

Telecommunications industry

Introduction

A typical telecommunications(telco) company sells telco-oriented applications as a service to its consumers. Telco use-case requirements vary depending on the virtual network functions (VNFs) that are being serviced. These functions include:

- Content delivery network (CDN)
- Edge infrastructure and towers of power
- NFV management and operations (NFV-MANO)
- Software-defined networking (SDN) and SD-WAN management
- Radio access networks (RAN) and 5G, and their component service infrastructures
- Multiaccess Edge Computing (MEC)
- Core network and 5G Next Generation Core (NGC)

This use case identifies some key design factors for a telco container platform.

Content delivery network

Online video consumption has grown in recent years. High-quality video delivery over public networks requires a CDN. To handle growth, many operators are considering the virtualization of the CDN, giving them an ability to scale CDN on demand. CDN virtualization permits simple provisioning and sharing of resources with other telco services, simplifying operations and avoiding costly dedicated infrastructure.

The following figure shows a virtual CDN (vCDN):

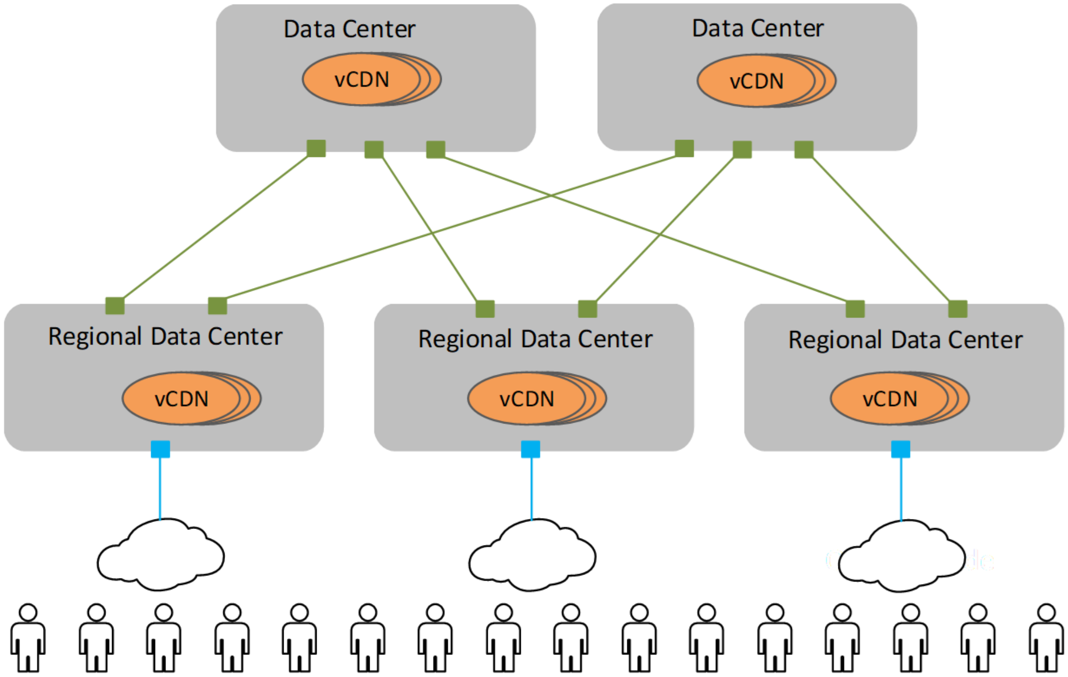


Figure 15. Virtual content delivery network (vCDN)

vCDN platform requirements

A vCDN stack requires the following principal capabilities:

- Large application storage space to store video and other files
- High-speed and low-latency network options to serve the content
- Rapid ramp-up of on-demand processing capacity

Note: The Dell EMC Isilon storage array has been renamed to PowerScale.

