IBM Storage Scale Container Native Storage Access 5.2.0

Tables of Contents

Overview

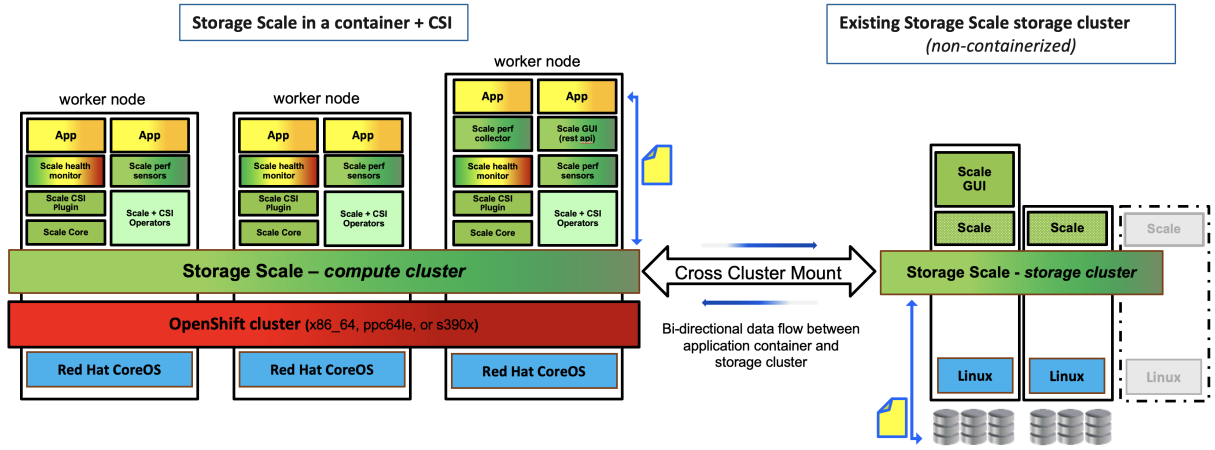
IBM Storage Scale container native is a containerized version of IBM Storage Scale.

IBM Storage Scale is a clustered file system that provides concurrent access to a single file system or set of file systems from multiple nodes. The nodes can be SAN attached, network attached, a mixture of SAN attached, and network attached, or in a shared-nothing cluster configuration. IBM Storage Scale enables high-performance access to this common set of data to support a scale-out solution or to provide a high availability platform. For more information about IBM Storage Scale features, see Product overview in IBM Storage Scale documentation.

IBM Storage Scale container native allows the deployment of IBM Storage Scale in a Red Hat OpenShift cluster. IBM Storage Scale container native deployment makes persistent data store available via either IBM Storage Scale remote filesystem or local filesystem. This data store can be used by the applications through the IBM Storage Scale CSI driver by using Persistent Volumes (PVs). For more information, see IBM Storage Scale Container Storage Interface Driver in IBM Storage Scale CSI documentation.

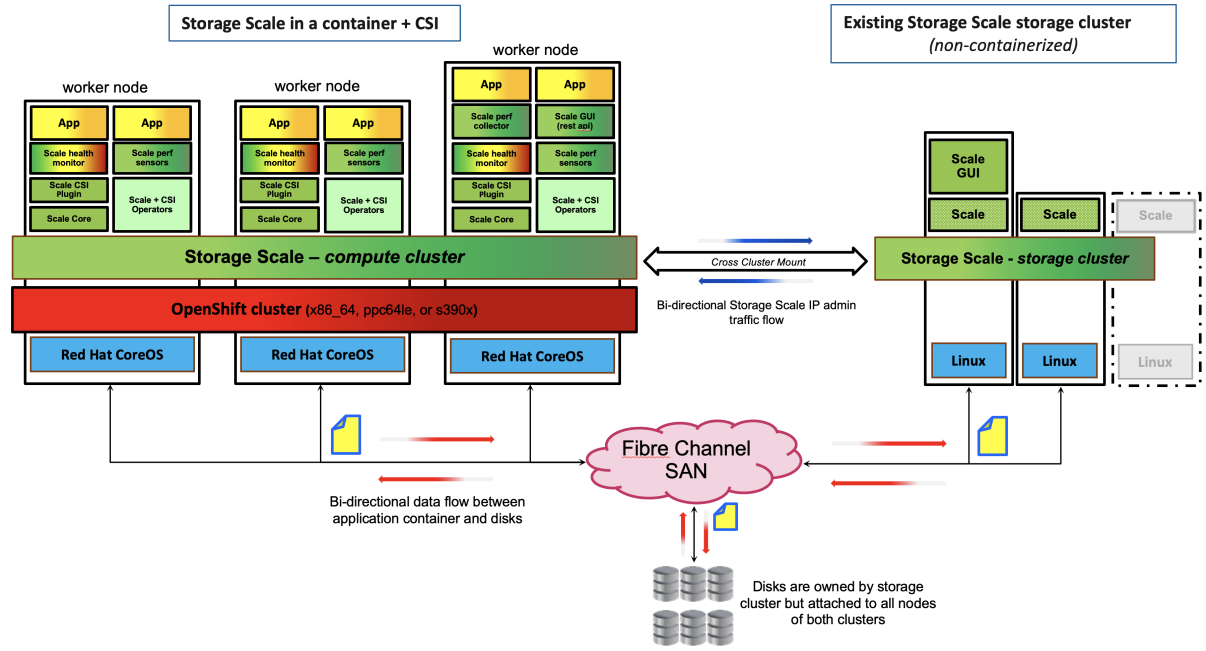
Remote mount use case

Figure 1. Remote mount



Direct attach use case

Figure 2. Direct attach

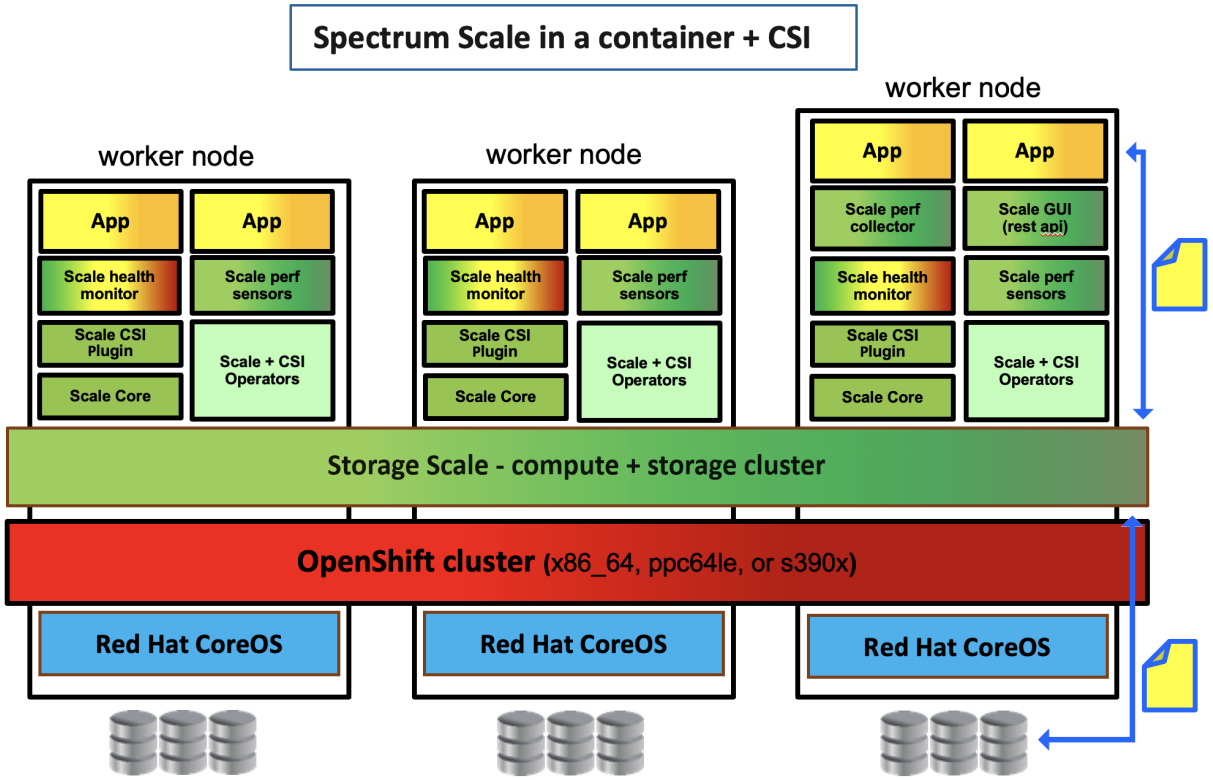


For more information about direct storage attachment, see Deployment Considerations

Local file system use case

Technology preview features are not supported for use within production environments. Use with nonproduction workloads, in demo or proof of concept environments only. IBM production service level agreements (SLAs) are not supported. Technology preview features may not be functionally complete. The purpose of technology preview features is to give access to new and exciting technologies, enabling customers to test functionality and provide feedback during the development process. The existence of a technology preview feature does not mean a future release is guaranteed. Feedback is welcome and encouraged.

Figure 3. Local file system



What's new?

The following enhancements are made in this release:

- Container Storage Interface (CSI) 2.11.1. For more information, see Summary of changes in IBM Storage Scale CSI Documentation
- Support for Red Hat OpenShift 4.13, 4.14 and 4.15 with IBM Storage Scale container native v5.2.0.x
- Ability to configure resource limits for core pods
  - CPU and memory resource limits for core pods have different default values. For more information, see Pod memory requests and limits
  - CPU and memory resource limits for core pods can be modified. For more information, see Cluster custom resource description
- Support GUI high availability configuration for CSI functionality
- Removed pod disruption budget that blocked core pod evictions
  - Instead, an eviction webhook now blocks core pod evictions and it provides a useful response
- Enhanced status reporting for daemon
  - Node cordon and drain information displayed
  - Display detailed reasons for core pod deletions
  - Display reasons why the operator cannot delete a core pod that is scheduled for deletion
- Technology preview section added in our documentation. For more information, see Technology preview

Supported features

IBM Storage Scale container native with Red Hat OpenShift Container Platform supports the following features:

- IBM Storage Scale node labels to establish node affinity.
- Automated client-only cluster creation
- Automated remote file system mount for IBM Storage Scale Storage cluster
- Automated local file system creation that uses disks or volumes which are attached to Red Hat OpenShift nodes in a shared nothing configuration (as technology preview)
- Integrated IBM Storage Scale Container Storage Interface (CSI) driver for application persistent storage
- Automated deployment of IBM Storage Scale Container Storage Interface (CSI) driver
- IBM Storage Scale container native client cluster node expansion on Red Hat OpenShift Container Platform
- Cluster monitoring by using Red Hat OpenShift Container Platform liveness and readiness probe
- Call home
- Performance data collection
- Storage cluster encryption
- Rolling upgrade
- Automated IBM Storage Scale performance monitoring bridge for Grafana
- File audit logging (FAL)
- Compression
- Quotas on the storage cluster
- ACLs on the storage cluster
- ILM support on the storage cluster
- File clones on the storage cluster
- Snapshots on the storage cluster
- TCP/IP network connectivity among cluster nodes
- Direct storage attachment on s390x, x86_64, and ppc64le (Power) servers
- Automatic quorum selection is Kubernetes topology aware.

Limitations

- IBM Storage Scale container native currently supports remote mount of the file system and local file system in a shared nothing configuration
- Deployment of IBM Storage Scale pods on master nodes is not supported except for "compact" Red Hat OpenShift clusters.
- Deployment of IBM Storage Scale pods on RHEL worker nodes is not supported.
- Deployment of IBM Storage Scale pods on nodes with ARM CPUs is not supported.
- Single node Red Hat OpenShift clusters are not supported.

Scalability constraints

Overall

Table 1. Maximum Capacity Specification

Description	Max supported
Number of worker nodes	128

Remote file system

Table 2. Maximum Capacity Specification for Remote File Systems

Description	Max supported
Number of remote clusters	4
Number of remote file systems	16

Local file system

Table 3. Maximum Capacity Specification for Local File Systems

Description	Max supported
Number of local file systems	3
Number of storage nodes	48
Maximum disk size	4 TiB
Number of disks per local file system	128
Number of disks per node	16

Description	Max supported
Number of disks for all local file systems	128
Maximum raw file system capacity	384 TiB
Maximum usable file system capacity	128 TiB

Technology preview

Technology preview features are not supported for use within production environments. Use with nonproduction workloads, in demo or proof of concept environments only. IBM production service level agreements (SLAs) are not supported. Technology preview features may not be functionally complete. The purpose of technology preview features is to give access to new and exciting technologies, enabling customers to test functionality and provide feedback during the development process. The existence of a technology preview feature does not mean a future release is guaranteed. Feedback is welcome and encouraged.

Current features in technology preview

- Local Disk attachment for creation of a local file system

  Ability to use local disks of the Red Hat OpenShift Nodes as file system storage in a shared nothing configuration. For more information, see:

  - local file system
  - local disks
- Infiniband Remote Direct Memory Access (RDMA)

Planning

The planning for IBM Storage Scale container native includes the following topics:

- Prerequisites
- Hardware requirements
- Software requirements
- Deployment considerations
- Network and firewall requirements
- Container Network Interface (CNI) configuration
- IBM Storage Scale container native and SELinux
- Roles and personas
- Container images
- Air-gapped installs

Prerequisites

The planning process to install the IBM Storage Scale on Red Hat OpenShift consists of many steps.

These steps are expanded on previous steps and it is critical to follow the sequence defined in the following sections. Before you begin installation, several things need to be considered. The list of questions helps you to be prepared for the procedure.

- What version of Red Hat OpenShift Container Platform do you need?
- What are the hardware requirements?
- Are the necessary ports opened?
- Is the Red Hat OpenShift Container Platform cluster in a restricted network environment?
- What is the minimum level of IBM Storage Scale that is needed on the storage cluster?

Preparations for deploying the IBM Storage Scale container native cluster

The following section summarizes the prerequisites for deploying the IBM Storage Scale container native cluster. All steps are explained in more detail in the following chapters.

- Validate that the Red Hat OpenShift cluster, or the node from where you are managing the Red Hat OpenShift cluster, has access to the manifest files in the IBM Storage Scale container native repository of GitHub. For more information, see IBM Storage Scale container native repository on GitHub.

GitHub YAML manifests are inline with the Installation steps and are either accessed directly or pulled through **curl** through an existing internet connection. If an air-gapped environment is running, the manifest files must be made locally available for use.

- Validate and apply the configuration to the Red Hat OpenShift installation settings.
- Obtain an IBM Cloud Container Registry entitlement key to access the container images of the IBM Storage Scale container native.
- If you are in a restricted network environment, then mirror the container images of the IBM Storage Scale container native into a site-managed private image registry.
- Create an Red Hat OpenShift global pull secret for the image registry that the cluster uses (either IBM Cloud Container Registry or private image registry).

Deploying the IBM Storage Scale container native cluster with mounting a file system from a remote storage cluster

To deploy a cluster, complete the following steps:

1. Create the IBM Storage Scale container native and IBM Storage Scale CSI operators by deploying the operator installer file: **install.yaml**.
2. Download the sample **cluster.yaml** CR, make necessary change, and apply to the cluster.
   - Specify the IBM Storage Scale Edition in the license field.
   - Configure appropriate node selectors for the IBM Storage Scale container native deployment.
   - Configure host aliases or configure the proper DNS for your environment to allow for communication to the storage cluster.
   - Configure Ephemeral Port Range, if necessary.
   - Enable the optional Grafana Bridge.
3. Download the sample **callhome.yaml** CR, make necessary changes, and apply to the cluster.
4. Download the sample **remotecluster.yaml** CR, make necessary changes, and apply to the cluster.
5. Download the sample **filesystem.remote.yaml** CR, make necessary changes, and apply to the cluster.
6. If accessing encrypted data on the storage cluster, download the sample **encryptionconfig.remote.yaml** CR, make necessary changes, and apply to the cluster.
7. Complete the storage cluster configuration.

   a. Create a GUI user on the storage cluster with the **ContainerOperator** role.

   b. Create a GUI user on the storage cluster with the **CsiAdmin** role.

   c. Configure CSI prerequisites on the storage cluster.
8. Use the storage cluster GUI user credentials for **ContainerOperator** GUI user to create a secret in the **ibm-spectrum-scale** namespace.
9. Use the storage cluster GUI user credentials for **CsiAdmin** GUI user to create a secret in the **ibm-spectrum-scale-csi** namespace.
10. Create a storage class to create volumes to use with your container native cluster.

