Red Hat OpenShift Container Storage 4.8

Planning your deployment

Important considerations when deploying RHOCS 4.8

Last Updated: 2022-06-15

Red Hat OpenShift Container Storage 4.8 Planning your deployment

Important considerations when deploying RHOCS 4.8

Abstract

Read this document for important considerations when planning your Red Hat OpenShift Container Storage deployment.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message .

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Do let us know how we can make it better. To give feedback:

- For simple comments on specific passages:
  1. Make sure you are viewing the documentation in the *Multi-page HTML* format. In addition, ensure you see the **Feedback** button in the upper right corner of the document.
  2. Use your mouse cursor to highlight the part of text that you want to comment on.
  3. Click the **Add Feedback** pop-up that appears below the highlighted text.
  4. Follow the displayed instructions.
- For submitting more complex feedback, create a Bugzilla ticket:
  1. Go to the Bugzilla website.
  2. In the **Component** section, choose **documentation**.
  3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
  4. Click **Submit Bug**.

CHAPTER 1. INTRODUCTION TO OPENSHIFT CONTAINER STORAGE

Red Hat OpenShift Container Storage is a highly integrated collection of cloud storage and data services for Red Hat OpenShift Container Platform. It is available as part of the Red Hat OpenShift Container Platform Service Catalog, packaged as an operator to facilitate simple deployment and management.

Red Hat OpenShift Container Storage services are primarily made available to applications by way of storage classes that represent the following components:

- Block storage devices, catering primarily to database workloads. Prime examples include Red Hat OpenShift Container Platform logging and monitoring, and PostgreSQL.
- Shared and distributed file system, catering primarily to software development, messaging, and data aggregation workloads. Examples include Jenkins build sources and artifacts, Wordpress uploaded content, Red Hat OpenShift Container Platform registry, and messaging using JBoss AMQ.
- Multicloud object storage, featuring a lightweight S3 API endpoint that can abstract the storage and retrieval of data from multiple cloud object stores.
- On premises object storage, featuring a robust S3 API endpoint that scales to tens of petabytes and billions of objects, primarily targeting data intensive applications. Examples include the storage and access of row, columnar, and semi-structured data with applications like Spark, Presto, Red Hat AMQ Streams (Kafka), and even machine learning frameworks like TensorFlow and Pytorch.

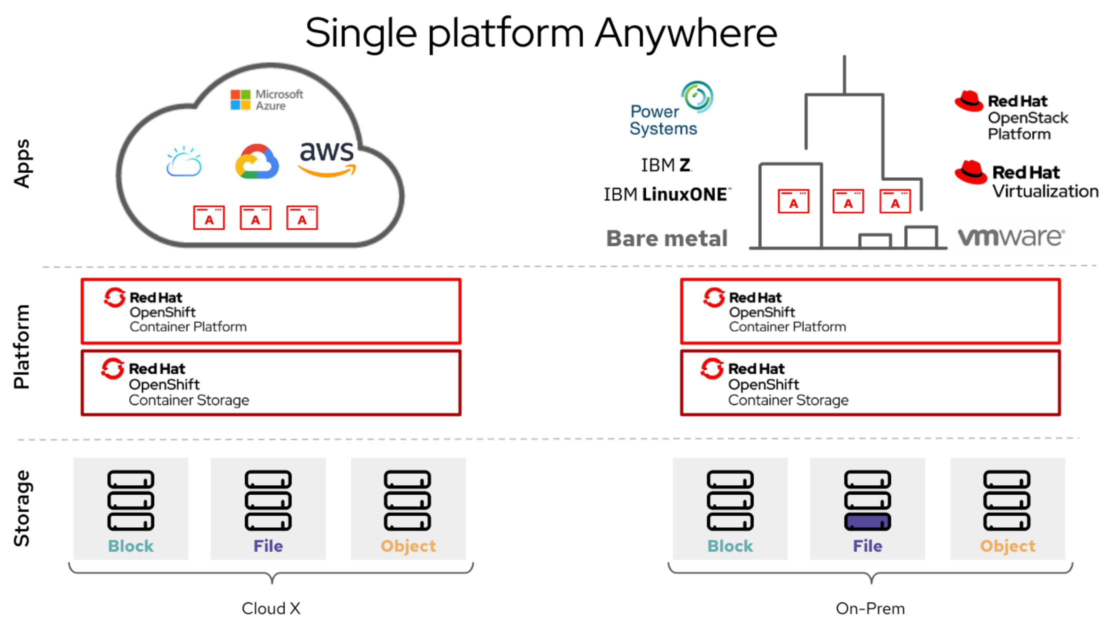
Red Hat OpenShift Container Storage version 4.x integrates a collection of software projects, including:

- Ceph, providing block storage, a shared and distributed file system, and on-premises object storage
- Ceph CSI, to manage provisioning and lifecycle of persistent volumes and claims
- NooBaa, providing a Multicloud Object Gateway
- OpenShift Container Storage, Rook-Ceph, and NooBaa operators to initialize and manage OpenShift Container Storage services.

CHAPTER 2. ARCHITECTURE OF OPENSHIFT CONTAINER STORAGE

Red Hat OpenShift Container Storage provides services for, and can run internally from Red Hat OpenShift Container Platform.