OpenShift Container Platform 4.6 on PowerEdge hardware platforms meets these demands by providing the following capabilities:

- The CSI storage drivers for Dell EMC Unity, PowerMax, PowerFlexOS, and PowerScale (formerly Isilon) are being developed and validated currently. These drivers can be integrated into your OpenShift Container Platform deployment using the new CSI plug-ins.
- High-speed (25 GbE/100 GbE) network interfaces of Dell Technologies server and switch portfolios meet the networking needs of network I/O-intensive applications.
- OpenShift Container Storage (based on Ceph) is supported as part of OpenShift Container Platform 4.6.
- Multus CNI plug-in support by which additional networks can be added to each container so that the container can meet capacity needs on targeted networks.
- SR-IOV is now natively supported by OpenShift Container Platform. Red Hat provides an SR-IOV Operator over OperatorHub, enabling administrators to manage virtual functions on nodes through Kubernetes CRDs.

- Telco applications generally use huge pages. In OpenShift Container Platform, applications can allocate and consume huge pages.
- OpenShift Container Platform 4.6 includes support for IPv6.

Container ecosystem clusters in telco operations are likely to be large, spanning multiple racks. OpenShift Container Platform running on PowerEdge servers scales to approximately 210 nodes (seven racks when you use PowerEdge R640 servers). We highly recommend using leaf-spine networking when scaling to more than three racks per cluster. Although the cluster can be scaled beyond seven racks, undertake this effort only as a custom engineering project. The deployment of large clusters requires significant modification of the Ansible playbooks that we generated to facilitate large-scale deployment.

Data analytics and artificial intelligence

Introduction

Enterprises are rapidly increasing their investments in infrastructure platforms to support data analytics and artificial intelligence (AI), including the more specific AI disciplines of machine learning (ML) and deep learning (DL). All these disciplines benefit from running in containerized environments. The benefits of running these applications on OpenShift Container Platform are available to developers, data scientists, and IT operators.

For simplicity, we use “data analytics as a service” (DAaaS) for analytics and AI that are operated and instantiated in a containerized environment. OpenShift Container Platform enables operators to create a DAaaS environment as an extensible analytics platform with a private cloud-based delivery model. This delivery model makes various tools available for data analytics and can be configured to efficiently process and analyze huge quantities of heterogeneous data from shared data stores.

The data analytics life cycle, particularly the ML life cycle, is a multiphase process of integrating large volumes and varieties of data, abundant compute power, and open-source languages, libraries, and tools to build intelligent applications and predictive outcomes. At a high level, the life cycle consists of these phases:

- **Data acquisition and preparation**—Ensures that the input data is complete and of a high quality
- **Modeling creation**—Includes training, testing, and selection of the model with the highest prediction accuracy
- **Model deployment**—Includes inferencing in the application development and operations processes

Key challenges

Data scientists and engineers are primarily responsible for developing modeling methods that ensure that the selected outcome continues to provide the highest prediction accuracy. The key challenges that data scientists face include:

- Selection and deployment of the right AI tools (such as Apache Spark, TensorFlow, PyTorch, and so on)
- Complexities and time required to train, test, select, and retrain the AI model that provides the highest prediction accuracy

- Slow execution of AI modeling and inferencing tasks because of a lack of hardware acceleration
- Limited IT operations to provision and manage infrastructure
- Collaboration with data engineers and software developers to ensure input data hygiene and successful AI model deployment in application development processes

Containers and Kubernetes are key to accelerating the data analytics life cycle because they provide data scientists and IT operators with the agility, flexibility, portability, and scalability needed to train, test, and deploy ML models.