Deploying the IBM Storage Scale container native cluster with using a local file system

To deploy a cluster, complete the following steps:

1. Create the IBM Storage Scale container native and IBM Storage Scale CSI operators by deploying the operator installer file: **install.yaml**.
2. Download the sample **cluster.yaml** CR, make necessary change, and apply to the cluster.
   - Specify the IBM Storage Scale Edition in the license field.
   - Configure appropriate node selectors for the IBM Storage Scale container native deployment.
   - Enable the optional Grafana Bridge.
3. Download the sample **callhome.yaml** CR, make necessary changes, and apply to the cluster.
4. Download the sample **localdisk.yaml** CR, make necessary changes, and apply to the cluster.
5. Download the sample **filesystem.local.yaml** CR, make necessary changes, and apply to the cluster.

6. Create a storage class to create volumes to use with your container native cluster.

Hardware requirements

The following sections describe hardware requirements to consider to deploy an IBM Storage Scale container native.

Network

- All nodes in the compute cluster must be able to communicate with all nodes in the storage cluster.
- A minimum of 10 Gb network is needed but 40 - 100 Gb is recommended.
- Remote Direct Memory Access (RDMA) for InfiniBand is supported as technology preview.
- RDMA over Converged Ethernet (RoCE) for Ethernet is not supported.

Worker node requirements

IBM Storage Scale container native supports x86_64, ppc64le, and s390x CPU architectures. All nodes in the Red Hat OpenShift cluster must have the same architecture. The ARM architecture is not supported.

Starting with IBM Storage Scale container native v5.1.9, the UBI 9 base image is used and it is based on RHEL 9. And as RHEL 9 no longer supports Power8, IBM Storage Scale container native also no longer supports Power8. For more information, see Minimum IBM Power requirements.

IBM Storage Scale container native deploys several pods in the cluster. The following table shows the resource consumption of those pods.

Table 1. Hardware requirements

Pod	Where deployed	CPU request	Memory request	Storage	Notes
core (created with the k8s Node shortname)	Nodes that are labeled with nodeSelector in cluster CR	Minimum 1000mCPU. The sample CR sets 2000mCPU.	Minimum 4GiB (2 GiB on s390). The sample CR sets 4GiB for client role and 8GiB for storage role.	config in `/var` (~25 GiB)	This is the pod that provides the file system service for the node. It must be deployed on all nodes where PVs are accessed from application pods. The CPU and memory requests and limits can be customized in the cluster CR.
operator	Single Node	500mCPU	200 MiB		The controller runtime that manages all custom resources
gui	Two Nodes	630mCPU	1.25 GiB	Local PV for DB	The graphical user interface and ReST API
pmcollector	Two Nodes	120 mCPU	3-7GiB depending on cluster size	Local PV for DB	The performance collector database
grafana-bridge	Single Node	100mCPU	1GiB		The bridge for accessing pmcollector from grafana

The shown values are requests. For more information, see Kubernetes resource management. The limits are higher. This means that for CPU the pods might have bursts with more CPU usage at times where the CPU has free cycles. For memory the pods should not exceed their request significantly. By default the core pods take 25% of the worker node capacity. 25% might be oversized in many applications. For more information about configuring the requests for both CPU and memory, see Cluster custom resource.

```
* Allocating more resources to IBM Storage Scale results in better storage performance.
* Allocating less allows more applications to be scheduled on a node.
```

For CPU, allocation can be reduced if the core fs pods consistently stay under the request. The CPU allocation can be monitored on the Red Hat OpenShift console. When going too low the file system daemon might starve on CPU cycles that destabilizes the whole cluster and can result in outages. For memory, there is no real monitoring, allocating more results in more data is cached which can boost performance. But this will be only seen indirectly by observing application performance.

The CPU request can be reduced under the 1000mCPU minimum. Your system might run fine with, for example, a 100mCPU. But, if a service ticket is opened for an issue that might be in any way that is related to this setting you will be asked to go up to 1000mCPU. The ticket is accepted only if the problem keeps showing up. Examples for related issues are, node expels, lag on PV creation in CSI, slow policy runs, bad performance, long servers, and so on. The Red Hat OpenShift console reports all worker nodes as overcommitted. The reason is that the CPU and memory limits of the pods add up to more than then total capacity of the node. This is normal and no reason for concern. Pods are scheduled based on their requests and the scheduler ensures that nodes will not be overcommitted in this regard. Higher limits allow pods to use resources that are free at the moment, but only the requested resources are guaranteed to them by Kubernetes. For more information about pod scheduling, see Kubernetes resource management

Pod memory request and limits

The following table shows the memory requests and limits configuration for each pod created.

Table 2. Pod memory requests and limits

Namespace	Pod name	Container	Memory request \	limit	CPU request \	limit
`ibm-spectrum-scale`	Grafana Bridge	`grafanabridge`	1000Mi \	4000Mi	100 m \	500 m
`ibm-spectrum-scale`	GUI	`liberty`	750Mi \	1000Mi	500 m \	2
`ibm-spectrum-scale`	GUI	`postgres`	250Mi \	500Mi	100 m \	1
`ibm-spectrum-scale`	GUI	`sysmon`	200Mi \	500Mi	20 m \	100 m
`ibm-spectrum-scale`	GUI	`logs`	20Mi \	60Mi	10 m \	100 m
`ibm-spectrum-scale`	PM Collector	`pmcollector`	5000Mi \	10000Mi	100 m \	500 m
`ibm-spectrum-scale`	PM Collector	`sysmon`	200Mi \	500Mi	20 m \	100 m
`ibm-spectrum-scale`	Core	`gpfs`	4Gi \	8Gi[1]	2 \	4[2]
			8Gi \	16Gi[3]	2 \	4[4]
`ibm-spectrum-scale`	Core	`logs`	20Mi \	60Mi	10 m \	100 m
`ibm-spectrum-scale-operator`	controller-manager	`manager`	200Mi \	400Mi	500 m \	1500 m
`ibm-spectrum-scale-dns`	CoreDNS	`coredns`	70Mi \	140Mi	50 m \	100 m

For more information, see Requests and limits in the Kubernetes documentation.

Local file system

This feature is available as a technology preview. Technology preview features are not supported for use within production environments. Use with nonproduction workloads, in demo or proof of concept environments only. IBM production service level agreements (SLAs) are not supported. Technology preview features can not be functionally complete. The purpose of technology preview features is to give access to new and exciting technologies, enabling customers to test functionality and provide feedback during the development process. The existence of a technology preview feature does not mean that a future release is guaranteed. Feedback is welcome and encouraged.

The Red Hat OpenShift nodes can provide disks or volumes to be used as local storage for creating PVs. These disks or volumes serve as storage for a local file system. A disk or volume can be a single drive, a partition of a single drive, or a volume from a RAID controller. Disks must be attached to the Red Hat OpenShift nodes as a shared nothing configuration. Shared disks or SANs are not supported.

Red Hat OpenShift nodes that provide disks or volumes are called storage nodes. At least 3 storage nodes must exist.

The local file system uses 3-way replication, which causes the amount of usable storage to be a third of the total capacity of the disks or volumes. Each data block is written to three disks that are located in different failure groups. Per default, a failure group consists of the disks or volumes of one storage node. If Kubernetes zones are defined for the storage nodes, a separate failure group is assigned to all nodes within a zone. In this case, at least 3 failure zones must exist.

Having equal disk or volume capacity within each zone allows optimal usage of the storage. Having disks or volumes of the same size and performance characteristics is beneficial.

Disks or volumes with a maximum capacity of 4 TiB are supported. VMware[5] thin provisioned virtual disks are not supported. For more information, see the "Disk Questions" section in IBM Storage Scale FAQ.

---

1. The memory limit is set to double of request memory by default. It can be set to another custom value into the **`Cluster`** custom resource. The new custom value must be at least twice as the memory request value.↩
2. The CPU limit is set to double of the CPUs of request by default. This can also be modified into the **`Cluster`** custom resource, but the new custom value must be at least twice as the CPU requested value.↩
3. For core pods with the **`storage`** role we request **`8Gi`** of memory (instead of only **`4Gi`** as requested for core pods with the **`client`** role). The **`storage`** role is used for core pods that run on Red Hat OpenShift nodes that provide disks or volumes for local file systems.↩
4. For core pods with the **`storage`** role, the CPU limit is set to double of the CPUs of request by default. The **`storage`** role is used for core pods that run on Red Hat OpenShift nodes that provide disks or volumes for local file systems.↩
5. VMware, the VMware logo, VMware Cloud Foundation, VMware Cloud Foundation Service, VMware vCenter Server, and VMware vSphere are registered trademarks or trademarks of VMware, Inc. or its subsidiaries in the United States and/or other jurisdictions. ↩

Software requirements

Use the following table to determine the software requirement levels for each release.

Table 1. Software Requirements

IBM Storage Scale Container Native	IBM Storage Scale CSI	Architecture	IBM Storage Scale Remote Storage cluster level	File system version cannot be newer than	OpenShift Container Platform level	Red Hat CoreOS	UBI level
5.1.5.0	2.7.0	x86_64, ppc64le, s390x	5.1.3.0+	29.00	4.9, 4.10, 4.11	4.9, 4.10, 4.11	8.6
5.1.6.0	2.8.0	x86_64, ppc64le, s390x	5.1.3.0+	30.00	4.9, 4.10, 4.11	4.9, 4.10, 4.11	8.7
5.1.7.0	2.9.0	x86_64, ppc64le, s390x	5.1.3.0+	31.00	4.10, 4.11, 4.12	4.10, 4.11, 4.12	8.7
5.1.9.1	2.10.0	x86_64, ppc64le, s390x	5.1.3.0+	33.00	4.12, 4.13, 4.14	4.12, 4.13, 4.14	9.2
5.1.9.3	2.10.1	x86_64, ppc64le, s390x	5.1.3.0+	33.00	4.12, 4.13, 4.14	4.12, 4.13, 4.14	9.3
5.1.9.4	2.10.2	x86_64, ppc64le, s390x	5.1.3.0+	33.00	4.12, 4.13, 4.14	4.12, 4.13, 4.14	9.3
5.2.0.0	2.11.0	x86_64, ppc64le, s390x	5.1.3.0+	34.00	4.13, 4.14, 4.15	4.13, 4.14, 4.15	9.3
5.2.0.1	2.11.1	x86_64, ppc64le, s390x	5.1.3.0+	34.00	4.13, 4.14, 4.15	4.13, 4.14, 4.15	9.3

The storage cluster is supported to be down-level from the IBM Storage Scale container native cluster, but it is ideal that the versions match. CSI functions is highly dependent upon the IBM Storage Scale release, file system level, and version, which is installed on the storage cluster. If the storage cluster is running an earlier version, some functions might not be available. For more information about the minimum levels required for specific CSI functions, see *Table 1 in Hardware and Software Requirements* in IBM Storage Scale CSI documentation. For more information about compatibility and software matrix, see Section 17.3 in the IBM Storage Scale FAQ documentation.

IBM Storage Scale container native doesn't support secure boot, the secure boot feature needs to be disabled for Red Hat OpenShift Container Platform nodes

IBM Storage Scale Container Storage Interface (CSI)

IBM Storage Scale Container Storage Interface (CSI) will be installed along with IBM Storage Scale container native.

Storage cluster

- The storage cluster must be at IBM Storage Scale 5.1.3.0 or later.
- To take advantage of all functions provided by IBM Storage Scale Container Storage Interface Driver (CSI), the storage cluster is recommended to be at the latest versions configured with the latest file system level format. If using earlier versions, functional restrictions may apply. For more information, see Hardware and software requirements in IBM Storage Scale CSI documentation.

  For more information, see Upgrading multi-cluster environments in IBM Storage Scale documentation.

- Determine whether the storage cluster is running a GUI high availability configuration, for example, having 2 or more GUI nodes installed.

  On the storage cluster, issue the following command. If two or more GUI nodes are displayed, then the storage cluster is running a GUI high availability configuration.

  ```
  /usr/lpp/mmfs/gui/cli/lsnode
  ```

  If the storage cluster is running a GUI high availability configuration, ensure that the storage cluster is running IBM Storage Scale 5.1.6.1 or higher before use with IBM Storage Scale container native and IBM Storage Scale CSI.

  For more information, see Ensuring high availability of the GUI service in IBM Storage Scale documentation.

- Enable the **--auto-inode-limit** parameter on the remotely mounted file system.

  For more information about **auto-inode-limit** parameter, see mmchfs command in IBM Storage Scale documentation.

  The **--auto-inode-limit** option is available only with file system format level 28.00 or later.

- Encrypted file systems are supported. Configure the EncryptionConfig custom resource with the necessary key client and key server information. For more information, see Encryption.

External container images

There are some external container images that are required to run IBM Storage Scale container native. If running in an air gap environment, these images are required for successful deployment. For more information, see Container images.

Auxiliary helper applications

- `curl` is used to retrieve some files required for the IBM Storage Scale container native installation.
- `jq` 1.5+ is used to help parse and format JSON output.

Deployment considerations

Before deployment, make sure that you are aware of the Red Hat OpenShift version, cluster network, persistent storage, and the IBM Storage Scale storage cluster considerations.

Red Hat OpenShift cluster considerations

The following list includes the Red Hat OpenShift cluster considerations:

- A minimum configuration of three master nodes and three worker nodes, with a maximum of 128 worker nodes is required.
- Deploying IBM Storage Scale pods on master nodes is not supported. An exception is in a "compact" cluster configuration. For more information, see Master node configuration (compact cluster).
- Single node Red Hat OpenShift clusters are not supported. Instead, access data on an IBM Storage Scale storage cluster through NFS.
- Red Hat Enterprise Linux™ CoreOS (RHCOS) restricts new file system mounts to the `/mnt` subtree. IBM Storage Scale mounts any file system under `/mnt` on the Red Hat OpenShift cluster regardless of the default mount point that is defined on the storage cluster.

Red Hat OpenShift cluster network considerations

The IBM Storage Scale container native comes with a collection of different pods. A subset of these pods can be considered regular pods that behave like typical application pods. Those pods are the operator, the GUI pods, and the performance data collector pods. The exception is what is referred to as the "core pods" as they provide the actual file system services. The core pods are not deployed by the Kubernetes scheduler through a regular DaemonSet. Instead, the IBM Storage Scale container native operator handles the management of those pods.

- The file system daemon inside the core pods requires a static IP address for communication between daemons on different nodes.
- All core pods must be able to communicate with each other through the chosen network.

There are two network configurations that can be employed: host network or container network interface (CNI) network. Only one network configuration can be chosen.

Host network

By default, the IBM Storage Scale pods use the host network. This is the simplest configuration but has some disadvantages:

- Using the host network breaks the network isolation that usually comes with containers. For example, any network port opened by IBM Storage Scale may conflict with a network port opened by another component on the host.
- Security features, like network policies, are not available for the host network.
- If the node has multiple network adapters, there is no way to select a different adapter. The host network always uses the network adapter that the worker node IP is assigned to.

Container Network Interface (CNI) network

As an alternative to host network, IBM Storage Scale can use a CNI network. There is more configuration effort to set up the CNI:

- In this configuration, core pods have an IP address on the usual Red Hat OpenShift SDN and another one on the CNI network.
  - Red Hat OpenShift SDN is used for communication with other pods.
  - CNI network is used for communication between file system daemons, both inter-cluster and with the storage cluster.
- If the node is equipped with high-speed network, the CNI interface should use the high-speed network.
  - This is the daemon network where the file system I/O runs on. High bandwidth and low latency are highly beneficial for performance.
- The CNI network will be used exclusively by IBM Storage Scale and eliminates the potential for port conflicts with other components.
- Security features like network policies work on **`MACVLAN`** CNIs.
- The DNS must be configured properly to allow the worker nodes the ability to resolve the storage cluster nodes.
  - For more information, see Host aliases.

Advanced features of **`SR-IOV`** type CNIs, such as RDMA and GPUdirect, are not yet supported.

For more information about configuring CNI with IBM Storage Scale container native, see Container network interface (CNI) configuration.