Red Hat OpenShift Container Storage architecture



Red Hat OpenShift Container Storage supports deployment into Red Hat OpenShift Container Platform clusters deployed on Installer Provisioned Infrastructure or User Provisioned Infrastructure. For details about these two approaches, see OpenShift Container Platform - Installation process. To know more about interoperability of components for the Red Hat OpenShift Container Storage and Red Hat OpenShift Container Platform, see the interoperability matrix.

For information about the architecture and lifecycle of OpenShift Container Platform, see OpenShift Container Platform architecture.

NOTE

For IBM Power Systems refer OpenShift Container Platform - Installation process.

2.1. ABOUT OPERATORS

Red Hat OpenShift Container Storage comprises three main operators, which codify administrative tasks and custom resources so that task and resource characteristics can be easily automated. Administrators define the desired end state of the cluster, and the OpenShift Container Storage operators ensure the cluster is either in that state, or approaching that state, with minimal administrator intervention.

OpenShift Container Storage operator

A meta-operator that codifies and enforces the recommendations and requirements of a supported Red Hat OpenShift Container Storage deployment by drawing on other operators in specific, tested ways. This operator provides the storage cluster resource that wraps resources provided by the Rook-Ceph and NooBaa operators.

Rook-Ceph operator

This operator automates the packaging, deployment, management, upgrading, and scaling of persistent storage and file, block, and object services. It creates block and file storage classes for all environments, and creates an object storage class and services object bucket claims made against it in on-premises environments.

Additionally, for internal mode clusters, it provides the Ceph cluster resource, which manages the deployments and services representing the following:

- Object storage daemons (OSDs)
- Monitors (MONs)
- Manager (MGR)
- Metadata servers (MDS)
- Object gateways (RGW) on-premises only

NooBaa operator

This operator automates the packaging, deployment, management, upgrading, and scaling of the Multicloud Object Gateway object service. It creates an object storage class and services object bucket claims made against it.

Additionally, it provides the NooBaa cluster resource, which manages the deployments and services for NooBaa core, database, and endpoint.

2.2. STORAGE CLUSTER DEPLOYMENT APPROACHES

Flexibility is a core tenet of Red Hat OpenShift Container Storage, as evidenced by its growing list of operating modalities. This section provides you with information that will help you to select the most appropriate approach for your environments. OpenShift Container Storage can be deployed either entirely within OpenShift Container Platform (Internal approach) or to make available the services from a cluster running outside of OpenShift Container Platform (External approach).

2.2.1. Internal approach

Deployment of Red Hat OpenShift Container Storage entirely within Red Hat OpenShift Container Platform has all the benefits of operator based deployment and management. Internal-attached device approach in the graphical user interface can be used to deploy Red Hat OpenShift Container Storage in internal mode using the local storage operator and local storage devices.

There are two different deployment modalities available when Red Hat OpenShift Container Storage is running entirely within Red Hat OpenShift Container Platform:

- Simple
- Optimized

Simple deployment

Red Hat OpenShift Container Storage services run co-resident with applications, managed by operators in Red Hat OpenShift Container Platform.

A simple deployment is best for situations where

- Storage requirements are not clear
- OpenShift Container Storage services will run co-resident with applications
- Creating a node instance of a specific size is difficult (bare metal)

In order for Red Hat OpenShift Container Storage to run co-resident with applications, they must have local storage devices, or portable storage devices attached to them dynamically, like EBS volumes on EC2, or vSphere Virtual Volumes on VMware, or SAN volumes dynamically provisioned by PowerVC.

Optimized deployment

OpenShift Container Storage services run on dedicated infrastructure nodes managed by Red Hat OpenShift Container Platform.

An optimized approach is best for situations when:

- Storage requirements are clear
- OpenShift Container Storage services run on dedicated infrastructure nodes
- Creating a node instance of a specific size is easy (Cloud, Virtualized environment, etc.)

2.2.2. External approach

Red Hat OpenShift Container Storage exposes the Red Hat Ceph Storage services running outside of the OpenShift Container Platform cluster as storage classes.

The external approach is best used when:

- Storage requirements are significant (600+ storage devices)
- Multiple OpenShift Container Platform clusters need to consume storage services from a common external cluster.
- Another team (SRE, Storage, etc.) needs to manage the external cluster providing storage services. Possibly pre-existing.

2.3. NODE TYPES

Nodes run the container runtime, as well as services, to ensure that containers are running, and maintain network communication and separation between pods. In OpenShift Container Storage, there are three types of nodes.

Table 2.1. Types of nodes

Node Type	Description
Master	These nodes run processes that expose the Kubernetes API, watch and schedule newly created pods, maintain node health and quantity, and control interaction with underlying cloud providers.

Node Type	Description
Infrastructure (Infra)	Infra nodes run cluster level infrastructure services such as logging, metrics, registry, and routing. These are optional in OpenShift Container Platform clusters. It is recommended to use infra nodes for OpenShift Container Storage in virtualized and cloud environments. To create Infra nodes, you can provision new nodes labeled as **infra**. See How to use dedicated worker nodes for Red Hat OpenShift Container Storage?
Worker	Worker nodes are also known as application nodes since they run applications. When OpenShift Container Storage is deployed in internal mode, a minimal cluster of 3 worker nodes is required, where the nodes are recommended to be spread across three different racks, or availability zones, to ensure availability. In order for OpenShift Container Storage to run on worker nodes, they must either have local storage devices, or portable storage devices attached to them dynamically. When it is deployed in external mode, it runs on multiple nodes to allow rescheduling by K8S on available nodes in case of a failure. Examples of portable storage devices are EBS volumes on EC2, or vSphere Virtual Volumes on VMware, or SAN volumes dynamically provisioned by PowerVC.