OpenShift Container Platform provides all these benefits. Through its DevOps capabilities and integration with hardware accelerators, the platform enables better collaboration between data scientists and software developers. OpenShift Container Platform also accelerates the roll-out of analytics applications to departments as needed. The benefits include the ability to:

- Empower data scientists with a consistent, self-service-based, cloud-like experience:
  - Gives data scientists the flexibility and portability to use containerized ML tools of their choice to quickly build, scale, reproduce, and share ML modeling results in a consistent way with peers and software developers.
  - Eliminates dependency on IT to provision infrastructure for iterative, compute-intensive ML modeling tasks.
- Accelerate compute-intensive ML modeling and inferencing jobs:

  On-demand access to high-performance hardware can seamlessly meet the high compute resource requirements to help determine the best ML model, providing the highest prediction accuracy.
- Streamline the development and operations of intelligent applications:

  Extending OpenShift DevOps automation capabilities to the ML life cycle enables collaboration between data scientists, software developers, and IT operations so that ML models can be quickly integrated into the development of intelligent applications.

MLPerf on OpenShift

A recent white paper explored the implications of running resource-intensive ML applications on top of OpenShift. MLPerf benchmarks are an independent valuation of performance for various parts of the machine learning ecosystem, including both the cloud and hardware platforms being used. The MLPerf training and inference benchmarks were run on top of OpenShift and compared to Nvidia's MLPerf benchmark results. The Nvidia MLPerf benchmarks were not run on top of a container automation platform. The results indicated that the addition of the OpenShift platform did not hamper the performance of intensive ML applications and demonstrated that OpenShift provides valuable benefits for running ML applications in production environments.

Kubeflow ML on OpenShift

One example of ML on OpenShift Container Platform is the work that Dell Technologies and Red Hat did to deploy Kubeflow on OpenShift.

Kubeflow is an open-source Kubernetes-native platform for ML workloads that enables enterprises to accelerate their ML/DL projects. Based originally on Google's use of TensorFlow on Kubernetes, Kubeflow is a composable, scalable, portable ML stack that includes components and contributions from a variety of sources and organizations. Kubeflow bundles popular ML/DL frameworks such as TensorFlow, MXNet, PyTorch, and Katib with a single deployment binary file. By running Kubeflow on OpenShift Container Platform, you can quickly operationalize a robust ML pipeline.

For more information, see the *Machine Learning Using the Dell EMC Ready Architecture for Red Hat OpenShift Container Platform White Paper* (this white paper is based on the OpenShift Container Platform 4.2 release).

For more information, see Kubeflow: The Machine Learning Toolkit for Kubernetes.

Spark analytics on OpenShift

An example of large-scale data analytics being run on OpenShift Container Platform is the Dell EMC Spark on Kubernetes Ready Solution for Data Analytics.

Apache Spark, a unified analytics engine for big data and ML, is one of the largest open-source projects in data processing. Data scientists want to run Spark processes that are distributed across multiple systems to have access to additional memory and computing cores. OpenShift orchestrates the creation, placement, and life cycle management of those Spark processes across a cluster of servers by using container virtualization to host the processes.

For more information, see the *Spark on Kubernetes reference architecture* on the Dell Technologies Info Hub for AI and Data Analytics.

SQL Server big data clusters on OpenShift

Another example of big data analytics being run on OpenShift is the Dell EMC solution for Microsoft SQL Server 2019 Big Data Clusters.

SQL Server Big Data Clusters enable deployment of scalable clusters consisting of SQL Server, Spark, and HDFS containers running on Kubernetes. These components run side by side to enable you to read, write, and process big data so that you can easily combine and analyze your high-value relational data with high-volume big data. OpenShift Container Platform is one of the Kubernetes platforms on which you can run SQL Server Big Data Clusters.

For more information, see the *Microsoft SQL Server 2019 Big Data Clusters White Paper* on the Dell Technologies Info Hub for SQL Server.

Chapter 7 References

This chapter presents the following topics:

Dell Technologies documentation

The following Dell Technologies documentation provides additional information. Access to these documents depends on your login credentials. If you do not have access to a document, contact your Dell Technologies representative.