Red Hat OpenShift cluster persistent storage considerations

The following list includes the Red Hat OpenShift cluster persistent storage considerations:

- The IBM Storage Scale pods use host path mounts to store IBM Storage Scale cluster metadata and various logs.
- The IBM Storage Scale container native operator creates two local PersistentVolumes (PVs) on two eligible worker nodes. At least 25 GB of free space must be available in the file system that contains the `/var` directory on all eligible worker nodes to avoid potential failures during the deployment. These PVs are created with the ReadWriteOnce (RWO) access mode.
- Both the host path mounts and local PVs are not automatically cleaned up when you delete the associated IBM Storage Scale container native cluster. You must manually clean these up. For more information about cleaning up the persistent storage, see Cleaning up the worker nodes and Cleaning up IBM Storage Scale container native
- IBM Storage Scale container native does not support the use of dynamically created or pre-created PVs.

Considerations for enterprise grade image registry

The following list includes the considerations for enterprise grade image registry:

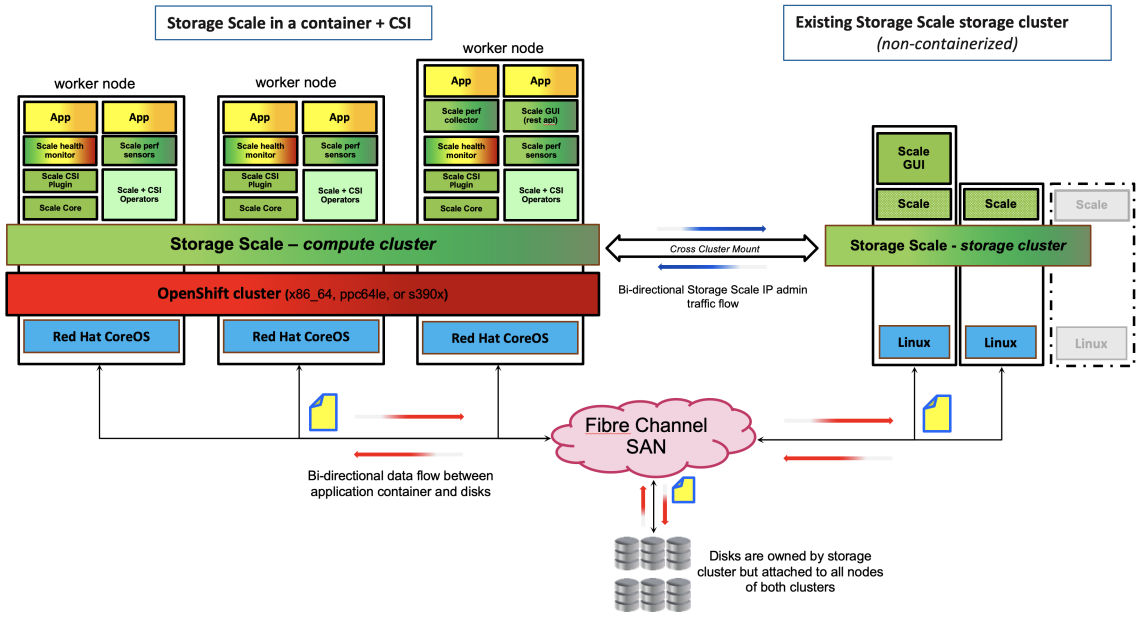
- In a restricted network environment where the Red Hat OpenShift Container Platform cluster cannot pull IBM Storage Scale images from the IBM Container Repository, images must be mirrored to a production grade enterprise image registry that the Red Hat OpenShift Container Platform cluster can access.
- In a restricted network environment, there must be a node that can communicate externally and also with the target Red Hat OpenShift Container Platform cluster.
- Any registry that is used for hosting the container images of IBM Storage Scale container native must not be accessible to external users. Also, it must be restricted to the service account used for IBM Storage Scale container native management. All users and machines that are accessing these container images must be authorized per the IBM Storage Scale license agreement.

Considerations for direct storage attachment

The following list includes the considerations for direct storage attachment:

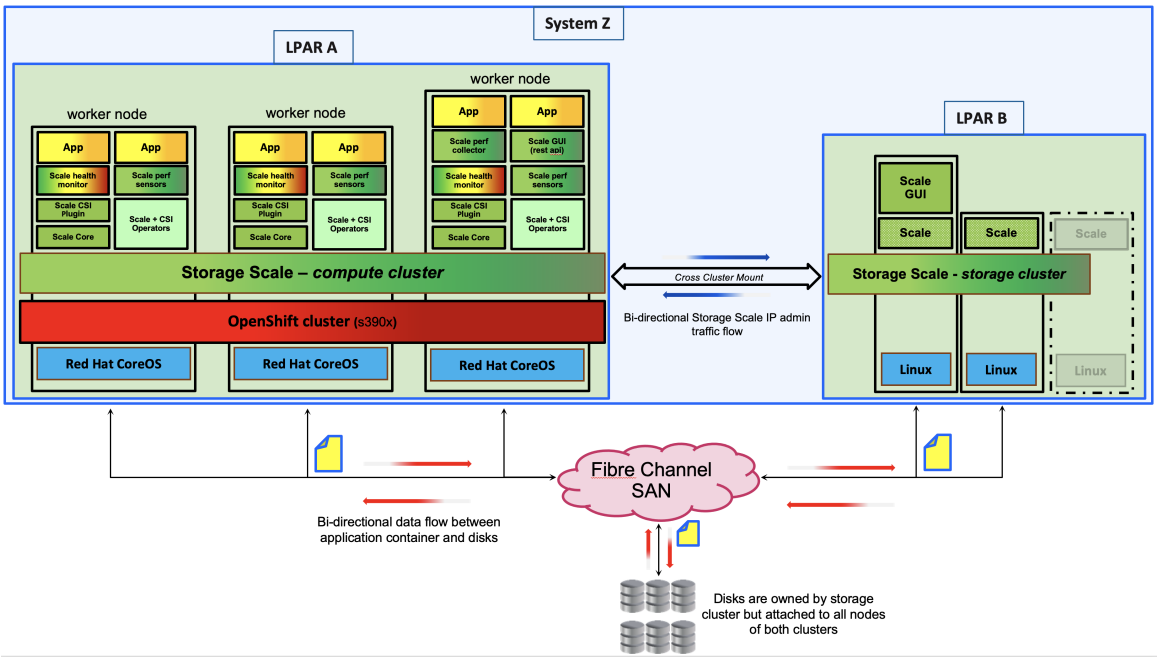
- Support for direct storage attachment on x86_64, ppc64le (Power), and s390x (IBM Z) servers. In direct storage attachment configuration, the worker nodes use the SAN fabric instead of the IBM Storage Scale NSD protocol for I/O traffic.
- If using x86_64 or ppc64le (Power) servers, it might be necessary to load multi-path drivers through Red Hat CoreOS before storage can be seen.

Figure 1. Direct attach config



- The virtualization layers of an IBM Z server allow the physical connection of the disks containing the IBM Storage Scale file system data to both the storage cluster and the IBM Storage Scale container native cluster.
- For more information about setting up a direct storage attachment, see Attaching direct storage on IBM Z in IBM Storage Scale documentation.

Figure 2. Direct attach SystemZ



Network and firewall requirements

Introduction

It is best practice to restrict network traffic by using firewalls. In particular with Red Hat OpenShift Container Platform, pods can talk to servers on the internet by default. This abets attacks where malicious code is sideloaded from the internet. Therefore, it is recommended to restrict any internet access from within the Red Hat OpenShift Container Platform cluster to required sites. For more information, see Configuring your firewall.

The following list covers solely the requirements of IBM Storage Scale container native and IBM Storage Scale CSI.

IBM Storage Scale uses several communication channels that need to be opened in the corresponding firewalls otherwise some or all functions might be broken. The channels can be categorized into three major groups:

1. Communication between pods inside a Red Hat OpenShift Container Platform cluster.
   - Communication inside Red Hat OpenShift Container Platform cluster is subject to firewall rules on worker node's kernels and/or network policies that are defined in Red Hat OpenShift Container Platform.
2. Communication between Red Hat OpenShift Container Platform cluster and storage cluster.
   - Communication with the storage cluster is in addition subject to the firewall that sits on the load-balancer in front of Red Hat OpenShift Container Platform.
3. Communication between Red Hat OpenShift Container Platform cluster and servers on the internet.
   - Communication with the internet is subject to the boundary firewall of the data center and the boundary firewall of the intranet.

Communication between pods

IBM Storage Scale container native core pods use two network interfaces, an "admin" and a "daemon" interface. The "admin" interface is used for monitoring and management, while the "daemon" interface is used for file system I/O. In a default configuration, the pod is created with **`hostNetwork: true`** and both networks use the Kubernetes node internal network on the host and is exposed to the pod.

When using the default host networking, the networking requirements that are outlined must be satisfied and opened across the Kubernetes node internal network.

It is recommended to use Container Network Interface (CNI) driver for the daemon network instead of host networking. This allows for better security and isolation. If using CNI, only **Core Pod Daemon Network Requirements** need to be satisfied and the Admin network will use the default Red Hat OpenShift Pod SDN Network. For more information, see Container network interface (CNI) configuration.

Core pod admin network requirements

The following network requirements must be met when **`hostNetwork: true`** on the pod specification. This is the default deployment method when CNI is not used. These are only needed to be open between nodes within the local IBM Storage Scale container native cluster.

Table 1. Admin network requirements

Port number	Protocol	Use	Initiated direction
12345	TCP	Administrative SSH	All nodes to all nodes within the local cluster

Core pod daemon network requirements

Those are identical to the requirements for communication with a storage cluster. For more information, see Daemon network requirements.

Communication with storage cluster

Daemon network requirements

The following list of ports and protocols are used to communicate between all nodes within the IBM Storage Scale container native cluster and between nodes in any configured remote storage clusters. Each node acts as a server that may initiate connections to any other node. Ensure the list of ports and protocols are open for both inbound and outbound packet flows.

Table 2. Daemon network requirements

Port number	Protocol	Use	Initiated direction
1191	TCP	GPFS	All nodes to all nodes
--	ICMP	GPFS	All nodes to all nodes
Configurable ephemeral port range	TCP	GPFS Policy Engine and compatibility with clusters less than 5.1.3	All nodes to all nodes

Ephemeral port ranges

Ephemeral ports are used by the GPFS Policy Engine, which is used by the CSI snapshot and compression features. When **`tscCmdAllowRemoteConnections=yes`** is configured in the Cluster CR, ephemeral port ranges are also used for communication and compatibility with remote clusters running IBM Storage Scale version 5.1.3 or less.

If ephemeral ports are configured on the remote storage cluster, ensure that they are also configured in the Cluster CR for the IBM Storage Scale container native deployment.

For more information on how to set ephemeral port ranges, see Ephemeral port range.

REST API access

Table 3. REST API network requirements

Port number	Protocol	Use	Initiated direction
443	TCP	Storage cluster REST API	Pod network to storage cluster GUI nodes

Pod network encapsulates the requirement that IBM Storage Scale container native operator and IBM Storage Scale CSI operator and driver require the ability to reach the storage cluster GUI nodes REST API. These respective pods use the default pod network and schedule to any node selected.

Name resolution between IBM Storage Scale container native and storage cluster

The IBM Storage Scale container native core pods require name resolution to all storage cluster nodes, specifically the node's daemon node name. If these names cannot be configured in the environment's domain name service (DNS), then host aliases can be added in the cluster CR **`.spec.daemon.hostAliases`**. This configuration adds these host aliases to the internal DNS managed by IBM Storage Scale container native and applies only to name resolution performed by the IBM Storage Scale container native core pods.

Communication with the internet

Container registries

The images of IBM Storage Scale container native containers are located in the IBM Cloud Registry.

To access the IBM Cloud Container Registry by using the domains **`cp.icr.io`** and **`icr.io`**, you must add the following hostnames to your firewall rules:

- dd0.icr.io
- dd2.icr.io
- dd4.icr.io
- dd6.icr.io

Users that are located in China must also allow the following hostnames:

- dd1-icr.ibm-zh.com
- dd3-icr.ibm-zh.com
- dd5-icr.ibm-zh.com

- dd7-icr.ibm-zh.com

Egress to these sites is required unless the cluster is configured in airgap mode. For more information, see Disconnected installs.

Callhome

If callhome function is enabled, ingress and egress need to be open to **esupport.ibm.com** (**129.42.0.0/18**).

For more information, see Firewall recommendations for call home.

Encryption

If encryption is configured on the file system, egress on the following ports is required for each service to function:

Table 4. Encryption network requirements

Port number	Protocol	Use	Service
9083	TCP	initial setup	WebSphere Application Server
9443	TCP	initial setup	Security Key Lifecycle Manager REST API
5696	TCP	key retrieval	Security Key Lifecycle Manager KMIP API

For more information, see Firewall recommendations for IBM Security Key Lifecycle Manager.

Container network interface (CNI) configuration

IBM Storage Scale container native core pods uses two network interfaces, an "admin" and a "daemon" interface. The "admin" interface is used for monitoring and management, while the "daemon" interface is used for filesystem I/O. In a default configuration, both networks utilize the Kubernetes node internal network exposed on the host, and the pods are exposed to all host networking.

Configuring CNI provides advantages over the default configuration. The "admin" and "daemon" network interfaces become private to the IBM Storage Scale container native core pod. The "admin" interface moves to use the Kubernetes pod network, which includes security and isolation provided by Kubernetes and NetworkPolicies. The "daemon" interface, if configured by CNI, can be setup to be isolated, private, and high-speed.

When configuring custom network interfaces, CNI is the only supported method for IBM Storage Scale container native network interface. It offers enhanced security and isolation benefits beyond what configuring network interfaces directly on the host provides.

How to configure CNI

1. To configure CNI on OpenShift, follow the steps documented by Red Hat for understanding multiple networks.

   - If the node has only a single physical network attachment, then the network adapter needs to be shared between networks. There are several CNI flavors that allow this: **Bridge**, **IPVLAN** and **MACVLAN**.
     - The **MACVLAN** plugin supports network policies and should be the default choice.
   - If the node has multiple physical network attachments and you want to dedicate one of the physical networks to IBM Storage Scale, select **host-adapter** CNI. It will map a physical network adapter into a pod, making it inaccessible to the host and other pods.
   - **SR-IOV** surpasses the capabilties of **host-adapter**. Advanced features like RDMA, GPUdirect and bonding of network ports are accessible via **SR-IOV** hardware network. For more information, see About Single Root I/O Virtualization (SR-IOV) hardware networks.

     Features: RDMA, GPUdirect, and bonding of network ports are not currently supported by IBM Storage Scale. Also **SR-IOV** allows to partition the hardware adapter and hand those partitions to different pods. Configuration is more complex compared to other CNIs and choice of supported network adapters is limited. As of today, IBM Storage Scale has not been tested with **SR-IOV**.
   - The IP address mapping is required to be static. This can be achieved by setting up static IPs or by configuring DHCP static mapping. For remote mount of a filesystem from a IBM Storage Scale storage cluster, this network must be routed to the storage cluster's daemon network.

2. Configure each of the OpenShift nodes that comprise the IBM Storage Scale container native cluster:

   a) Create runtime configuration node annotation that has the CNI definition. The specific node annotations for IBM Storage Scale is **scale.spectrum.ibm.com/daemon-network**. The format of the CNI annotation value is formed in the same format of a single network as defined in the format specified by **k8s.v1.cni.cncf.io/networks**.

   Example:

```
annotations:
scale.spectrum.ibm.com/daemon-network: |-
    {
        "name": "daemon-network",
```

```
        "mac": "22:22:0a:11:37:b2",
        "ips": [
            "10.17.99.63"
        ]
    }
```

b) **ips** field must be set as this is the static IP desired for this CNI network. c) **mac** field may be set if you use dhcp ipam (backed by a statically mapped dhcp). **ips** still must be set in addition to **mac**.

This might seem redundant, but IBM Storage Scale container native uses **ips** to set up its own name resolution. This process is asynchronous and independent of the pod actually being created. If **ips** was not set, then dhcp address would not be discovered until after pod creation.