NOTE

Nodes that run only storage workloads require a subscription for Red Hat OpenShift Container Storage. Nodes that run other workloads in addition to storage workloads require both Red Hat OpenShift Container Storage and Red Hat OpenShift Container Platform subscriptions. See Chapter 6, *Subscriptions* for more information.

CHAPTER 3. INTERNAL STORAGE SERVICES

Red Hat OpenShift Container Storage service is available for consumption internally to the Red Hat OpenShift Container Platform running on the following infrastructure:

- Amazon Web Services
- Bare metal
- VMware vSphere
- Microsoft Azure
- Google Cloud [Technology Preview]
- Red Hat Virtualization 4.4.x or higher (IPI)
- Red Hat OpenStack 13 or higher (IPI) [Technology Preview]
- IBM Power Systems
- IBM Z and LinuxONE

Ease of deployment and management are the highlights of running OpenShift Container Storage services internally on OpenShift Container Platform. Creation of an internal cluster resource will result in the internal provisioning of the OpenShift Container Storage base services, and make additional storage classes available to applications.

CHAPTER 4. EXTERNAL STORAGE SERVICES

Red Hat OpenShift Container Storage can make services from an external Red Hat Ceph Storage cluster available for consumption through OpenShift Container Platform clusters running on the following platforms:

- VMware vSphere
- Bare Metal
- Red Hat OpenStack platform (Technology preview)

The OpenShift Container Storage operators create and manage services to satisfy persistent volume and object bucket claims against external services. External cluster can serve Block, File and Object storage classes for applications running on OpenShift Container Platform. External clusters are not deployed or managed by operators.

CHAPTER 5. SECURITY CONSIDERATIONS

5.1. FIPS-140-2

The Federal Information Processing Standard Publication 140-2 (FIPS-140-2) is a standard defining a set of security requirements for the use of cryptographic modules. This standard is mandated by law for US government agencies and contractors and is also referenced in other international and industry specific standards.

Red Hat OpenShift Container Storage is now using FIPS validated cryptographic modules as delivered by Red Hat Enterprise Linux OS/CoreOS (RHCOS).

The cryptography modules are currently being processed by Cryptographic Module Validation Program (CMVP) and their state can be seen at Modules in Process List . For more up-to-date information, see the knowledge base article.

NOTE

FIPS mode must be enabled on the OpenShift Container Platform, prior to installing OpenShift Container Storage. OpenShift Container Platform must run on RHCOS nodes, as OpenShift Container Storage deployment on RHEL 7 is not supported for this feature.

For more information, see installing a cluster in FIPS mode and support for FIPS cryptography .

5.2. PROXY ENVIRONMENT

A proxy environment is a production environment that denies direct access to the internet and provides an available HTTP or HTTPS proxy instead. Red Hat Openshift Container Platform is configured to use a proxy by modifying the proxy object for existing clusters or by configuring the proxy settings in the install-config.yaml file for new clusters.

Red Hat supports deployment of Openshift Container Storage versions 4.5 and higher in proxy environments when OpenShift Container Platform has been configured according to configuring the cluster-wide proxy.

5.3. DATA ENCRYPTION OPTIONS

Encryption lets you encode your data to make it impossible to read without the required encryption keys. This mechanism protects the confidentiality of your data in the event of a physical security breach that results in a physical media to escape your custody. Data is encrypted when it is written to the disk, and decrypted when it is read from the disk. Working with encrypted data might incur a small penalty to performance.

Encryption is only supported for new clusters deployed using OpenShift Container Storage 4.6 or higher. An existing encrypted cluster that is not using an external Key Management System (KMS) cannot be migrated to use an external KMS.

Currently, HashiCorp Vault is the only supported KMS for Cluster-wide and Persistent Volume encryptions. With OpenShift Container Storage 4.7.0 and 4.7.1, only HashiCorp Vault Key/Value (KV) secret engine API, version 1 is supported. Starting with OpenShift Container Storage 4.7.2, HashiCorp Vault KV secret engine API, versions 1 and 2 are supported.

IMPORTANT

- KMS is required for Persistent Volume (PV) encryption, and is optional for cluster-wide encryption.
- Red Hat works with the technology partners to provide this documentation as a service to the customers. However, Red Hat does not provide support for the Hashicorp product. For technical assistance with this product, contact Hashicorp.

5.3.1. Cluster-wide encryption

Red Hat OpenShift Container Storage supports cluster-wide encryption (encryption-at-rest) for all the disks and Multicloud Object Gateway operations in the storage cluster. OpenShift Container Storage uses Linux Unified Key System (LUKS) version 2 based encryption with a key size of 512 bits and the **aes-xts-plain64** cipher where each device has a different encryption key. The keys are stored using a Kubernetes secret or an external KMS. Both methods are mutually exclusive and you can not migrate between methods.

Encryption is disabled by default. You can enable encryption for the cluster at the time of deployment. See the deployment guides for more information.

Cluster wide encryption is supported in OpenShift Container Storage 4.6 without Key Management System (KMS), while it is supported in OpenShift Container Storage 4.7 with and without KMS.

Currently, HashiCorp Vault is the only supported KMS. With OpenShift Container Storage 4.7.0 and 4.7.1, only HashiCorp Vault KV secret engine, API version 1 is supported. Starting with OpenShift Container Storage 4.7.2, HashiCorp Vault KV secret engine API, versions 1 and 2 are supported.