- Dell Technologies Info Hub for Red Hat OpenShift Container Platform
- Dell EMC Ready Stack Converged Infrastructure
- Dell EMC PowerEdge R640 Technical Guide
- Dell EMC PowerEdge R740 and R740xd Technical Guide
- Machine Learning Using the Dell EMC Ready Architecture for Red Hat OpenShift Container Platform (this white paper is based on OpenShift Container Platform 4.2).
- Dell EMC Unity: Best Practices Guide

Red Hat documentation

The following Red Hat resources provide additional information:

- Installing On Bare Metal
- What are Operators?
- Understanding Red Hat OpenShift Service Mesh
- Recommended Cluster Scaling Practices
- Understanding the monitoring stack
- Examples of using metering
- Understanding cluster logging
- Machine Management
- Planning your environment according to object maximums

Other resources

The following resources provide additional information:

- Intel Xeon Gold Processors
- Kubeflow: The Machine Learning Toolkit for Kubernetes
- Prometheus: From metrics to insight
- Operating etcd clusters for Kubernetes

Appendix A Dell EMC PowerEdge BOMs

This appendix presents the following topics:

Dell EMC PowerEdge R640 node BOM

The following table lists the key recommended parts per node. Memory, CPU, NIC, and drive configurations are preferred but not mandated.

Note: When orders are placed, the Dell Technologies ordering center adds new SKUs and substitutes those that are shown in the table with current local SKUs.

Table 12. PowerEdge R640 baseline server BOM

Qty	SKU	Description
1	210-AKWU	PowerEdge R640 server
1	329-BEIJ	PowerEdge R640 MLK motherboard
1	321-BCQQ	2.5 in. chassis with up to 10 hard drives, 8 NVMe drives, and 3 PCIe slots, 2 CPU only
2	338-BTSI	Intel Xeon Gold 6238 2.1G, 22C/44T, 10.4GT/s, 30.25M Cache, Turbo, HT (140W) DDR4-2933
1	370-ABWE	DIMM blanks for system with 2 processors
2	412-AAIQ	Standard 1U Heatsink
1	370-AEVR	3200 MT/s RDIMMs
1	370-AAIP	Performance-optimized
12	370-AEVN	32 GB RDIMM, 3200MT/s, Dual Rank
1	405-AAJU	HBA330 12 Gbps SAS HBA Controller (NON-RAID), minicard
1	385-BBKT	iDRAC9, Enterprise
1	379-BCQV	iDRAC Group Manager, enabled
1	379-BCSG	iDRAC, legacy password
1	379-BCRB	DHCP with Zero Touch Configuration
1	330-BBGN	Riser Config 2, 3 x 16 LP
1	406-BBLG	Mellanox ConnectX-4 Lx Dual Port 25 GbE SFP 28 rNDC
1	406-BBLD	Mellanox ConnectX-4 Lx dual port 25 GbE SFP28 NIC, low profile
1	429-AAIQ	No internal optical drive
1	384-BBQI	8 performance fans for the R640 server
1	450-ADWS	Dual, hot-plug, redundant power supply (1+1), 750W
2	492-BBDH	C13 to C14, PDU Style, 12 AMP, 2 ft. (.6m) power cable, North America
1	800-BBDM	UEFI BIOS boot mode with GPT partition
1	770-BBBC	ReadyRails sliding rails without cable management arm
1	366-0193	Std BIOS setting power management—maximum performance

Qty	SKU	Description
2 min – 8 max	400-BELT	Dell 1.6 TB, NVMe, Mixed Use Express Flash, 2.5 SFF Drive, U.2, P4610 with Carrier
2	400-AZQO	800 GB SSD SAS Mix Use 12Gbps512e 2.5in Hot-plug AG Drive, 3 DWPD, 4380 TBW
1	403-BCHI	BOSS Cntrl + 2 M.2 240G, R1, LP1

Dell EMC PowerEdge R740xd node BOM

The following table shows the PowerEdge Server R740xd baseline configurations that are used in the design of the Ready Stack for OpenShift Container Platform 4.6.