IBM Storage Scale container native and SELinux

IBM Storage Scale container native offers a default for Container Storage Interface (CSI) volume attachment behavior for Security-Enhanced Linux (SELinux) labels. This default is introduced beginning with IBM Storage Scale container native 5.1.7.0 release.

What is SELinux?

SELinux, or Security-Enhanced Linux, defines access controls for the applications, processes, and files on a system. It uses security policies, which are a set of rules that tell SELinux what can or cannot be accessed, to enforce the access allowed by a policy.

For more information about SELinux, see What is SELinux? in Red Hat Documentation.

How is SELinux controlled in Kubernetes?

Container processes are started with the SELinux context in the **.spec.securityContext.seLinuxOptions** field, while CSI-based volumes have all their files labeled to match on pod attachment during container creation.

SELinux Multi-Category Security (MCS) is used in Red Hat OpenShift. By default, containers have their SELinux level set to values annotated on the namespace. The category defaults annotated on the namespace are assigned automatically by OpenShift to limit overlap.

Red Hat OpenShift is based on Kubernetes.

Problems with SELinux relabel on volume attach

When attaching CSI-based volumes to pods in Kubernetes the container-runtime SELinux relabels all files in the volume. This is problematic for shared filesystems that support SELinux, such as the IBM Storage Scale filesystem.

Relabeling on volume attachment moves the security control to the consumer of file volumes instead of the owner of the files. In classic shared filesystem environments, such as IBM Storage Scale, access is controlled by node administrators and file owners. In Kubernetes, volume access is isolated to a namespaced volume claim and controlled using Kubernetes role-based access control (RBAC). While in classic shared filesystem environments, such as IBM Storage Scale, access is controlled by node administrators and file owners. This Kubernetes behavior makes it very difficult to maintain access controls for volumes and files that are shared outside of Kubernetes. This is because Kubernetes ignores and overwrites any SELinux security isolation set by external administrators.

In addition, relabeling all files in a volume is a non-trivial operation, which may generate a large I/O load on the backend storage system. This introduces a performance and denial of service concern. Volumes with too many files, or shared volumes that are attached concurrently, may easily swamp the storage subsystem and cause cascading pod creation timeouts.

Upstream Kubernetes limitation for SELinux relabel

The SELinux relabel issue is not unique to IBM Storage Scale. Upstream Kubernetes does not give CSI volume driver implementations control of SELinux relabel on volume attachment.

Legacy in-tree volume drivers, such as the in-tree NFS volume driver, have the ability to control SELinux relabeling. However, legacy in-tree volume drivers are deprecated.

In Red Hat OpenShift, the container-runtime that performs the SELinux relabel will skip the relabeling if the container SELinux context is set to **spc_t**, which is the super privileged container. It is an uncontained type and may access any file on the system allowed by standard file permissions.

Default volume attachment behavior by IBM Storage Scale container native

To prevent Kubernetes from relabeling SELinux file labels, IBM Storage Scale container native by default will mount the filesystem with a container permissive context. This disallows the **security.selinux** label from being set, and all files inside the filesystem will be considered to have the context defined on the filesystem mount. By default that context would allow all containers running as **container_t** SELinux type to access files on the filesystem allowed by standard file permissions. By default, the mount context is set to **system_u:object_r:container_file_t:s0**.

Comparing SELinux relabel on attach and SELinux mount context

Security considerations

Setting a container permissive SELinux mount context or doing an SELinux relabel on volume attachment have similar access control within Kubernetes. Any container that can claim a volume may access its files.

However, if a process could escape a container or the host was able to be compromised, then any **container_t** constrained context would have access to the filesystem from a SELinux access perspective. Standard linux file permissions would still prevent access. This means containers that run within the "restricted" SecurityContextConstraint defined by OpenShift would not be able to access files in a volume that it did not explicitly share.

If a volume is shared with Kubernetes from an external application, then SELinux relabeling breaks SELinux security isolation managed externally. SELinux relabel on attach allows a volume consumer to ignore and overwrite SELinux labels, potentially exposing the files to other external systems. Setting the SELinux mount context will only allow the external SELinux labels to be ignored within the confines of the Kubernetes cluster.

Performance considerations

Since SELinux mount context disallows SELinux relabeling, volume attachments to pods do not generate extra I/O load and will attach faster than if SELinux relabeling is done.

If SELinux relabeling occurs, it must complete within a minute or the container creation will timeout and fail. Multiple relabel operations occurring concurrently are more likely to trigger timeout and failure. This cascading failure may only be exposed due to wider outages (node, cluster, lab), or maintenance.

External access considerations

External access refers to any components outside of the IBM Storage Scale container native cluster. This includes the storage cluster that is remotely mounted to the IBM Storage Scale container native cluster.

SELinux mount context means all new files are created without SELinux labels, and are considered **unlabeled_t**. This would include files created by application containers using IBM Storage Scale container native volumes. Generally, access to **unlabeled_t** files is only allowed by uncontained or more privileged process contexts. To access the unlabeled files, the external applications outside of Kubernetes must be given access to **unlabeled_t** via SELinux user policies, the files must be labeled manually, or the mount used by the application has a valid SELinux context configured.

The behavior without mount context, which does an SELinux relabel of all files, is similar in that the SELinux context of the container should be configured to match what is desired externally, or the external application should be granted access to files created by the container SELinux context.

How to change mount context

The mount context may be set on the Filesystem kind. This applies to all applications using volumes within the filesystem. For more details, run **oc explain fs.spec.seLinuxOptions** command.

When changing the SELinux context, applications with ReadWriteMany volumes running on nodes with mixed SELinux contexts may experience "Permission Denied" errors. Applications running on nodes with differing SELinux contexts may not have access to each other's files during the duration of the SELinux context update.

Changing the **fs.spec.seLinuxOptions** field of a Filesystem kind will cause pods to restart, and the restart of the pod will cause a reboot of the node itself. Ensure that the proper maintenance window and precautions, similar to upgrade, have been taken prior to changing this field.

```
spec:
  seLinuxOptions:
    user: <user>
    role: <object>
    type: <type>
    level: <level>
```

Legacy relabel on attach behavior

Earlier behavior of relabel on attach is considered legacy and the support is limited for it. If this behavior is required, contact IBM support.

Related links

- Red Hat solution - pods fail to start due to volumes with high volume counts in Red Hat Openshift
- Kubernetes Enhancements 1710 - Speed up SELinux volume relabeling using mounts
  - https://github.com/kubernetes/enhancements/issues/1710
- Red Hat OpenShift - Understanding host and VM security
- Red Hat OpenShift and SELinux

Roles and personas

Different roles, cluster roles, and levels of access are needed to deploy a fully functioning IBM Storage Scale container native cluster.

Personas

Red Hat OpenShift cluster administrator

A user with cluster administrator privileges needs to deploy the IBM Storage Scale container native cluster. Cluster admin privileges are needed in order to create higher privilege artifacts such as:

- Namespaces
- Custom Resource Definitions
- Cluster Roles and Cluster Role Bindings
- Security Context Constraints

By having the OpenShift cluster administrator execute the cluster deployment, the operator pod can be configured in a more restricted manner with minimal privilege.

IBM Storage Scale Storage cluster administrator

A user with privilege and access to configure the existing IBM Storage Scale storage cluster for remote access is also required for a successful deployment of an IBM Storage Scale container native cluster. Since the container native cluster utilizes remote mounts, the storage cluster admin must be able to execute commands against the storage cluster. For more information, see IBM Storage Scale storage cluster.

Lab administrator

A user with access to customer infrastructure and network is required to ensure a successful IBM Storage Scale container native cluster deployment. All OpenShift nodes that comprise the IBM Storage Scale container native cluster must be able to communicate with the remote IBM Storage Scale storage cluster. This may require a Lab Administrator to tune the customer network firewall to allow such communications. For more information, see Firewall recommendations.

Operator permissions

The IBM Storage Scale container native operator is a cluster-scoped operator. The operator watches all namespaces on the OpenShift cluster it is deployed into. Since the operator is cluster scoped, it requires access to cluster level resources to successfully deploy. Access to cluster level resources is handled through a cluster role that is deployed via RBAC YAML files. The cluster role is bound to the custom **`ibm-spectrum-scale-operator`** ServiceAccount, which the operator uses to create the IBM Storage Scale container native cluster.

To view the permissions of the operator, use the following query on a system that has a deployed container native operator.

```
oc describe clusterrole ibm-spectrum-scale-operator
```

This command will list out every resource the operator has access to and what it can do with them.

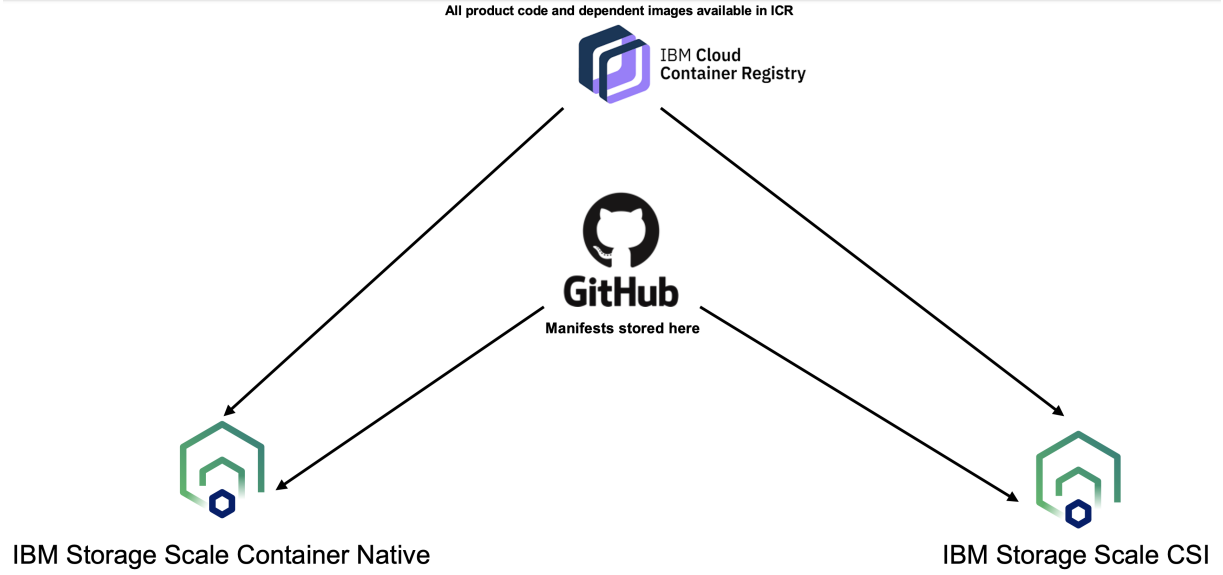
Roles

Once a IBM Storage Scale container native cluster is operational, users can authenticate to the IBM Storage Scale container native GUI via existing OCP roles. For more information, see OpenShift users.

Container images

The container images are required for the successful deployment of IBM Storage Scale container native. All images required for the deployment of IBM Storage Scale container native cluster are sourced from the IBM Container Repository.

Figure 1. Dependent images available in ICR



It is recommended to use the latest fixpack release available.

See Container image list for IBM Storage Scale container native on GitHub for the container image digests.

Air-gapped installs

In production, it is common to have a cluster that does not have internet access. If your Red Hat OpenShift Container Platform cluster is air-gapped (otherwise known as offline or disconnected), you can install IBM Storage Scale container native by mirroring images. These tasks must be ran by a Red Hat OpenShift administrator.

A bastion host is a device that has access to both the public internet and the network-restricted environment where a local registry and Red Hat OpenShift Container Platform clusters reside. Using the bastion host, you can mirror your images directly to the local registry.

You need to do the Air gap setup if the worker nodes are not able to access the repository due to network and firewall restrictions.

Prerequisites

- A production-grade Docker V2 registry that is available and accessible from the OpenShift Container Platform cluster nodes such as Quay Enterprise, JFrog Artifactory, or Docker Registry. The Red Hat OpenShift Internal Registry is not supported.
- A bastion node that:
  - Access to the public internet.
  - Access to the network-restricted environment where the local registry exists.
  - Access to the entitled registry: `icr.io:443`, `cp.icr.io:443`.
  - `skopeo` command is installed.
- Access to the Red Hat OpenShift Container Platform cluster as a user with the `cluster-admin` role.

Do not use Red Hat OpenShift internal image registry as your local registry. The Red Hat OpenShift registry does not support multi-architecture images or path separators in the image name.

Create an ImageContentSourcePolicy resource

Create a **`ImageContentSourcePolicy`** resource on your Red Hat OpenShift cluster to enable the redirection of requests to pull images from a repository on a mirrored image registry. Complete the following steps from the bastion host:

1. Create the sample **`registrymirror.yaml`**. Edit and replace **`example.io/<path_to_images>`** to match the path structure of your internal image registry.

```
cat << EOF > registrymirror.yaml
apiVersion: operator.openshift.io/v1alpha1
kind: ImageContentSourcePolicy
metadata:
  name: icr-mirror
spec:
  repositoryDigestMirrors:
  - mirrors:
    - example.io/<path_to_images>
```

```
    source: cp.icr.io/cp/spectrum/scale
  - mirrors:
    - example.io/<path_to_images>
    source: icr.io/cpopen
EOF
```

Do not prefix mirrors with **http://** or **https://** and make sure that there is no trailing **/** character.

2. Apply the sample to create an **ImageContentSourcePolicy** named **icr-mirror**:

```
oc apply -f registrymirror.yaml
```

3. Verify that the **ImageContentSourcePolicy** resource is created:

```
oc get imageContentSourcePolicy
```

4. This update is rolled out to all nodes. Verify your cluster node status and wait for all nodes to be updated before proceeding:

```
oc get MachineConfigPool
```

For more information, see About disconnected installation mirroring in Red Hat OpenShift documentation.

Mirror the images

Mirror the images by copying images from the source registry to the internal image registry.

Complete the following steps from the bastion node:

1. Log in to the IBM Entitled Container Registry with the **skopeo** command:

```
skopeo login cp.icr.io
```

2. Log in to the internal image registry with the **skopeo** command:

```
skopeo login example.io
```

3. Use **skopeo** to copy the images from the IBM Entitled Container Registry to your internal image registry.

   The image listing for each release is available on GitHub. See Container image list for IBM Storage Scale container native in the GitHub repository.

   Copy the air-gapped images from GitHub to a file called **images.txt**. You can use the following helper script to get started in creating the **skopeo copy** commands to run:

```
# Set the INTERNAL registry name
export INTERNAL=example.io
for image in `cat images.txt | grep -v \#`; do
    if [[ $image == *"cpopen"* ]]; then
        echo "skopeo copy --all docker://${image} docker://$INTERNAL/${image#*icr.io/}"
    else
        echo "skopeo copy --all docker://${image} docker://$INTERNAL/${image#*cp.icr.io/}"
    fi
done
```

   The helper script is provided as an example. Use at your own risk.

   A generic **skopeo copy** command is:

```
skopeo copy --all docker://<source image registry>/<image> docker://<internal image registry>/<image>
```

   For more information, see Skopeo Copy to the Rescue.

4. Log out of the IBM Entitled Container Registry by entering the **skopeo** command.

```
skopeo logout cp.icr.io
```

5. Log out of your internal image registry by entering the **skopeo** command.

```
skopeo logout example.io
```

Red Hat OpenShift Container Registry pull secret

For images to be properly pulled at the pod level, the Red Hat OpenShift global pull secrets must be modified to contain credentials to access your internal Container Registry.

Complete the following steps:

1. Create a base64 encoded string of the credentials used to access your internal Container Registry.

The following example uses **example.io/subdir** as the internal Container Registry.

- Use the credentials to access your **example.io/subdir** internal Container Registry.

  ```
  echo -n "<username>:<password>" | base64 -w0
  ```

2. Create an **authority.json** to include the base64 encoded string of your credentials. Use your username and password to access the internal Container Registry **example.io/subdir**.

   ```
   {
     "auth": "<base64 encoded string from previous step>",
     "username":"<example.io username>",
     "password":"<example.io generated entitlement key>"
   }
   ```

3. Enter the following command to include the **authority.json** as a new authority in your **.dockerconfigjson** and store it as **temp_config.json**:

   If the internal Container Registry is **example.io/subdir**, use **example.io** as the input key for the contents of **authority.json**.