IMPORTANT

Red Hat works with the technology partners to provide this documentation as a service to the customers. However, Red Hat does not provide support for the Hashicorp product. For technical assistance with this product, contact Hashicorp.

5.3.2. Storage class encryption

You can encrypt persistent volumes (block only) with storage class encryption using an external Key Management System (KMS) to store device encryption keys. Persistent volume encryption is only available for RADOS Block Device (RBD) persistent volumes. See how to create a storage class with persistent volume encryption.

Storage class encryption is supported in OpenShift Container Storage 4.7 or higher.

CHAPTER 6. SUBSCRIPTIONS

6.1. SUBSCRIPTION OFFERINGS

Red Hat OpenShift Container Storage subscription is based on “core-pairs,” similar to Red Hat OpenShift Container Platform. The Red Hat OpenShift Container Storage 2-core subscription is based on the number of logical cores on the CPUs in the system where OpenShift Container Platform runs.

As with OpenShift Container Platform:

- OpenShift Container Storage subscriptions are stackable to cover larger hosts.
- Cores can be distributed across as many virtual machines (VMs) as needed. For example, ten 2-core subscriptions will provide 20 cores and in case of IBM Power Systems a 2-core subscription at SMT level of 8 will provide 2 cores or 16 vCPUs that can be used across any number of VMs.
- OpenShift Container Storage subscriptions are available with Premium or Standard support.

6.2. DISASTER RECOVERY SUBSCRIPTIONS

Red Hat OpenShift Container Storage does not offer disaster recovery (DR), cold backup, or other subscription types. Any system with OpenShift Container Storage installed, powered-on or powered-off, running workload or not, requires an active subscription.

6.3. CORES VERSUS VCPUS AND HYPERTHREADING

Making a determination about whether or not a particular system consumes one or more cores is currently dependent on whether or not that system has hyperthreading available. Hyperthreading is only a feature of Intel CPUs. Visit the Red Hat Customer Portal to determine whether a particular system supports hyperthreading.

For systems where hyperthreading is enabled and where one hyperthread equates to one visible system core, the calculation of cores is a ratio of 2 cores to 4 vCPUs. Therefore, a 2-core subscription covers 4 vCPUs in a hyperthreaded system. A large virtual machine (VM) might have 8 vCPUs, equating to 4 subscription cores. As subscriptions come in 2-core units, you will need two 2-core subscriptions to cover these 4 cores or 8 vCPUs.

Where hyperthreading is not enabled, and where each visible system core correlates directly to an underlying physical core, the calculation of cores is a ratio of 2 cores to 2 vCPUs.

6.3.1. Cores versus vCPUs and simultaneous multithreading (SMT) for IBM Power Systems

Making a determination about whether or not a particular system consumes one or more cores is currently dependent on the level of simultaneous multithreading configured (SMT). IBM Power systems provide simultaneous multithreading levels of 1, 2, 4 or 8 for each core which correspond to the number of vCPUs as in the table below.

Table 6.1. Different SMT levels and their corresponding vCPUs

SMT level	SMT=1	SMT=2	SMT=4	SMT=8
1 Core	# vCPUs=1	# vCPUs=2	# vCPUs=4	# vCPUs=8

SMT level	SMT=1	SMT=2	SMT=4	SMT=8
2 Cores	# vCPUs=2	# vCPUs=4	# vCPUs=8	# vCPUs=16
4 Cores	# vCPUs=4	# vCPUs=8	# vCPUs=16	# vCPUs=32

For systems where SMT is configured the calculation for the number of cores required for subscription purposes depends on the SMT level. Therefore, a 2-core subscription corresponds to 2 vCPUs on SMT level of 1, and to 4 vCPUs on SMT level of 2, and to 8 vCPUs on SMT level of 4 and to 16 vCPUs on SMT level of 8 as seen in the table above. A large virtual machine (VM) might have 16 vCPUs, which at a SMT level 8 will require a 2 core subscription based on dividing the # of vCPUs by the SMT level (16 vCPUs / 8 for SMT-8 = 2). As subscriptions come in 2-core units, you will need one 2-core subscription to cover these 2 cores or 16 vCPUs.

6.4. SPLITTING CORES

Systems that require an odd number of cores need to consume a full 2-core subscription. For example, a system that is calculated to require only 1 core will end up consuming a full 2-core subscription once it is registered and subscribed.

When a single virtual machine (VM) with 2 vCPUs uses hyperthreading resulting in 1 calculated vCPU, a full 2-core subscription is required; a single 2-core subscription may not be split across two VMs with 2 vCPUs using hyperthreading. See section Cores versus vCPUs and hyperthreading for more information.

It is recommended that virtual instances be sized so that they require an even number of cores.

6.4.1. Shared Processor Pools for IBM Power Systems

IBM Power Systems have a notion of shared processor pools. The processors in a shared processor pool can be shared across the nodes in the cluster. The aggregate compute capacity required for a OpenShift Container Storage should be a multiple of core-pairs.

6.5. SUBSCRIPTION REQUIREMENTS

OpenShift Container Storage components can run on either OpenShift Container Platform worker or infrastructure nodes, for which you can use either Red Hat CoreOS (RHCOS) or Red Hat Enterprise Linux (RHEL) 7 as the host operating system. OpenShift Container Storage subscriptions are required for every OpenShift Container Platform subscribed core with a ratio of 1:1.