Note: When orders are placed, the Dell Technologies ordering center adds new SKUs and substitutes those that are shown in the table with current local SKUs.

Table 13. PowerEdge R740xd baseline server BOM

Qty	SKU	Description
1	210-AKZR	PowerEdge R740XD Server
1	329-BEIK	PowerEdge R740/R740XD MLK motherboard
1	321-BCRC	Chassis up to 24 x 2.5 in. hard drives including 12 NVME drives, 2 CPU configuration
1	338-BTSI	Intel Xeon Gold 6238 2.1G, 22C/44T, 10.4GT/s, 30.25M Cache, Turbo, HT (140W) DDR4-2933
1	412-AAIR	Standard 2U Heatsink
1	370-AEVR	3200 MT/s RDIMMs
12	370-AEVN	32 GB RDIMM, 2933MT/s, Dual Rank
1	780-BCDI	No RAID
1	405-AANK	HBA330 controller adapter, low profile
1	365-0354	CFI, standard option not selected
1	385-BBKT	iDRAC9, Enterprise
1	379-BCQV	iDRAC Group Manager, enabled
1	379-BCSG	iDRAC, legacy password
1	385-BBLG	Static IP
1	330-BBHD	Riser Config 6, 5 x 8, 3 x1 6 slots
1	406-BBLG	Mellanox ConnectX-4 Lx Dual Port 25 GbE SFP28 rNDC
1	406-BBLE	Mellanox ConnectX-4 Lx Dual Port 25 GbE SFP28 network interface controller
1	384-BBPZ	6 performance fans for R740/740XD
1	450-ADWM	Dual, hot-plug, redundant power supply (1+1), 1100W
1	492-BBDH	C13 to C14, PDU Style, 12 AMP, 2 ft (0.6m) power cable, North America
1	325-BCHU	PowerEdge 2U standard bezel
1	800-BBDM	UEFI BIOS Boot Mode with GPT partition
1	770-BBBQ	ReadyRails sliding rails without cable management arm
1	366-0193	Std Bios setting power management - maximum performance
1	403-BCHP	BOSS Cntrl + 2 M.2 240G, R1, FH

Qty	SKU	Description
Select *one* of the following rows:		
1 to 24	Check part at time of ordering	800 GB, 1.92 TB, or 3.84 TB SSD SAS mixed use 12 Gbps 512e 2.5 in. hot-plug AG drive with carrier, 3 DWPD, 4380 TBW, CK
1 to 12	Check part at time of ordering	Dell 1.6 TB, 3.2 TB, or 6.4 TB, NVMe, mixed use express flash, 2.5 SFF drive, U.2, P4610 with carrier, CK

Dell EMC Unity 380F BOM

The following table shows the Dell EMC Unity 380F baseline configurations that are used in the design of the Ready Stack for OpenShift Container Platform 4.3.

Note: When orders are placed, the Dell Technologies ordering center adds new SKUs and substitutes those that are shown in the table with current local SKUs.

Table 14. Dell EMC Unity 380F BOM

Qty	SKU	Description
8	D4F-2SFXL2-1920	D4F 1.92 TB ALL FLASH 25X2.5 SSD
1	D4ODPEKITAF	UNITY 380F DPE INSTALL KIT
2	C13-PWR-12	2 C13 CORDS NEMA 5-15 125V 10A - NON DPE
1	D4BD6C25FAFLL	UNITY 380F DPE 25 X 2.5 DELL FLD RCK
1	D4SFP16FAF	UNITY CNA 4X16GB FC SFPS AF
1	D4SL25IO4PTAF	UNITY 2X4 PORT IO 25GBE OPT AF
1	M-PSM-HWE-005	PROSUPPORT 4HR/MC HARDWARE SUPPORT
1	458-002-526	UNITY AFA BASE SOFTWARE=IC
1	M-PSM-SWE-005	PROSUPPORT 4HR/MC SOFTWARE SUPPORT
1	PS-PD-UXAFXDP	PD FOR UNITY XT AF

Dell EMC PowerMax BOM

The following table shows the Dell EMC PowerMax (formerly VMAX) baseline configurations that are used in the design of the Ready Stack for OpenShift Container Platform 4.6.