   ```
   # Set the INTERNAL registry name
   export INTERNAL=example.io

   oc get secret/pull-secret -n openshift-config -ojson | \
   jq -r '.data[".dockerconfigjson"]' | \
   base64 -d - | \
   jq --arg key $INTERNAL '.[] | .[$key] += input' - authority.json > temp_config.json
   ```

   This command is supported by **jq** version 1.5 or higher.

   - Enter the following command to verify that your authority credentials were created in the resulting file:

     ```
     cat temp_config.json
     ```

     The content of **temp_config.json** displays all your existing credentials for global pull secrets with **example.io** appended at the end:

     ```
     # cat temp_config.json
     {
         ...
         ...
           "example.io": {
              "auth": "<base64 encoded string created in previous step>",
              "username": "<example.io username>",
              "password": "<example.io password>"
           }
       }
     }
     ```

4. Apply the pull secret configuration to the Red Hat OpenShift cluster.

   ```
   oc set data secret/pull-secret -n openshift-config --from-file=.dockerconfigjson=temp_config.json
   ```

5. Verify that your pull-secret is updated with your new authority, enter the following command and confirm that your authority is present:

   ```
   oc get secret/pull-secret -n openshift-config -ojson | \
   jq -r '.data[".dockerconfigjson"]' | \
   base64 -d -
   ```

6. This update is rolled out to all nodes. Verify your cluster node status and wait for all nodes to be updated before proceeding:

   ```
   oc get MachineConfigPool
   ```

7. Remove the temporary files that were created.

   ```
   rm authority.json temp_config.json
   ```

Testing the pull of images from the mirrored registry

Complete the following steps from the bastion node:

1. Pick a worker node from **oc get nodes** and start a debug pod:

   ```
   oc debug node/<worker node>
   ```

2. When you see a prompt, use host binaries by entering the **chroot /host** command.

   ```
   # oc debug node/worker0.example.com
   Starting pod/worker0examplecom-debug ...
   To use host binaries, run `chroot /host`
   ```

```
Pod IP: 12.34.56.78
If you don't see a command prompt, try pressing enter.
# chroot /host
```

3. Use the **podman login** command to authenticate your mirrored image registry.

```
podman login example.io
```

4. Attempt to pull one of the images from the source image registry by using **podman**. The Red Hat OpenShift cluster must be able to redirect the request from the external image registry to the internal image registry and successfully pull the image.

```
podman pull cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-
gui@sha256:78216ac2a8157753bd70f273f5e696479538007f43312bacd8e7f8e30f146276
```

   A valid image from the **image.txt** can be used here to verify that things are working.

5. Verify that the image is pulled.

```
# podman images | grep cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-gui
cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-gui    <none>     9c215ae62f37   22 hours ago    851 MB
```

Storage cluster

IBM Storage Scale container native supports remote mounting file systems in the following scenarios.

- On-premise
- Cloud

A storage cluster is not needed if only local file systems are used.

On-premise

Some additional tasks need to be performed on the IBM Storage Scale storage cluster that runs on-premise.

IBM Storage Scale storage cluster

The operators of IBM Storage Scale container native and IBM Storage Scale CSI interact with the storage cluster through ReST API (which is part of the GUI stack). To enable the operator to interact with storage cluster, user IDs need to be created on the storage cluster GUI. Specific roles are used to grant those user IDs only the operations that are needed to provide their functionality. In addition, some settings on the cluster and the filesystem are required for interoperability with CSI.

If the storage cluster is running a GUI high availability configuration, for example, having 2 or more GUI nodes installed, ensure that the storage cluster is running IBM Storage Scale 5.1.6.1 or higher before use with IBM Storage Scale container native and IBM Storage Scale CSI.

Creating container operator and CSI users

Complete the following steps in the shell of the GUI node of the storage cluster:

1. To create container native operator GUI user, enter the following command:

```
/usr/lpp/mmfs/gui/cli/mkuser cnsa_storage_gui_user -p cnsa_storage_gui_password -g ContainerOperator
```

   By default, user passwords expire after 90 days. If the security policy of your organization permits it, use the **-e 1** option on the **mkuser** command to create a user with a password that does not expire.

2. To create the CSI GUI user, enter the following commands:

```
/usr/lpp/mmfs/gui/cli/mkuser csi_storage_gui_user -p  csi_storage_gui_password -g CsiAdmin
```

   By default, user passwords expire after 90 days. If the security policy of your organization permits it, use the **-e 1** option on the **mkuser** command to create a user with a password that does not expire.

To update the passwords, see Updating authentication to the storage cluster.

Storage cluster configuration for Container Storage Interface (CSI)

Complete the following steps on the *storage cluster* to ensure that the IBM Storage Scale CSI driver can operate successfully:

1. Ensure that the perfileset quota on the file systems that are used by IBM Storage Scale Container Storage Interface driver is set to **No**.

The IBM Storage Scale Container Storage Interface driver will create many filesets (one per PV). Tracking user and group quotas on a per-fileset basis will significantly increase the overhead of quota management. As a result, the file system performance can suffer.

```
mmlsfs fs1 --perfileset-quota
```

2. Enter the following command to enable the Quota in the file systems:

   The IBM Storage Scale Container Storage Interface driver translates capacity of persistent volumes to fileset quotas. For this to work, quotas are required to be enabled in the file system.

   ```
   mmchfs fs1 -Q yes
   ```

   Enter the following command to verify quota is enabled:

   ```
   mmlsfs fs1 -Q
   ```

3. Enable the quota for root user by entering the following command:

   On Kubernetes, the containers may run as root, so ensure that quotas are enforced for the root user as well.

   ```
   mmchconfig enforceFilesetQuotaOnRoot=yes -i
   ```

4. Ensure that the **controlSetxattrImmutableSELinux** parameter is set to "yes" by entering the following command:

   Kubernetes does not honor immutability of files/directories when setting SELinux labels. This creates issues, for example, with the immutable **.snapshot** directory.

   ```
   mmchconfig controlSetxattrImmutableSELinux=yes -i
   ```

5. Enable **filesetdf** of the file system by entering the following command:

   IBM Storage Scale Container Storage Interface driver will only be able to report free space on persistant volumes if **filesetdf** is set correctly.

   ```
   mmchfs fs1 --filesetdf
   ```

6. IBM Storage Scale Container Storage Interface driver has no information about the number of inodes a persistent volume will consume. Therefore, corresponding independent filesets are created with default values for maxIndoes. To enable automatic expansion of the inode space so persistent volumes do not run out of inodes, enter the following command:

   ```
   mmchfs fs1 --auto-inode-limit
   ```

   The **--auto-inode-limit** option is available only with filesystem format level 28.00 or later. Enable **auto-inode-limit** as soon as the filesystem format level is updated to 28.00 or later. On older filesystem levels the administrator of the storage cluster needs to manually increase the inode limit when warnings for low inodes are raised by the health monitoring.

   For more information about **auto-inode-limit** parameter, see mmchfs command in IBM Storage Scale documentation.

Configure cluster profile with tscCmdAllowRemoteConnections

Starting with version IBM Storage Scale and IBM Storage Scale container native 5.1.3, the **tscCmdAllowRemoteConnections** configuration is recommended to be set to **no**. If the storage cluster and all client clusters (including IBM Storage Scale container native) are at versions >= 5.1.3, it is recommended to set this value to **no**. However, if any version is < 5.1.3, **tscCmdAllowRemoteConnections** must be set to **yes** on the storage cluster and client clusters to successfully communicate between the clusters.

Use the following table as a reference.

Table 1. tscCmdAllowRemoteConnections configuration

Storage cluster version	IBM Storage Scale container native version	tscCmdAllowRemoteConnections
< 5.1.3	< 5.1.3.0	yes
>= 5.1.3	< 5.1.3.0	yes
>= 5.1.3	>= 5.1.3.0	no

- To change this value on the storage cluster, run: **mmchconfig tscCmdAllowRemoteConnections='yes|no'**.
- To change this value on the IBM Storage Scale container native cluster, set **tscCmdAllowRemoteConnections: yes|no** in the clusterProfile section of the cluster spec:

```
kind: Cluster
metadata:
name: ibm-spectrum-scale
spec:
   ...
   daemon:
   ...
```

```
    ...
    clusterProfile:
       tscCmdAllowRemoteConnections: "yes"
```

For more information to configure the **clusterProfile** section of the cluster spec, see Cluster profile.

For more information about all IBM Storage Scale services, see Securing the IBM Storage Scale system using firewall in IBM Storage Scale documentation.

Cloud

AWS storage cluster

IBM Storage Scale container native with Red Hat OpenShift Service on AWS (ROSA) provides a highly scalable and performant solution for containerized applications. You can use the IBM Storage Scale Storage cluster on public clouds (such as AWS) to store its persistent data. IBM Storage Scale storage clusters on public clouds can be deployed and managed with the **cloudkit** CLI. For more information, see Cloudkit in IBM Storage Scale documentation.

cloudkit is included in the IBM Storage Scale Self-Extracting (SE) package and enables users to deploy IBM Storage Scale clusters across multiple cloud providers (currently limited to AWS), providing deployment topology flexibility. **cloudkit** helps:

- Reduce the complexity and time that is needed to set up the storage cluster.
- Offers the ability to configure various cluster parameters, such as disk type, instance type, and network configurations.
- Offers the ability to integrate with other IBM Storage Scale features, such as data replication, data tiering, and data encryption.

Considerations

Consider the list:

- Understand the requirements in terms of performance, scalability, data availability, and data protection.
- Consider the structure of your storage, including the number of storage nodes, the file system size, and the number of replicas you want to maintain.
- ROSA and **cloudkit** both use Multi-Availability Zone (AZ) to ensure resiliency, but in Multi-AZ mode, the inter-AZ network limit constrains the throughput.
- Review the AWS quota limits to ensure that your existing Virtual Private Cloud (VPC) has enough free IP addresses to accommodate both ROSA and **cloudkit**. If you are planning to create a new VPC, you should plan the network to accommodate the IP requirements for both services.
- Users have the flexibility to start with either ROSA in new VPC mode and **cloudkit** in existing VPC mode (or) **cloudkit** in new VPC mode and ROSA in existing VPC mode, depending on the requirements.
- For optimal performance, choose a storage cluster deployment within the same subnet that is in use by the ROSA, providing low latency and higher throughput.

The following tasks assume that the installation of IBM Storage Scale container native on AWS ROSA is complete.

Deploy storage cluster

Use the **cloudkit** command to deploy the storage cluster.

```
cloudkit create cluster
```

cloudkit provides various deployment profiles, including **Throughput-Performance-Scratch-Storage**, **Throughput-Performance-Persistent-Storage**, **Throughput-Advance-Persistent-Storage**, and **Balanced**.

If you are deploying to a Single AZ, it is advisable to use the **Throughput-Performance-Persistent-Storage** profile. For multi-AZ, use the **Balanced** profile.

Do not use the **Throughput-Performance-Scratch-Storage** profile for IBM Storage Scale container native workloads. This profile uses instance storage and results in data loss when storage nodes are shut down.

Details of the cluster can be viewed by using the describe option:

```
cloudkit describe cluster
```

Perform grant filesystem operation

cloudkit performs setting up security rules for data access and exchange between the ROSA worker security group and storage cluster. It performs the prerequisite configurations such as creating users in the **ContainerOperator** and **CsiAdmin** groups on the storage cluster.

Use the **cloudkit** command to simplify the data management across multiple IBM Storage Scale container native clusters.

```
cloudkit grant filesystem
```

Details of the file system access granted by **cloudkit** can be viewed by using the describe option:

```
cloudkit describe grant
```

Perform revoke filesystem operation

If the IBM Storage Scale container native is uninstalled, revoke the access to the file system by using the following command:

```
cloudkit revoke filesystem
```

Delete storage cluster

The following action cannot be undone and will destroy all scale instances and wipe the disks that are used for the file system. Take the appropriate data backups before proceeding, failure to do so will result in permanent loss of data.

Use the following **cloudkit** command to delete the storage cluster:

```
cloudkit delete cluster
```

Red Hat OpenShift Container Platform

Refer to the following sections to help prepare your Red Hat OpenShift environments.

- IBM Cloud Container Registry (ICR) pull secrets
- On-premise
  - Worker node configuration
  - Master node configuration (compact cluster)
  - Infrastructure node configuration
- Red Hat OpenShift service on AWS
  - Support matrix for IBM Storage Scale container native and ROSA
  - Red Hat OpenShift configuration on AWS
- Validating Red Hat OpenShift configuration

IBM Cloud Container Registry (ICR) pull secrets

IBM Storage Scale container native images are hosted in the IBM Cloud Container Registry. Obtain an entitlement key from IBM container software library.

Entitlement keys determine whether the IBM Storage Scale operator can automatically pull the IBM Storage Scale container native images. Image pull failures may occur due to an invalid entitlement key or a key belonging to an account that does not have valid entitlement.

Adding IBM Cloud Container Registry credentials

For images to be properly pulled at the pod level, the Red Hat OpenShift global pull secrets must be modified to contain credentials to access the IBM Cloud Container Registry.

The following steps apply to clusters that can directly access IBM Cloud Container Registry. For air gap installs, see Disconnected installs

1. Create a base64 encoded string of the credentials used to access the image registry.

   - For using IBM Cloud Container Registry, the credentials must use the **cp** user along with the entitlement key.

     ```
     echo -n "cp:REPLACE_WITH_GENERATED_ENTITLEMENT_KEY" | base64 -w0
     ```

2. Create an **authority.json** to include the base64 encoded string of your credentials, the fixed username **cp** (used to access the **cp.icr.io** repository), and the entitlement key for the IBM Cloud Container Registry.

   ```
   {
     "auth": "REPLACE_WITH_BASE64_ENCODED_KEY_FROM_PREVIOUS_STEP",
     "username":"cp",
     "password":"REPLACE_WITH_GENERATED_ENTITLEMENT_KEY"
   }
   ```

3. Enter the following command to include the **authority.json** as a new authority in your **.dockerconfigjson** and store it as **temp_config.json**:

   Using the IBM Cloud Container Registry as the authority, use **cp.icr.io** as the input key for the contents of **authority.json**.

```
oc get secret/pull-secret -n openshift-config -ojson | \
jq -r '.data[".dockerconfigjson"]' | \
base64 --decode | \
jq '.["cp.icr.io" += input' - authority.json > temp_config.json
```

This command is supported by `jq` 1.5.

- Verify that your authority credentials are appended at the end of the file:

```
# cat temp_config.json
{
   "auths": {
      ...
      ...
      ...
      "cp.icr.io": {
         "auth": "REPLACE_WITH_BASE64_ENCODED_KEY_FROM_PREVIOUS_STEP",
         "username": "cp",
         "password": "REPLACE_WITH_GENERATED_ENTITLEMENT_KEY"
      }
   }
}
```

4. Use the contents of the `temp_config.json` file, and apply the updated config to the Red Hat OpenShift cluster.

   ```
   oc set data secret/pull-secret -n openshift-config --from-file=.dockerconfigjson=temp_config.json
   ```

   To verify that your pull-secret is updated with your new authority, issue the following command and confirm that your authority is present:

   ```
   oc get secret/pull-secret -n openshift-config -ojson | \
   jq -r '.data[".dockerconfigjson"]' | \
   base64 -d -
   ```

5. This update is rolled out to all nodes. Use `oc get mcp` to track the progress of the roll out.

6. Enter the following command to remove the temporary files that were created:

   ```
   rm authority.json temp_config.json
   ```

On-premise

Extra configurations need to be made to the Red Hat OpenShift cluster installation for IBM Storage Scale container native cluster to operate correctly. For more information on, see Installing in Red Hat OpenShift documentation.

These configuration tasks can also be handled during the Red Hat OpenShift Container Platform installation by adding day-1 kernel arguments. For more information, see Installation configuration in Red Hat OpenShift documentation.

Validate the following tasks are complete ahead of installing the IBM Storage Scale container native code on an on-premises Red Hat OpenShift.

- Worker node configuration
- Master node configuration (compact cluster)
- Infrastructure node configuration

Worker node configuration

Applying Machine Config Operator (MCO) settings

The following `MachineConfiguration` is provided as a convenience to easily change the following configuration:

- Kernel development and kernel header packages

  Install the kernel-related packages for IBM Storage Scale to successfully build its portability layer.