When using infrastructure nodes, the rule to subscribe all OpenShift worker node cores for OpenShift Container Storage applies even though they don’t need any OpenShift Container Platform or any OpenShift Container Storage subscriptions. You can use labels to state whether a node is a worker or an infrastructure node.

For more information, see How to use dedicated worker nodes for Red Hat OpenShift Container Storage in the Managing and Allocating Storage Resources guide.

CHAPTER 7. INFRASTRUCTURE REQUIREMENTS

7.1. PLATFORM REQUIREMENTS

Red Hat OpenShift Container Storage 4.8 is supported only on OpenShift Container Platform version 4.8 and its next minor version.

Bug fixes for previous version of OpenShift Container Storage will be released as bug fix versions. For details, see Red Hat OpenShift Container Platform Life Cycle Policy .

For external cluster subscription requirements, see this Red Hat Knowledgebase article .

For a complete list of supported platform versions, see the Red Hat OpenShift Container Storage and Red Hat OpenShift Container Platform interoperability matrix.

7.1.1. Amazon EC2

Supports internal Red Hat Openshift Container Storage clusters only.

An Internal cluster must meet both, storage device requirements and have a storage class providing

- EBS storage via the aws-ebs provisioner

7.1.2. Bare Metal

Supports internal clusters and consuming external clusters.

An Internal cluster must meet both, storage device requirements and have a storage class providing local SSD (NVMe/SATA/SAS, SAN) via the Local Storage Operator.

7.1.3. VMware vSphere

Supports internal clusters and consuming external clusters.

Recommended version: vSphere 6.7, Update 2

See VMware vSphere infrastructure requirements for details.

Additionally, an Internal cluster must meet both, storage device requirements and have a storage class providing either

- vSAN or VMFS datastore via the vsphere-volume provisioner
- VMDK, RDM, or DirectPath storage devices via the Local Storage Operator.

7.1.4. Microsoft Azure

Supports internal Red Hat Openshift Container Storage clusters only.

An Internal cluster must meet both, storage device requirements and have a storage class providing

- Azure disk via the azure-disk provisioner

7.1.5. Google Cloud [Technology Preview]

Supports internal Red Hat Openshift Container Storage clusters only.

An Internal cluster must meet both, storage device requirements and have a storage class providing

- GCE Persistent Disk via the gce-pd provisioner

7.1.6. Red Hat Virtualization Platform

Supports internal Red Hat Openshift Container Storage clusters only.

An Internal cluster must meet both, storage device requirements and have a storage class providing local SSD (NVMe/SATA/SAS, SAN) via the Local Storage Operator.

7.1.7. Red Hat OpenStack Platform [Technology Preview]

Supports internal Red Hat Openshift Container Storage clusters and consuming external clusters.

An Internal cluster must meet both, storage device requirements and have a storage class providing

- standard disk via the Cinder provisioner

7.1.8. IBM Power Systems

Supports internal Red Hat Openshift Container Storage clusters only.

An Internal cluster must meet both, storage device requirements and have a storage class providing local SSD (NVMe/SATA/SAS, SAN) via the Local Storage Operator.

7.1.9. IBM Z and LinuxONE

Supports internal Red Hat Openshift Container Storage clusters only.

An Internal cluster must meet both, storage device requirements and have a storage class providing local SSD (NVMe/SATA/SAS, SAN) via the Local Storage Operator.

7.2. EXTERNAL MODE REQUIREMENT

7.2.1. Red Hat Ceph Storage

Red Hat Ceph Storage (RHCS) version 4.2z1 or later is required. For more information on versions supported, see this knowledge base article on Red Hat Ceph Storage releases and corresponding Ceph package versions.

For instructions regarding how to install a RHCS 4 cluster, see Installation guide.

7.3. RESOURCE REQUIREMENTS

OpenShift Container Storage services consist of an initial set of base services, and can be extended with additional device sets. All of these OpenShift Container Storage services pods are scheduled by kubernetes on OpenShift Container Platform nodes according to resource requirements. Expanding the cluster in multiples of three, one node in each failure domain, is an easy way to satisfy pod placement rules.

IMPORTANT

These requirements relate to OpenShift Container Storage services only, and not to any other services, operators or workloads that are running on these nodes.

Table 7.1. Aggregate available resource requirements for OpenShift Container Storage only

Deployment Mode	Base services	Additional device Set
Internal	• 30 CPU (logical) • 72 GiB memory • 3 storage devices	• 6 CPU (logical) • 15 GiB memory • 3 storage devices
External	• 4 CPU (logical) **IMPORTANT** The external RedHat Ceph Storage cluster uses the normal Ceph sizing rules, for detailed information refer to Hardware Guide. • 16 GiB memory	Not applicable

Example: For a 3 node cluster in an internal mode deployment with a single device set, a minimum of 3 x 10 = 30 units of CPU are required.

For more information, see Chapter 6, *Subscriptions* and CPU units.

For additional guidance with designing your OpenShift Container Storage cluster, see the OCS Sizing Tool.

CPU units

In this section, 1 CPU Unit maps to the Kubernetes concept of 1 CPU unit.

- 1 unit of CPU is equivalent to 1 core for non-hyperthreaded CPUs.
- 2 units of CPU are equivalent to 1 core for hyperthreaded CPUs.
- OpenShift Container Storage core-based subscriptions always come in pairs (2 cores).