Note: When orders are placed, the Dell Technologies ordering center adds new SKUs and substitutes those that are shown in the table with current local SKUs.

Table 15. : Dell EMC PowerMax BOM

Qty	SKU	Description
1	SYSTEM	VMAX100K
1	E-FE80000E	VMAX VG 8MM 8G FIBRE
1	EL6101200SB	VMAX VG 1200GB 10K SAS DRV SPARE
1	EL6F3960SBT0	VMAX3 960GB FLASH SPARE
1	EL-512BASE	VMAX 100K BASE 512GB
1	E-DE120	VMAX VG 120 SLT DR ENCL
1	E-DIR3MCBL	VMAX VG DIRECT CONNECT 3 METER

Qty	SKU	Description
2	E-ACON3P-50	ADPTR AC 3PH 50A W3-4IN CONDUIT ADPTR
8	EL61012006B	VMAX VG 1200 10K SAS DRV R6(6+2)
8	EL6F39605BT0	VMAX3 960GB FLASH R5(3+1)
8	E-GE-ISCSI	VMAX VG GIGE ISCSI PORT TRACKING MODEL
1	E-FE00800T	VMAX VG 8MM 10GIGE
1	E-SKINS	VMAX VG SIDE PANELS
1	E-PCBL3DHR	PWR CBL HBL-RSTOL 3D
1	E-FDOORSYS1E	SB1 SINGLE ENGINE HEX DELL DOOR
13	E-OPROVISION	OPROVISION FACTOR TRACKING MODEL
1	E-SYS1L-3D	VMAX 100K SYS BAY1 3D
1	E-1ENG	VMAX VG SINGLE ENGINE SYS BAY
1	DX-LITE-EM	DX LITE TRACKING MODEL
2	E-1600MODS	VMAX VG FLASH MODS 1600
1	E-Q217C	VMAX Q217C TRACKING MODEL
1	E-SMOD	SZR CONFIG TRACKING MODEL
1	WKPROFILE-BAL	VMAX VG WORKPROFILE BALANCED
1	W-PSM-HW-001	PROSUPPORT W/MISSION CRITICAL-HW WARRANT
1	WU-PSP-HW-001	PROSUPPORT PLUS HARDWARE WARRANTY UPG
1	ESRS-GW-200	EMC SECURE REMOTE SUPT GATEWAY CLIENT=IC
1	458-001-534	VMAX3 HYPERMAX OS PRODUCT
1	456-108-357	V100K HYPERMAX OS BASE LIC=IC
1	M-PSP-SW-016	PROSUPPORT PLUS 4HR/MC SOFTWARE SUPPORT
1	450-001-220	V100K ADVANCED PACKAGE=IC
1	M-PSP-SW-016	PROSUPPORT PLUS 4HR/MC SOFTWARE SUPPORT
1	458-001-535	VMAX3 HYPERMAX OS CAPACITY
12	456-106-425	VMAX3 HYPERMAX OS BASE 0-50TB=CC
1	M-PSP-SW-016	PROSUPPORT PLUS 4HR/MC SOFTWARE SUPPORT
12	450-001-221	VMAX3 ADVANCED PKG 0-50TB=CC
1	M-PSP-SW-016	PROSUPPORT PLUS 4HR/MC SOFTWARE SUPPORT
1	PS-PD-VMXDP	PD FOR VMAX
1	PSINST-ESRS	ZERO DOLLAR ESRS INSTALL
1	CE-VMAXAF-VIDVPK	VMAX VIDEO VALUEPAK VILTS 2 TITLES=UC