- Increase vmalloc kernel parameter

  Modify the kernel parameters that are needed to operate properly with Red Hat CoreOS. This applies only to the IBM Storage Scale running on Linux™ on Z.

Apply the correct sample file based on your OpenShift Container Platform version and machine architecture.

- If you are running x86_64, enter the following command:

For 4.13:

```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_x86_64.yaml
```

For 4.14:

```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_x86_64.yaml
```

For 4.15:

```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_x86_64.yaml
```

- If you are running ppc64le, enter the following command:

  For 4.13:

  ```
    oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_ppc64le.yaml
  ```

  For 4.14:

  ```
    oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_ppc64le.yaml
  ```

  For 4.15:

  ```
    oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_ppc64le.yaml
  ```

- If you are running s390x, enter the following command:

  For 4.13:

  ```
    oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_s390x.yaml
  ```

  For 4.14:

  ```
    oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_s390x.yaml
  ```

  For 4.15:

  ```
    oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_s390x.yaml
  ```

Infrastructure node configuration

Similar to the configuration tasks on the workers nodes, Machine Configuration Operator (MCO) settings must be applied to the infrastructure nodes if you want to run IBM Storage Scale core pods.

If your infrastructure nodes share roles with the worker pool, indicated by `ROLES: infra,worker`, you can ignore the following instructions.

Apply the correct sample file based on your OpenShift Container Platform version and machine architecture.

- If you are running x86_64, enter the following command:

  For 4.13:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_x86_64.yaml | sed 's/worker/infra/g' | oc apply -f -
  ```

  For 4.14:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_x86_64.yaml | sed 's/worker/infra/g' | oc apply -f -
  ```

  For 4.15:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_x86_64.yaml | sed 's/worker/infra/g' | oc apply -f -
  ```

- If you are running ppc64le, enter the following command:

For 4.13:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_ppc64le.yaml | sed 's/worker/infra/g' | oc apply -f -
```

For 4.14:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_ppc64le.yaml | sed 's/worker/infra/g' | oc apply -f -
```

For 4.15:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_ppc64le.yaml | sed 's/worker/infra/g' | oc apply -f -
```

- If you are running s390x, enter the following command:

  For 4.13:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_s390x.yaml | sed 's/worker/infra/g' | oc apply -f -
```

  For 4.14:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_s390x.yaml | sed 's/worker/infra/g' | oc apply -f -
```

  For 4.15:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_s390x.yaml | sed 's/worker/infra/g' | oc apply -f -
```

Master node configuration (compact cluster)

You can deploy compact-3-node clusters with Red Hat OpenShift Container Platform 4.5 and later. For more information, see Delivering a Three-node Architecture for Edge Deployments in Red Hat Hybrid Cloud documentation.

In this configuration, verify that the system is sized correctly to operate smoothly. In compact clusters, the Kubernetes control plane, IBM Storage Scale container native pods, and user applications, all compete for the same resources in the cluster. High application loads can impact the control plane resources, causing the Red Hat OpenShift cluster to become unusable or unstable. etcd is sensitive to latency and might present as frequent leader elections, and other instabilities. IBM Storage Scale container native might also take down the file system if resources are constrained.

For more information, see Recommended etcd practices in Red Hat OpenShift documentation.

Schedulable control plane nodes

To allow pod placement for master nodes, also known as control plane nodes, make sure that they are configured as **schedulable**. By default, control plane nodes are not schedulable.

Verify **mastersSchedulable** is set to **true** by entering the following command:

```
oc get schedulers.config.openshift.io cluster -ojson | jq -r ".spec.mastersSchedulable"
```

If this value is not **true**, patch the cluster by entering the following command:

```
oc patch schedulers.config.openshift.io cluster --type='json' \
-p='[{"op": "replace", "path": "/spec/mastersSchedulable", "value":true}]'
```

For more information, see Configuring control plane nodes as schedulable in the Red Hat OpenShift documentation.

Applying Machine Config Operator (MCO) settings

Similar to the configuration tasks required on the workers nodes, these Machine Config Operator (MCO) settings must be applied to the master nodes in a compact-cluster environment.

Apply the correct sample file based on your OpenShift Container Platform version and machine architecture.

- If you are running x86_64, enter the following commands for the relevant versions:

  For 4.13:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_x86_64.yaml | sed 's/worker/master/g' | oc apply -f -
```

For 4.14:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_x86_64.yaml | sed 's/worker/master/g' | oc apply -f -
```

For 4.15:

```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_x86_64.yaml | sed 's/worker/master/g' | oc apply -f -
```

- If you are running ppc64le, enter the following commands for the relevant versions:

  For 4.13:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_ppc64le.yaml | sed 's/worker/master/g' | oc apply -f -
  ```

  For 4.14:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_ppc64le.yaml | sed 's/worker/master/g' | oc apply -f -
  ```

  For 4.15:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_ppc64le.yaml | sed 's/worker/master/g' | oc apply -f -
  ```

- If you are running s390x, enter the following commands for the relevant versions:

  For 4.13:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.13/mco_s390x.yaml | sed 's/worker/master/g' | oc apply -f -
  ```

  For 4.14:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.14/mco_s390x.yaml | sed 's/worker/master/g' | oc apply -f -
  ```

  For 4.15:

  ```
    curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
  native/v5.2.0.x/generated/scale/mco/ocp4.15/mco_s390x.yaml | sed 's/worker/master/g' | oc apply -f -
  ```

Red Hat OpenShift Service on AWS

Red Hat OpenShift Service on AWS (ROSA) is a fully managed service that provides an enterprise-ready Red Hat OpenShift cluster to run applications on AWS. It offers an easy and quick deployment of an operational Red Hat OpenShift cluster without the concerns to administer the underlying infrastructure. The service also provides a range of features to help improve the security and reliability of Red Hat OpenShift clusters, including automated backups and recovery, fine-grained access control, and integration with AWS security services.

For more information on ROSA, see Red Hat OpenShift Service on AWS Documentation.

Refer to the following sections to know about the configuration requirements for Red Hat OpenShift on AWS.

- Support matrix for IBM Storage Scale container native and ROSA
- Red Hat OpenShift configuration on AWS

Support matrix for IBM Storage Scale container native and ROSA

The following table lists the support matrix for IBM Storage Scale container native and Red Hat Open Shift Service on AWS (ROSA).

Table 1. AWS support matrix

ROSA CLI version	OCP version	Autoscaling workers	Spot instances
1.2.22	See supported OCP versions in Software requirements	No	No

In case of issues related to IBM Storage Scale container native, contact IBM Support. For problems related to Red Hat OpenShift and AWS infrastructure (ROSA), contact AWS support.

Red Hat OpenShift configuration on AWS

Modify the Red Hat OpenShift Service on AWS (ROSA) platform for IBM Storage Scale container native to operate correctly.

Add Scale labels to the machine pool

- Modify the default machine pool

  ROSA provides a Command-Line Interface (CLI) which can be obtained from https://github.com/openshift/rosa/releases for editing node labels. This is needed to customize the machine pool that is used for running IBM Storage Scale container native. When you install ROSA, it creates a machine pool named "Default" that includes nodes with roles **worker** and **worker,infra**.

  To use these nodes for running IBM Storage Scale container native, you can modify the "Default" machine pool as follows:

  ```
  rosa edit machinepool -c <rosa-cluster-name> Default --labels node-
  role.kubernetes.io/scale=,scale.spectrum.ibm.com/daemon-selector= --replicas 3
  ```

  It is recommended to configure a minimum of 3 replicas.

- If you prefer to isolate your workload to a custom machine pool, you can create one and assign labels to the nodes by using the following command:

  ```
  rosa create machinepool -c <rosa-cluster-name> --name=<custom-pool-name> --labels node-
  role.kubernetes.io/scale=,scale.spectrum.ibm.com/daemon-selector=  --replicas 3
  ```

```
It is recommended to configure a minimum of 3 replicas.
```

Create a service account for applying Machine Config Operator (MCO)

As a regular user, you are prevented from modifying Red Hat managed resources. Therefore, it is advised to create a service account for Machine Config Operator (MCO) by following the provided steps:

1. Create a service account for Machine Config Operator (MCO):

   ```
   oc create serviceaccount scale-mco-sa
   ```

2. Add a role to service account. To add a "cluster-admin" role to a service account "scale-mco-sa", execute the following command:

   ```
   oc adm policy add-cluster-role-to-user cluster-admin -z scale-mco-sa
   ```

3. To create a new token for the service account (**scale-mco-sa**) and store it in the **MCO_SA_TOKEN** environment variable, execute the following command:

   ```
   export MCO_SA_TOKEN=$(oc create token scale-mco-sa)
   ```

4. Log in to OpenShift using the new token.

   ```
   oc login --token=$MCO_SA_TOKEN
   ```

For information on how to use service account in applications, see Using service accounts in applications in the Red Hat OpenShift documentation.

Create a new Machine Config Pool

To separate out nodes with labels "scale" and "worker" into a new machine pool, execute the following command:

```
echo '
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfigPool
metadata:
  name: scale-mcp
spec:
  machineConfigSelector:
    matchExpressions:
      - {key: machineconfiguration.openshift.io/role, operator: In, values: [worker,scale]}
  nodeSelector:
    matchLabels:
      node-role.kubernetes.io/scale: ""
' | oc apply -f -
```

Applying Machine Config Operator (MCO) settings

Apply the following MCO setting to install the kernel related packages for IBM Storage Scale to successfully build its portability layer.

```
echo '
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: "scale"
  name: 00-worker-ibm-spectrum-scale-kernel-devel
spec:
  selector:
    matchLabels:
      machineconfiguration.openshift.io/role: "scale"
  config:
    ignition:
      version: 3.2.0
  extensions:
     - kernel-devel
' | oc apply -f -
```

To validate the kernel-devel packages have been installed onto the machines, see Validate kernel packages.

Cleanup the service account

Once the Machine Config Operator task has been successfully completed, complete the following steps to clean up the service account:

1. Log out of the OpenShift cluster:

   ```
   oc logout
   ```

1. Log in to the OpenShift cluster as a regular user:

   ```
   oc login <OpenShift_URL> -u <regular_user> -p <regular_user_password>
   ```

2. Delete the service account that was created for performing MCO task:

   ```
   oc delete sa scale-mco-sa
   ```

Authorize ROSA worker security group to allow IBM Storage Scale container native ports

Complete the following steps to authorize ROSA worker security group:

1. In your AWS Management Console, navigate to the "EC2 Dashboard" in the region where ROSA is installed and select the "Security Groups" option from the navigation pane. Locate the Security Group for ROSA worker security group id and export the following variable.

   ```
   export AWS_ROSA_WORKER_SECURITY_GROUP=<security_group_id>
   ```

2. Use the following commands to Authorize Ingress Traffic to allow ports needed by IBM Storage Scale container native.

   ```
   aws ec2 authorize-security-group-ingress --group-id ${AWS_ROSA_WORKER_SECURITY_GROUP} --protocol tcp --port 12345  --source-group ${AWS_ROSA_WORKER_SECURITY_GROUP}
   aws ec2 authorize-security-group-ingress --group-id ${AWS_ROSA_WORKER_SECURITY_GROUP} --protocol tcp --port 1191  --source-group ${AWS_ROSA_WORKER_SECURITY_GROUP}
   aws ec2 authorize-security-group-ingress --group-id ${AWS_ROSA_WORKER_SECURITY_GROUP} --protocol tcp --port 60000-61000  --source-group ${AWS_ROSA_WORKER_SECURITY_GROUP}
   ```

Validating Red Hat OpenShift configuration

Red Hat OpenShift machine configuration changes are handled by the Machine Config Operator (MCO). To see the status of any machine configuration ongoing, run the following command:

```
oc get MachineConfingPool
```

Validate kernel packages

Validate that the **kernel-devel** package is installed on the nodes that deploy core pods.

In the following example, the label **scale.spectrum.ibm.com/daemon-selector=** is used to select the nodes that run core pods. Replace this selector to match what is configured in your cluster.

```
oc get nodes -lscale.spectrum.ibm.com/daemon-selector= \
-ojsonpath="{range .items[*]}{.metadata.name}{'\n'}" |\
xargs -I{} oc debug node/{} -T -- chroot /host sh -c "rpm -q kernel-devel"
```

This command creates a debug pod for all nodes that match the label selector. Use it with discretion if you have a large system.

s390x specific validation

Perform the extra validation steps if deployed on the s390x architecture.

Validate that the **vmalloc** kernel parameter is applied on the Red Hat OpenShift Container Platform worker nodes by entering the following command:

```
oc get nodes -lscale.spectrum.ibm.com/daemon-selector= \
-ojsonpath="{range .items[*]}{.metadata.name}{'\n'}" |\
xargs -I{} oc debug node/{} -T -- cat /proc/cmdline
```

In the following example, the value **vmalloc=4096G** is seen in the output at the end:

```
# oc debug node/worker1.example.com -- cat /proc/cmdline
Starting pod/worker1examplecom-debug ...
To use host binaries, run `chroot /host`
rhcos.root=crypt_rootfs random.trust_cpu=on ignition.platform.id=metal rd.luks.options=discard
$ignition_firstboot
ostree=/ostree/boot.1/rhcos/51e4c768b7c3dcec3bb63b01b9de9e8741486bf00dd4ae4df2d1ff1f872efe2e/0 vmalloc=4096G
```

This command creates a debug pod for all nodes that match the label selector. Use it with discretion if you have a large system.

Installing the IBM Storage Scale container native operator and cluster

The installation of the IBM Storage Scale container native operator and cluster includes several procedures.

- Labels and annotations
- Install
- Image pull secrets
- Kubernetes resources
  - Cluster
  - Callhome
  - RemoteCluster
    - Creating secrets for storage cluster GUI users
    - Configuring Certificate Authority (CA) certificates
  - Filesystems
  - Encryption

Labels and annotations

IBM Storage Scale container native assigns labels to worker nodes.

Designation labels

IBM Storage Scale container native automatically assigns designations to some worker nodes. You do not need to explicitly designate the worker nodes but if it is required then it can be done by using node labels.

The following mechanisms are supported to designate IBM Storage Scale container native nodes:

- Automatic *(Recommended)* - Allows the Operator to designate the nodes automatically.
- Manual *(Optional)* - Allows administrators to have more control of the placement of IBM Storage Scale node designations (like the quorum designation) to pods on specific worker nodes.

  Manual labeling requires insight about IBM Storage Scale and must not be used by unexperienced administrators.

Automatic

If the user does not label any nodes as quorum nodes, the Operator automatically applies quorum annotations to a subset of the nodes in the cluster. The number of nodes to be annotated depends on the number of nodes in the cluster:

- If the number of nodes in the cluster definition is less than 4, all nodes are designated as quorum nodes.

- If the number of nodes in the cluster definition is between 4 and 9 inclusive, 3 nodes are designated as quorum nodes.
- If the number of nodes in the cluster definition is between 10 and 18 inclusive, 5 nodes are designated as quorum nodes.
- If the number of nodes in the cluster definition is greater than 18, 7 nodes are designated as quorum nodes.

Kubernetes zones are considered whether they are configured in the Red Hat OpenShift cluster. The operator selects quorum nodes across all zones. For example, if 3 zones and 3 quorum nodes are to be designated as quorum nodes, then one node of each zone is designated as quorum node. For more information, see Kubernetes zones.

The automatic node designation works at initial cluster creation time only. When the cluster is created, the operator does not change node designations automatically (for example if nodes are added to the cluster). If node designations must be changed on an existing cluster, the Manual steps can be done, even if Automatic mode was used at cluster creation time.

Manual

Supported designation label values are **quorum** and **manager**. The nodes designated as **quorum** nodes also automatically assume the role of **manager**. If sufficient **quorum** nodes are designated, unlabeled nodes become client nodes within the cluster.

IBM Storage Scale quorum designation

For more information about IBM Storage Scale quorum designation, see Quorum in IBM Storage Scale documentation. It is mandatory to configure an odd number of nodes, with 3, 5, or 7 nodes are the typical numbers used.

IBM Storage Scale manager designation

For more information about IBM Storage Scale manager designation, see Special management functions in IBM Storage Scale documentation.