Table 7.2. Aggregate minimum resource requirements for IBM Power Systems

Deployment Mode	Base services
Internal	• 48 CPU (logical) • 192 GB memory • 3 storage devices, each with additional 500GB of disk
External	• Not applicable

Example: For a 3 node cluster in an internal-attached devices mode deployment, a minimum of 3 x 16 = 48 units of CPU and 3 x 64 = 192 GB of memory is required.

7.3.1. Resource requirements for IBM Z and LinuxONE infrastructure

OpenShift Container Storage services consist of an initial set of base services, and can be extended with additional device sets.

All of these OpenShift Container Storage services pods are scheduled by kubernetes on OpenShift Container Platform nodes according to the resource requirements.

Expanding the cluster in multiples of three, one node in each failure domain, is an easy way to satisfy the pod placement rules.

Table 7.3. Aggregate available resource requirements for OpenShift Container Storage only (IBM Z and LinuxONE)

Deployment Mode	Base services	Additional device Set	IBM Z and LinuxONE minimum hardware requirements
Internal	• 30 CPU (logical) ○ 3 nodes with 10 CPUs (logical) each • 72 GiB memory • 3 storage devices	• 6 CPU (logical) • 15 GiB memory • 3 storage devices	1 IFL
External	• 4 CPU (logical) • 16 GiB memory	Not applicable	Not applicable

CPU

Is the number of virtual cores defined in the hypervisor, IBM z/VM, Kernel Virtual Machine (KVM), or both.

IFL (Integrated Facility for Linux)

Is the physical core for IBM Z and LinuxONE.

Minimum system environment

- In order to operate a minimal cluster with 1 logical partition (LPAR), one additional IFL is required on top of the 6 IFLs. OpenShift Container Platform consumes these IFLs .

7.3.2. Minimum deployment resource requirements [Technology Preview]

An OpenShift Container Storage cluster will be deployed with minimum configuration when the standard deployment resource requirement is not met.

IMPORTANT

These requirements relate to OpenShift Container Storage services only, and not to any other services, operators or workloads that are running on these nodes.

Table 7.4. Aggregate resource requirements for OpenShift Container Storage only

Deployment Mode	Base services
Internal	• 24 CPU (logical) • 72 GiB memory • 3 storage devices

If you want to add additional device sets, we recommend converting your minimum deployment to standard deployment.

IMPORTANT

Deployment of OpenShift Container Storage with minimum configuration is a Technology Preview feature. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information, see Technology Preview Features Support Scope .

7.3.3. Compact deployment resource requirements

OpenShift Container Storage can be installed on a three-node OpenShift compact bare metal cluster, where all the workloads run on three strong master nodes. There are no worker or storage nodes.

IMPORTANT

These requirements relate to OpenShift Container Storage services only, and not to any other services, operators or workloads that are running on these nodes.

Table 7.5. Aggregate resource requirements for OpenShift Container Storage only

Deployment Mode	Base services	Additional device Set
Internal	● 24 CPU (logical) ● 72 GiB memory ● 3 storage devices	● 6 CPU (logical) ● 15 GiB memory ● 3 storage devices

To configure OpenShift Container Platform on a compact bare metal cluster, see Configuring a three-node cluster and Delivering a Three-node Architecture for Edge Deployments .

7.4. POD PLACEMENT RULES

Kubernetes is responsible for pod placement based on declarative placement rules. The OpenShift Container Storage base service placement rules for Internal cluster can be summarized as follows:

- Nodes are labeled with the **cluster.ocs.openshift.io/openshift-storage** key
- Nodes are sorted into pseudo failure domains if none exist
- Components requiring high availability are spread across failure domains
- A storage device must be accessible in each failure domain

This leads to the requirement that there be at least three nodes, and that nodes be in three distinct rack or zone failure domains in the case of pre-existing topology labels.

For additional device sets, there must be a storage device, and sufficient resources for the pod consuming it, in each of the three failure domains. Manual placement rules can be used to override default placement rules, but generally this approach is only suitable for bare metal deployments.

7.5. STORAGE DEVICE REQUIREMENTS

Use this section to understand the different storage capacity requirements that you can consider when planning internal mode deployments and upgrades. We generally recommend 9 devices or less per node. This recommendation ensures both that nodes stay below cloud provider dynamic storage device attachment limits, and to limit the recovery time after node failures with local storage devices. Expanding the cluster in multiples of three, one node in each failure domain, is an easy way to satisfy pod placement rules.

NOTE

You can expand the storage capacity only in the increment of the capacity selected at the time of installation.

7.5.1. Dynamic storage devices

Red Hat OpenShift Container Storage permits the selection of either 0.5 TiB, 2 TiB or 4 TiB capacities as the request size for dynamic storage device sizes. The number of dynamic storage devices that can run per node is a function of the node size, underlying provisioner limits and resource requirements.

7.5.2. Local storage devices

For local storage deployment, any disk size of 4 TiB or less can be used, and all disks should be of the same size and type. The number of local storage devices that can run per node is a function of the node size and resource requirements. Expanding the cluster in multiples of three, one node in each failure domain, is an easy way to satisfy pod placement rules.

NOTE

Disk partitioning is not supported.

7.5.3. Capacity planning

Always ensure that available storage capacity stays ahead of consumption. Recovery is difficult if available storage capacity is completely exhausted, and requires more intervention than simply adding capacity or deleting or migrating content.