Node labeling

To see the list of nodes in your cluster, enter the **oc get nodes** command:

```
# oc get nodes
NAME                          STATUS   ROLES    AGE   VERSION
master0.example.com   Ready    master   50d   v1.16.2
worker0.example.com   Ready    worker   50d   v1.16.2
worker1.example.com   Ready    worker   50d   v1.16.2
worker2.example.com   Ready    worker   50d   v1.16.2
```

The following labels can be applied to nodes in the Red Hat OpenShift cluster to dictate how the pods are deployed on the nodes that are designated:

```
scale.spectrum.ibm.com/designation=quorum
scale.spectrum.ibm.com/designation=manager
```

To apply a label to a node, enter the **oc label node <node name> scale.spectrum.ibm.com/designation=<designation>** command as follows:

```
oc label node worker0.example.com scale.spectrum.ibm.com/designation=quorum
```

To verify that the label was applied to the node, enter the **oc describe node <node name>** command as follows:

```
# oc describe node worker0.example.com
Name:               worker0.example.com
...
Labels:             ...
                    ...
                    scale.spectrum.ibm.com/designation=quorum
...
```

To remove a label from a node, enter the following command:

```
oc label node <node name> scale.spectrum.ibm.com/designation-
```

Quorum node designations can be changed on the IBM Storage Scale container native cluster by manually applying or removing node labels. Manual labeling requires insight about IBM Storage Scale and must not be done by unexperienced administrators.

Install

If you are looking to upgrade an existing IBM Storage Scale container native cluster, see Upgrading.

The installation process for IBM Storage Scale container native begins with applying the **install.yaml** to create and define Kubernetes configuration across the following namespaces:

- **ibm-spectrum-scale-operator**

- `ibm-spectrum-scale-dns`
- `ibm-spectrum-scale-csi`
- `ibm-spectrum-scale`

Run the following command to configure the environment:

```
oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/install.yaml
```

Verification

Validate that the following namespaces have been created by running the command:

```
oc get namespaces | grep ibm-spectrum-scale
```

```
$ oc get namespaces | grep ibm-spectrum-scale
ibm-spectrum-scale                                Active   4s
ibm-spectrum-scale-csi                            Active   27s
ibm-spectrum-scale-dns                            Active   26s
ibm-spectrum-scale-operator                       Active   26s
```

Validate that operator pods are running in the following two namespaces:

1. `ibm-spectrum-scale-operator`

   ```
   oc get pods -n ibm-spectrum-scale-operator
   ```

   ```
   $ oc get pods -n ibm-spectrum-scale-operator
   NAME                                                READY   STATUS    RESTARTS   AGE
   ibm-spectrum-scale-controller-manager-78df9cf866-jd89q   1/1     Running   0          78s
   ```

2. `ibm-spectrum-scale-csi`

   ```
   oc get pods -n ibm-spectrum-scale-csi
   ```

   ```
   $ oc get pods -n ibm-spectrum-scale-csi
   NAME                                          READY   STATUS    RESTARTS   AGE
   ibm-spectrum-scale-csi-operator-7f94bfd897-w88fr   1/1     Running   0          40s
   ```

Image pull secrets

If you have already created the OpenShift global pull secret from Adding IBM Cloud Container Registry credentials, skip this section and go onto the next section.

IBM Storage Scale container native images are hosted in IBM Cloud Container Registry.

Create the **`ibm-entitlement-key`** pull secret for each of the following namespaces:

- ibm-spectrum-scale
- ibm-spectrum-scale-dns
- ibm-spectrum-scale-csi

- Obtain an entitlement key from IBM container software library and export it:

  ```
  export ENTITLEMENT_KEY=<REPLACE WITH ICR ENTITLEMENT KEY>
  ```

- Create the docker-registry secret for each namespace:

  ```
  for namespace in ibm-spectrum-scale ibm-spectrum-scale-dns ibm-spectrum-scale-csi; do
      oc create secret docker-registry ibm-entitlement-key -n ${namespace} \
      --docker-server=cp.icr.io \
      --docker-username cp \
      --docker-password ${ENTITLEMENT_KEY}
  done
  ```

- Unset the export:

  ```
  unset ENTITLEMENT_KEY
  ```

Kubernetes resources

The following sections describe Kubernetes resources that need to be defined to the Red Hat OpenShift cluster to drive features of the IBM Storage Scale container native cluster.

The following table lists the Custom Resource Definitions (CRDs) managed by the operator:

Table 1. IBM Storage Scale container native cluster custom resources

Resource	Short name	Description
`cluster`	`gpfs`	Set attributes for the IBM Storage Scale container native cluster.
`callhome`	none	Configures IBM Storage Scale callhome.
`remoteclusters`	`remotegpfs`	Provide configuration details to the IBM Storage Scale remote cluster. For more information, see Remote file system section.
`filesystem`	`fs`	Configure the file systems for the container native cluster.
`localdisk`	`ld`	Configure the disks or volumes to be used as storage for local file systems.
`encryptionconfig`	`ec`	Allows users to configure encryption.

Use the following sections to guide you to create the custom resources:

- Cluster
- Callhome
- RemoteCluster
  - Creating secrets for storage cluster GUI users
  - Configuring Certificate Authority (CA) certificates
- File Systems
  - Remote file system
  - Local file system
    - Local disks
- Encryption

Cluster

A cluster definition is needed to declare the properties of the IBM Storage Scale container native cluster. The following steps describe creating a **`Cluster`** custom resource.

1. Download a copy of the sample **`cluster.yaml`** from the GitHub repository.

   ```
   curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.2.0.x/generated/scale/cr/cluster/cluster.yaml > cluster.yaml || echo "Failed to download Cluster Sample CR"
   ```

2. Make changes specific to your installation. For more information on the Cluster specification, see Cluster spec.

3. Set the labels for the **`nodeSelector`** in the cluster **`spec.daemon.nodeSelector`** field. The **`nodeSelector`** in the sample **`Cluster`** custom resource is defined as **`scale.spectrum.ibm.com/daemon-selector: ""`**. For example, if you want scale core pods to run on the Red Hat OpenShift worker nodes, **`node-role.kubernetes.io/worker`**, then apply the following command to set the **`nodeSelector`** label on the worker nodes:

   ```
   oc label nodes -lnode-role.kubernetes.io/worker= scale.spectrum.ibm.com/daemon-selector=
   ```

4. Apple the cluster **`yaml`** by entering the following command:

   ```
   oc apply -f cluster.yaml
   ```

   If you apply the **`Cluster`** custom resource and do not see any core pods to be created, check that you labeled your nodes with the configured **`daemon.nodeSelector`**.

After the cluster resource is defined to Kubernetes, use **`oc edit cluster ibm-spectrum-scale`** modify properties of the resource.

Cluster spec

The following table describes the properties for the **`Cluster`** custom resource **`spec`**.

Table 1. Cluster property and description

Property	Required	Default	Description
`license`	Yes	Not accepted	The license section allows the user to accept the license and specify the edition of IBM Storage Scale. See License section for more details.
`license.accept`	Yes	`false`	The license must be accepted. Read the license and change to "true" to accept. See License section for more details.

Property	Required	Default	Description
`license.license`	Yes	`data-access`	Specifies the IBM Storage Scale edition, "data-access" or "data-management". See License section for more details.
`daemon`	Yes	N/A	`daemon` specifies the configuration of the IBM Storage Scale daemons. See Daemon section for more details.
`daemon.nodeSelector`	No	`scale.spectrum.ibm.com/daemon-selector`	Allows user to set a `nodeSelector` to indicate which Red Hat OpenShift nodes to run scale core pods. See Daemon section for more details.
`grafanaBridge`	No	Disabled	Specifies the configuration of Grafana bridge. See Grafana bridge section for more details.
`gui`	No	N/A	Specifies extra configuration of the GUI.
`pmcollector`	No	N/A	Specifies extra configuration of the pmcollector.
`networkPolicy`	Yes	Enabled	The operator creates network policy rules in the IBM Storage Scale container native namespaces.

License

The `license` section allows accepting and choosing the IBM Storage Scale edition to be deployed in the IBM Storage Scale container native cluster. Complete the following activities:

- Review the appropriate license documentation through the URL in the CR.
- Accept the license by specifying `true` in the `license.accept` field.
- Supply the edition used in the `license.license` field.

The sample CR defaults to `data-access` under the `license.license` field, indicating IBM Storage Scale Data Access Edition. If you need the IBM Storage Scale Data Management Edition, then change the value in `license.license` to `data-management`.

The `data-access` edition is sufficient if remote mounted file systems are used. To use local file systems, the `data-management` license is required.

Specifying an edition without proper entitlement results in image pull failures during deployment.

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  ...
  license:
    accept: true
    license: data-access
```

Enter the `oc explain cluster.spec.license` command to view more details.

Daemon

The daemon section in the cluster specification specifies configuration for the IBM Storage Scale core pods.

Cluster name override

If you have multiple container native clusters that are attempting to remote mount from a single IBM Storage Scale storage cluster, each client GPFS cluster must have a unique GPFS Cluster name. Depending on the configuration of your Red Hat OpenShift cluster, the default cluster might not be unique and cause failures to mount the file system.

The cluster name is created by taking `ibm-spectrum-scale` and adding it to the base domain of the Red Hat OpenShift cluster (`oc get dns cluster -ojson | jq -r '.spec.baseDomain'`). If this cluster name is not unique, use the `daemon.clusterNameOverride` field to override the generated name.

The `clusterNameOverride` field must be configured before cluster creation. This field cannot be changed after a cluster is created. Setting this field after a cluster is created requires a full cleanup and redeployment of the cluster.

If the following value in the clusterNameOverride field is used:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
...
spec:
  daemon:
    clusterNameOverride: ocp.c2.example.com
```

As a result, the container native cluster that is created has the following GPFS cluster name:

```
mmlssh-4.4# mmlscluster

GPFS cluster information
========================
  GPFS cluster name:         ocp.c2.example.com
```

```
...
...
```

Tolerations

The **spec.daemon.tolerations** section allows the user to configure additional tolerations to determine where IBM Storage Scale core pods can be scheduled. For more information on tolerations, see Taints and Tolerations in Kubernetes documentation.

The operator rolls out the configured Tolerations to the IBM Storage Scale core pods.

Enter the **oc explain cluster.spec.daemon.tolerations** command to view more details.

Node selectors

Use the **daemon.nodeSelector** section to configure a nodeSelector to determine where IBM Storage Scale pods can be deployed.

The Operator checks that a node has all defined labels present to deem a node eligible to deploy IBM Storage Scale pods. In the **Cluster** custom resource sample, the operator deploys IBM Storage Scale pods on nodes with the following label: **scale.spectrum.ibm.com/daemon-selector**:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  ...
  daemon:
    nodeSelector:
      scale.spectrum.ibm.com/daemon-selector: ""
```

Enter the **oc explain cluster.spec.daemon.nodeSelector** command to view more details. For more information, see Compact cluster support.

Host aliases

It is recommended that proper DNS is configured in your environment.

The **daemon.hostAliases** section allows for user-defined entries to be added into the IBM Storage Scale CoreDNS service. The IBM Storage Scale CoreDNS service provides the name resolution for the core pods.

If the core pods are unable to resolve the hostname of the servers in the storage cluster by DNS, their hostname and their IP addresses can be specified in the **hostAliases** as follows:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  ...
  daemon:
    hostAliases:
      - hostname: node1.example.com
        ip: 10.0.0.1
      - hostname: node2.example.com
        ip: 10.0.0.2
```

The IBM Storage Scale CoreDNS service handles name resolution for the core pods. For **RemoteCluster** CR, the hostname that is provided in the **remotecluster.spec.gui.host** field must be DNS resolvable and using host aliases is not a valid workaround.

Enter the **oc explain cluster.spec.daemon.hostAliases** command to view more details.

Cluster profile

The **daemon.clusterProfile** allows the user to set default IBM Storage Scale configuration parameters for the cluster at cluster creation time.

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  daemon:
    ...
    clusterProfile:
      controlSetxattrImmutableSELinux: "yes"
      enforceFilesetQuotaOnRoot: "yes"
      ignorePrefetchLUNCount: "yes"
      initPrefetchBuffers: "128"
      maxblocksize: 16M
      prefetchPct: "25"
      prefetchTimeout: "30"
```

Changing the values in the **clusterProfile** is not supported and must be avoided unless advised by IBM Support. For the two exceptions where changing values in the **clusterProfile** is supported, see Ephemeral port range and Ephemeral port ranges.

Enter the **oc explain cluster.spec.daemon.clusterProfile** command to view more details.

Ephemeral port range

If the storage cluster configures ephemeral port ranges, you need to set **tscCmdPortRange** on the container native cluster to match the range.

For example, if the storage cluster is configured to use port range of 60000-61000, set this value under the **clusterProfile** section in the **Cluster** custom resource.

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  ...
  daemon:
    clusterProfile:
      ...
      tscCmdPortRange: "60000-61000"
```

Roles

The **daemon.roles** section under the **Cluster** spec allows the user to fine-tune memory and CPU requests by using the **resources** object on the nodes that are part of specific IBM Storage Scale roles. For more information, see Resource Management for Pods and Containers in Kubernetes documentation.

client and storage role

Nodes that mount a remote file system of an IBM Storage Scale storage cluster have the **client** role. Nodes that provide disks or volumes to a local file system have the **storage** role.

For the **client** role, the configuration recommendation for requests is 2 CPU and 4 GiB. For the **storage** role, the configuration recommendation for requests is 2 CPU and 8 GiB. On systems with many CPU cores, large memory, and/or high-speed network, the storage performance might increase with higher values. Therefore, encryption and compression of **PersistentVolumes** result in higher CPU load higher resource values can be beneficial. On smaller systems and/or applications with low I/O workloads, 1 CPU and 2 GiB can be set.

Low resource configurations might yield poor performance.

The resource properties of the core pods are as follows:

- Requests, if not specified in the cluster spec for the roles is set to 25% of the capacity of the nodes.
- Limits, if not specified in the cluster spec for the roles is set to 50% of the capacity of the nodes.

For example, to set memory and CPU requests and limits for the **client** role, specify the request values under **spec.daemon.roles.resources** and the limit values under **spec.daemon.roles.limits**. Limits, if specified, must have at least the double value as the requests:

These values must be set at cluster creation time. Changes made after the cluster is created do not take effect until the pods restart.

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  ...
  daemon:
    roles:
    - name: client
      resources:
        cpu: "2"
        memory: 4Gi
      limits:
        cpu: "4"
        memory: 8Gi
    - name: storage
      resources:
        cpu: "2"
        memory: 8Gi
      limits:
        cpu: "4"
        memory: 16Gi
```

For s390x, the sample cluster custom resource ships with "4Gi" memory request. You have to reduce the memory request to "2Gi" if your hardware does not have enough physical memory.

Enter the **oc explain cluster.spec.daemon.roles** command to view more details.

Grafana bridge

The **grafanaBridge** section allows the user to enable the deployment of the IBM Storage Scale bridge for Grafana application. For more information, see IBM Storage Scale bridge for Grafana.

Specify **grafanaBridge: {}** to enable Grafana Bridge:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  ...
  grafanaBridge: {}
```

Enter the **oc explain grafanabridge.spec** command to view more details.

Grafana bridge is optional

Infrastructure nodes

The GUI pods, pmcollector pods, and Grafana bridge pods can be placed onto Red Hat OpenShift infrastructure nodes. At least two infrastructure nodes are needed because two replicas of GUI and pmcollector pods have to run on different infrastructure nodes.

GUI, pmcollector and Grafana bridge pods

The **gui**, **pmcollector** and **grafanaBridge** sections allow to specify a Kubernetes Node Selector and Kubernetes Taints and Tolerations.