Capacity alerts are issued when cluster storage capacity reaches 75% (near-full) and 85% (full) of total capacity. Always address capacity warnings promptly, and review your storage regularly to ensure that you do not run out of storage space. When you get to 75% (near-full), either free space or expand the cluster. When you get to 85% (full) alert, it indicates that you have run out of storage space completely and cannot free up space using standard commands. At this point, contact Red Hat Customer Support .

The following tables show example node configurations for Red Hat OpenShift Container Storage with dynamic storage devices.

Table 7.6. Example initial configurations with 3 nodes

Storage Device size	Storage Devices per node	Total capacity	Usable storage capacity
0.5 TiB	1	1.5 TiB	0.5 TiB
2 TiB	1	6 TiB	2 TiB
4 TiB	1	12 TiB	4 TiB

Table 7.7. Example of expanded configurations with 30 nodes (N)

Storage Device size (D)	Storage Devices per node (M)	Total capacity (D * M * N)	Usable storage capacity (D*M*N/3)
0.5 TiB	3	45 TiB	15 TiB
2 TiB	6	360 TiB	120 TiB

Storage Device size (D)	Storage Devices per node (M)	Total capacity (D * M * N)	Usable storage capacity (D*M*N/3)
4 TiB	9	1080 TiB	360 TiB

7.6. MULTI NETWORK PLUG-IN (MULTUS) SUPPORT

By default, Red Hat OpenShift Container Storage is configured to use the Red Hat OpenShift Software Defined Network (SDN). In this default configuration the SDN carries the following types of traffic:

- Pod to pod traffic
- Pod to OpenShift Container Storage traffic, known as the OpenShift Container Storage public network traffic
- OpenShift Container Storage replication and rebalancing, known as the OpenShift Container Storage cluster network traffic

However, OpenShift Container Storage 4.8 and later supports as a technology preview the ability to use Multus to improve security and performance by isolating the different types of network traffic.

IMPORTANT

Multus support is a Technology Preview feature that is only supported and has been tested on bare metal and VMWare deployments. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information, see Technology Preview Features Support Scope .

7.6.1. Understanding multiple networks

In Kubernetes, container networking is delegated to networking plug-ins that implement the Container Network Interface (CNI).

OpenShift Container Platform uses the Multus CNI plug-in to allow chaining of CNI plug-ins. During cluster installation, you configure your *default* pod network. The default network handles all ordinary network traffic for the cluster. You can define an *additional network* based on the available CNI plug-ins and attach one or more of these networks to your pods. You can define more than one additional network for your cluster, depending on your needs. This gives you flexibility when you configure pods that deliver network functionality, such as switching or routing.

7.6.1.1. Usage scenarios for an additional network

You can use an additional network in situations where network isolation is needed, including data plane and control plane separation. Isolating network traffic is useful for the following performance and security reasons:

Performance

You can send traffic on two different planes in order to manage how much traffic is along each plane.

Security

You can send sensitive traffic onto a network plane that is managed specifically for security considerations, and you can separate private data that must not be shared between tenants or customers.

All of the pods in the cluster still use the cluster-wide default network to maintain connectivity across the cluster. Every pod has an **eth0** interface that is attached to the cluster-wide pod network. You can view the interfaces for a pod by using the **oc exec -it <pod_name> -- ip a** command. If you add additional network interfaces that use Multus CNI, they are named **net1**, **net2**, …, **netN**.

To attach additional network interfaces to a pod, you must create configurations that define how the interfaces are attached. You specify each interface by using a **NetworkAttachmentDefinition** custom resource (CR). A CNI configuration inside each of these CRs defines how that interface is created.

7.6.2. Segregating storage traffic using Multus

In order to use Multus, before deployment you must create network attachment definitions (NADs) that later will be attached to the cluster. For more information, see:

- Creating network attachment definitions for bare metal
- Creating network attachment definitions for VMware

Using Multus, the following configurations are possible depending on your hardware setup or your VMWare instance network setup:

- Nodes with a dual network interface recommended configuration
  - Segregated storage traffic
    - Configure one interface for OpenShift SDN (pod to pod traffic)
    - Configure one interface for all OpenShift Container Storage traffic
- Nodes with a triple network interface recommended configuration
  - Full traffic segregation
    - Configure one interface for OpenShift SDN (pod to pod traffic)
    - Configure one interface for all pod to OpenShift Container Storage traffic (OpenShift Container Storage public traffic)
    - Configure one interface for all OpenShift Container Storage replication and rebalancing traffic (OpenShift Container Storage cluster traffic)

7.6.3. Recommended network configuration and requirements for a Multus configuration

If you decide to leverage a Multus configuration, the following prerequisites must be met:

- All the nodes used to deploy OpenShift Container Storage must have the same network interface configuration to guarantee a fully functional Multus configuration. The network interfaces names on all nodes must be the same and connected to the same underlying switching mechanism for the Multus public network and the Multus cluster network.
- All the Worker nodes used to deploy applications that leverage OpenShift Container Storage

for persistent storage must have the same network interface configuration to guarantee a fully functional Multus configuration. One of the two interfaces must be the same interface name as that used to configure the Multus public network on the Storage nodes. All Worker network interfaces must be connected to the same underlying switching mechanism as that used for the Storage node's Multus public network.

NOTE

Network software, whether in-switch or SDN, sometimes filters, or drops, ethernet frames that come from unknown MAC addresses. The Cisco parlance for this feature is "Port Security," and in bare metal environments network administrators will need to ensure it, or equivalent functionality, is disabled for the ports that correspond to Multus managed physical network interfaces. The same is true for software defined networks. In the case of VMware, the Security Policy for the Port Group must not be configured to prevent MAC address changes.

NOTE

Only the Storage nodes where OpenShift Container Storage OSDs are running require access to the OpenShift Container Storage cluster network configured via Multus.

See Creating Multus networks for the necessary steps to configure a Multus based configuration on bare metal.

See Creating Multus networks for the necessary steps to configure a Multus based configuration on VMware.

CHAPTER 8. DISASTER RECOVERY

Disaster recovery (DR) helps an organization to recover and resume business critical functions or normal operations when there are disruptions or disasters.

Red Hat OpenShift Container Storage provides two flavors:

- **Regional-DR**: This is a multi-cluster asynchronous replication of storage volumes across two OpenShift Container Storage clusters serving two Openshift Container Platform clusters. Any stateful application, including its stateless counterparts need some preparation prior to deploying the same on a peer cluster.

  IMPORTANT

  This is a developer preview feature and is subject to developer preview support limitations. Developer preview releases are not intended to be run in production environments and are not supported through the Red Hat Customer Portal case management system. If you need assistance with developer preview features, reach out to the ocs-devpreview@redhat.com mailing list and a member of the Red Hat Development Team will assist you as quickly as possible based on their availability and work schedules.

- **Metro-DR (Stretched Cluster - Arbiter)**: In this case, a single cluster is stretched across two zones with a third zone as the location for the arbiter. This is a technology preview feature that is currently intended for deployment in the OpenShift Container Platform on-premises.

  NOTE

  This solution is designed to be deployed where latencies do not exceed 4 milliseconds round-trip time (RTT) between locations. Contact Red Hat Customer Support if you are planning to deploy with higher latencies.

  To use the Arbiter stretch cluster,

  - Must have minimum of five nodes in three zones where:
    - Two nodes per zone are used for each data-center zone and
    - One additional zone with one node is used for arbiter zone (the arbiter can be on a master node).
  - All nodes must be manually labeled with zone labels prior to cluster creation.
    For example, the zones can be labeled as:
    - **topology.kubernetes.io/zone=arbiter** (master or worker node)
    - **topology.kubernetes.io/zone=datacenter1** (minimum two worker nodes)
    - **topology.kubernetes.io/zone=datacenter2** (minimum two worker nodes)

  For more information, see

  - Configuring OpenShift Container Storage for Metro-DR stretch cluster
  - Recovering a Metro-DR stretch cluster

IMPORTANT

This is a technology preview feature and is available only for deployment using local storage devices. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.

For more information, see Technology Preview Features Support Scope .

CHAPTER 9. DISCONNECTED ENVIRONMENT

Disconnected environment is a network restricted environment where Operator Lifecycle Manager (OLM) cannot access the default Operator Hub and image registries, which require Internet connectivity.

Red Hat supports deployment of OpenShift Container Storage in disconnected environments where OpenShift Container Platform is installed in restricted networks.

NOTE

When you install OpenShift Container Storage in a restricted network environment, apply a custom Network Time Protocol (NTP) configuration to the nodes, because by default, internet connectivity is assumed in OpenShift Container Platform and chronyd is configured to use ***.rhel.pool.ntp.org** servers. See Red Hat Knowledgebase article and Configuring chrony time service for more details.

For more information, see Operators guide to using Operator Lifecycle Manager on restricted networks .

CHAPTER 10. UNSUPPORTED FEATURES FOR IBM POWER SYSTEMS AND IBM Z INFRASTRUCTURE

- FIPS is not supported on OpenShift Container Storage 4.8 on IBM Power Systems and IBM Z infrastructure as FIPS itself is not supported on RHEL.
- External mode is not applicable for IBM Power Systems and IBM Z infrastructure.
- Cluster wide encryption is not supported on IBM Z infrastructure.
- Storage data encryption during deployment is not supported on IBM Z infrastructure.
- OpenShift Container Storage 4.8 on IBM Power Systems does not support dynamic storage devices.
- OpenShift Container Storage 4.8 on IBM Power Systems and IBM Z infrastructure does not support Disaster Recovery.
- Compact deployment is not applicable for IBM Power Systems and IBM Z infrastructure.
- OpenShift Container Storage 4.8 on IBM Power Systems and IBM Z infrastructure does not support Multus.

CHAPTER 11. NEXT STEPS

To start deploying your OpenShift Container Storage, you can use the internal mode within OpenShift Container Platform or use external mode to make available services from a cluster running outside of OpenShift Container Platform.

Depending on your requirement, go to the respective deployment guides.

Internal mode

- Deploying OpenShift Container Storage using Amazon web services
- Deploying OpenShift Container Storage using Bare Metal
- Deploying OpenShift Container Storage using VMWare vSphere
- Deploying OpenShift Container Storage using Microsoft Azure
- Deploying OpenShift Container Storage using Google Cloud [Technology Preview]
- Deploying OpenShift Container Storage using Red Hat OpenStack Platform [Technology Preview]
- Deploying OpenShift Container Storage using Red Hat Virtualization Platform
- Deploying OpenShift Container Storage on IBM Power Systems
- Deploying OpenShift Container Storage on IBM Z Infrastructure

External mode

- Deploying OpenShift Container Storage in external mode