If the Red Hat OpenShift infrastructure nodes are labeled with **node-role.kubernetes.io/infra=""** and have the **node-role.kubernetes.io/infra:NoSchedule** and **node-role.kubernetes.io/infra:NoExecute** taints, add the following lines to run the **gui**, **pmcollector** and **grafanaBridge** sections:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Cluster
spec:
  ...
  gui:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
    tolerations:
    - effect: NoSchedule
      key: node-role.kubernetes.io/infra
      operator: Exists
    - effect: NoExecute
      key: node-role.kubernetes.io/infra
      operator: Exists
  pmcollector:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
    tolerations:
    - effect: NoSchedule
      key: node-role.kubernetes.io/infra
      operator: Exists
    - effect: NoExecute
      key: node-role.kubernetes.io/infra
      operator: Exists
  grafanaBridge:
    nodeSelector:
      node-role.kubernetes.io/infra: ""
    tolerations:
    - effect: NoSchedule
      key: node-role.kubernetes.io/infra
      operator: Exists
    - effect: NoExecute
      key: node-role.kubernetes.io/infra
      operator: Exists
```

IBM Storage Scale core pods

To use infrastructure nodes, it is required that IBM Storage Scale core pods also run on the infrastructure nodes.

- Label all worker nodes and infrastructure nodes with a common label. The sample **Cluster** custom resource defines the node selector: **scale.spectrum.ibm.com/daemon-selector: ""**. To use this selector, apply the following labels to worker and infrastructure nodes:

  ```
  oc label nodes --selector node-role.kubernetes.io/worker scale.spectrum.ibm.com/daemon-selector=""
  ```

  ```
  oc label nodes --selector node-role.kubernetes.io/infra scale.spectrum.ibm.com/daemon-selector=""
  ```

The IBM Storage Scale core pods tolerate the **NoExecute** and **NoSchedule** taint. Therefore, the core pods run on infrastructure nodes even if these taints are present.

Network policies

The **cluster.spec.networkPolicy** object indicates whether the container native operator creates and manages the network policy rules within the IBM Storage Scale container native namespaces. The default sample cluster CR enables this option: **networkPolicy: {}**.

Modifying the network policies that are owned by the Cluster controller gets overwritten by the operator.

Enabling and disabling network policy management

With IBM Storage Scale container native v5.1.9 or later, the sample custom resource enables network policies by default.

If network policies are not created by the operator, use the following command to patch the **Cluster** spec:

```
oc patch cluster.scale.spectrum.ibm.com ibm-spectrum-scale --type='json' \
-p='[{"op": "add", "path": "/spec/networkPolicy", "value":{}}]'
```

If you find that the network policy rules are interfering with functions of the container native cluster, disable the option in the **Cluster** spec with the following command:

```
oc patch cluster.scale.spectrum.ibm.com ibm-spectrum-scale --type='json' \
-p='[{"op": "remove", "path": "/spec/networkPolicy"}]'
```

Delete the network policies with the following commands:

```
oc delete networkpolicy -n ibm-spectrum-scale-operator --all
oc delete networkpolicy -n ibm-spectrum-scale --all
oc delete networkpolicy -n ibm-spectrum-scale-dns --all
```

Allowing communication between trusted namespaces

To allow ingress and egress communication to and from a trusted namespace within your Red Hat OpenShift cluster, label the namespace with the label: **scale.spectrum.ibm.com/networkpolicy=allow**.

To enable communication, add the label to the namespace:

```
oc label namespace <TARGET_NAMESPACE> scale.spectrum.ibm.com/networkpolicy=allow
```

To disable communication, remove the label on the namespace:

```
oc label namespace <TARGET_NAMESPACE> scale.spectrum.ibm.com/networkpolicy-
```

Cluster status

Status **Conditions** can be viewed as a snapshot of the current and most up-to-date status of a **Cluster**.

- The **Success** condition is set to **True** if the **Cluster** is successfully configured.

Callhome

To enable call home functionality, create a **CallHome** custom resource to the kubernetes cluster. The following steps describe creating a **Callhome** CR.

1. Download a copy of the sample **callhome.yaml** from the GitHub repository.

   ```
   curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
   native/v5.2.0.x/generated/scale/cr/callhome/callhome.yaml > callhome.yaml || echo "Failed to download
   Callhome Sample CR"
   ```

2. Edit the **callhome.yaml** file and make changes specific to your installation. For details on how to fill out the sections of the Callhome specification, see Callhome spec.

3. After you have made your changes, apply the callhome yaml using the following command:

   ```
   oc apply -f callhome.yaml
   ```

4. View the Callhome resources using the following command:

   ```
   oc get callhome -n ibm-spectrum-scale
   ```

Once deployed, use the command **oc edit callhome -n ibm-spectrum-scale** to modify properties of the resource.

Call home can be enabled, modified, or disabled at any time.

For more information, see Understanding call home in IBM Storage Scale documentation.

Callhome spec

The following table describe the properties for **Callhome**:

Table 1. Callhome property and description

Property	Required	Default	Description
companyEmail	Yes	None	The address of the system administrator who can be contacted by the IBM Support. Usually this e-mail address is directed towards a group or task e-mail address. For example, itsupport@mycompanyname.com.
companyName	Yes	None	The company to which the contact person belongs. This name can consist of any alphanumeric characters and these non-alphanumeric characters are '-', '_', '!', ' ', ';'.
countryCode	Yes	None	The two-letter capital letter country codes as defined in ISO 3166-1 alpha-2.
customerID	Yes	None	The customer ID of the system administrator who can be contacted by the IBM Support. This can consist of any alphanumeric characters and these non-alphanumeric characters are '-', '_', '!'.
license.accept	Yes	None	License must be accepted by the end user to enable Callhome.
proxy	No	None	If specified, defines a proxy server configuration.
proxy.host	Yes, if **proxy** is specified	None	The host of proxy server as hostname or IP address.
proxy.port	Yes, if **proxy** is specified	None	The port of proxy server.
proxy.secretName	Yes, if **proxy** is specified	None	The secret name of a basic authentication secret, which contains username and password for proxy server.

You should only define one **Callhome** resource to the namespace.

License agreement

To agree and accept the license, set **license.accept** property to **true**. If you do not accept the license, call home is not enabled.

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Callhome
...
spec:
  ...
  license:
    accept: true
```

Personal information

Under the **spec** for **Callhome**, enter your **companyName**, the **customerID** that IBM provided to you, the **companyEmail** and the **countryCode**.

The **countryCode** is a two-letter capital letter country codes as defined in ISO 3166-1 alpha-2. For example, **US** for the United States or **DE** for Germany.

Type

Set the **spec.type** to reflect the type of cluster, **test** or **production**.

Proxy (optional)

If you are using a proxy for communication, enter information about the proxy service in the **spec.proxy** field. Enter the **oc explain callhome.spec.proxy** command to view more details.

If your proxy requires authentication, you must create a kubernetes secret containing the credentials. For example, to create a secret **proxyServerSecret**, you can enter the following command:

```
oc create secret generic proxyServerSecret --from-literal=username='<proxy_username>' \
--from-literal=password='<proxy_password>' -n ibm-spectrum-scale
```

Then add your configuration into the CR:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Callhome
...
spec:
  ...
  proxy:
    host: proxyserver.example.com
    port: 443
    secretName: proxyServerSecret
```

Enter the **oc explain callhome** command to view more details.

Callhome status

Status **Conditions** can be viewed as a snapshot of the current and most up-to-date status of **Callhome**.

- The **Enabled** condition is set to **True** if **Callhome** functionality is enabled by accepting the license.
- The **Success** condition is set to **True** if **Callhome** configured successfully and is able to communicate with the IBM Callhome server.

RemoteClusters

To allow the IBM Storage Scale container native cluster to access remote IBM Storage Scale storage clusters, a **RemoteCluster** custom resources (CR) must be defined for each storage cluster. For more information to prepare the storage cluster, see Storage cluster.

To help with fields in the **RemoteCluster** custom resource specification, see:

- Creating secrets for storage cluster GUI users
- Configuring Certificate Authority (CA) certificates

The following steps help guide you in creating a **RemoteCluster** resource.

1. Download a copy of the sample **remotecluster.yaml** from the GitHub repository.

   ```
   curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.2.0.x/generated/scale/cr/remotecluster/remotecluster.yaml > remotecluster.yaml || echo "Failed to download RemoteCluster Sample CR"
   ```

   The sample **remotecluster.yaml** uses **remotecluster-sample** as the resource name. You can change this name. Each resource must have a unique **metadata.name** when multiple remote cluster resources are created.

2. Edit the **remotecluster.yaml** file and change the fields that are specific to your installation. For details on the Remote Cluster specification, see RemoteCluster spec.

3. Apply the resource by using the following command:

   ```
   oc apply -f remotecluster.yaml
   ```

4. View the remote cluster resources by using the following command:

   ```
   oc get remotecluster -n ibm-spectrum-scale
   ```

After the resource is defined to Kubernetes, use **oc edit remotecluster <remotecluster-name> -n ibm-spectrum-scale** to modify the properties of the resource.

RemoteCluster spec

The following table describes the properties for **RemoteCluster**:

Table 1. Remotecluster property and description

Property	Required	Default	Description
metadata.name	Yes	None	The name of the CR. This field is used to identify the remote storage cluster in the **Filesystem** CR.
contactNodes	No	None	This property is optional and provides a list of nodes from the storage cluster to be used as the remote cluster contact nodes. The names must be the daemon node names. If not specified, the operator uses all quorum nodes detected from the storage cluster. If the contact node names are not resolvable in local cluster, configure the 'hostAliases' with node name and IP in Cluster CR.
gui	Yes	None	It specifies the details for the IBM Storage Scale Remote Cluster GUI.
gui.cacert	No	None	It specifies the name of the RootCA configmap.
gui.csiSecretName	Yes	csi-remote-mount-storage-cluster-1	It references the secret that contains the username and password of the CSI admin user in the **ibm-spectrum-scale-csi** namespace.
gui.host	No	None	The host references a REST API endpoint on the remote cluster. This host must be resolvable by DNS. This field is DEPRECATED and will be removed in a future release. Use the 'hosts' field instead.
gui.hosts	No	None	The hosts references one or more REST API endpoints on the remote cluster. Up to 3 endpoints can be specified. The hosts must be resolvable by DNS.

Property	Required	Default	Description
`gui.insecureSkipVerify`	No	None	The parameter controls whether a client verifies the storage cluster's GUI certificate chain and hostname. If set to `true`, TLS is susceptible to machine-in-the-middle attacks. The default value is `false`.
`gui.port`	No	443	It specifies the port of the Remote Cluster.
`gui.scheme`	No	https	The default value is 'https'. No other value is supported.
`gui.secretName`	Yes	None	The name of the Kubernetes secret created during the storage cluster configuration.

You can define 1 or more `RemoteClusters` to the cluster, one for each Storage Cluster you want to mount file systems from.

Either `gui.host` or `gui.hosts` is mandatory in `Remotecluster` CR. Use `gui.hosts` instead of `gui.host`, as `gui.host` is deprecated and will get removed in a future release.

Contact Nodes

The `spec.contactNodes` field allows the administrator to specify the exact list of contact nodes from the storage cluster to be used in the remote cluster configuration. If none is provided, the operator picks all quorum nodes from querying the remote cluster REST API. The hostnames for the contact nodes need to be DNS resolvable within the core pods in the Red Hat OpenShift cluster. If DNS resolution does not work, you can specify the hostname and IP address mapping by using the `hostAlias` field in the Cluster Spec. For more information, see `hostAlias`.

GUI

The `spec.gui` contains the properties that are needed to communicate with the IBM Storage Scale remote storage cluster.

- `cacert` - provide the name of the ConfigMap storing the CA Certificate for the Storage Cluster GUI. For more information, see Configuring Certificate Authority (CA) certificates.
- `insecuritySkipVerify` - controls whether a client verifies the server's certificate chain and hostname. Default to false.
- `host` - provide the Remote Cluster GUI host endpoint. This field is deprecated. Use `hosts` instead.
- `hosts` - provide the Remote Cluster GUI host endpoints. Multiple (up to 3) GUI host endpoints can be specified when the GUI is configured on multiple nodes of the remote storage cluster. Use this property to get high availability of the GUI.
- `secretName` - The name of the Kubernetes secret that contains the credentials for the ContainerOperator user on the storage cluster
- `csiSecretName` - The name of the Kubernetes secret that contains the credentials for the CsiAdmin user on the storage cluster

Deleting a `RemoteCluster` custom resource definition does not delete the access permission of the IBM Storage Scale container native cluster to the file systems on the remote storage cluster.

Enter the `oc explain remotecluster.spec` command to view more details.

RemoteCluster status

Status `Conditions` can be viewed as a snapshot of the current and most up-to-date status of a `Remotecluster` instance.

- The `Ready` condition is set to `True` if the `Remotecluster` credentials are established.

Examples

Additional remote clusters

It is possible to mount other file systems that are served by multiple remote clusters. For each remote cluster, a separate resource must be defined into the Kubernetes cluster.

If applying the sample `remotecluster.yaml` CR from the examples, you would have a single resource defined:

```
$ oc get remotecluster -n ibm-spectrum-scale
NAME                   READY   AGE
remotecluster-sample   True    25h
```

Duplicate the sample `remotecluster` CR and make changes to reflect your second remote storage cluster. The `metadata.name` must be changed to be unique, in this example, we use `xyz-storagecluster`. Create the new resource and verify that it is defined:

```
$ oc get remotecluster -n ibm-spectrum-scale
NAME                   READY   AGE
remotecluster-sample   True    25h
xyz-storagecluster     True    10m
```

Creating secrets for storage cluster GUI users

Create a secret on the Red Hat OpenShift cluster that holds the credentials for GUI users defined on the IBM Storage Scale Storage cluster. The secret is used by the operator to communicate with the storage cluster to configure the remote mount.

The username and password that is specified for the secrets must match the GUI user that was created on the storage cluster Creating container operator user and group.

Two new secrets must be added for each storage cluster being configured.

1. Create a secret for the **ContainerOperator** GUI user defined on the storage cluster.

   To create a secret named **cnsa-remote-mount-storage-cluster-1** in the **ibm-spectrum-scale** namespace, enter the following command:

   ```
   oc create secret generic cnsa-remote-mount-storage-cluster-1 --from-
   literal=username='cnsa_storage_gui_user' \
   --from-literal=password='cnsa_storage_gui_password' -n ibm-spectrum-scale
   ```

2. Create a secret for the **CsiAdmin** GUI user defined on the storage cluster.

   To create the secret named **csi-remote-mount-storage-cluster-1** in the **ibm-spectrum-scale-csi** namespace, enter the following command:

   ```
   oc create secret generic csi-remote-mount-storage-cluster-1 --from-
   literal=username='csi_storage_gui_user' \
   --from-literal=password='csi_storage_gui_password' -n ibm-spectrum-scale-csi
   ```

3. Label the secret, enter the following command:

   ```
   oc label secret csi-remote-mount-storage-cluster-1 -n ibm-spectrum-scale-csi product=ibm-spectrum-
   scale-csi
   ```

When the passwords on the storage cluster for these users change, the credentials in the secrets must be updated. For instructions to update the secrets, see Updating user secrets for the storage cluster on Red Hat OpenShift.

Configuring Certificate Authority (CA) certificates

IBM Storage Scale container native uses Transport Layer Security (TLS) verification to guarantee secure HTTPS communication with the storage cluster GUI. It verifies the server's certificate chain and host name.

Configure a security protocol

A security protocol must be configured for use with IBM Storage Scale container native in one of three different ways.

Option 1 - CA certificate ConfigMap

A ConfigMap containing the CA certificate of the storage cluster GUI must be created to allow the IBM Storage Scale container native operator to perform TLS verification. CA certificate data can exist in base64 encoded or decoded forms.

In the following example, we create a ConfigMap from **storage-cluster-1.crt** file. This file contains the storage cluster CA certificate data in decoded form. The decoded form must appear as shown:

```
# cat storage-cluster-1.crt
-----BEGIN CERTIFICATE-----
MIIDZDC.....................................................
.............................................................
...........n/J9OJFdoXs=
-----END CERTIFICATE-----
```

Create the ConfigMap with one of the following two commands. The second command is provided to assist the users who wish to trust the self-signed certificate of the storage cluster GUI.

```
oc create configmap cacert-storage-cluster-1 --from-file=storage-cluster-1.crt=storage-cluster-1.crt -n ibm-
spectrum-scale
```

By default, the storage cluster GUI self-signs a certificate that can be used in lieu of a CA certificate. This certificate can be obtained and used to create the **cacert** ConfigMap by entering the following command. Replace the **gui host** with the hostname of the storage cluster GUI.

```
oc create configmap cacert-storage-cluster-1 --from-literal=storage-cluster-1.crt="$(openssl s_client -
showcerts -connect <gui host>:443 </dev/null 2>/dev/null|openssl x509 -outform PEM)" -n ibm-spectrum-scale
```

Option 2 - Storage cluster uses the OpenShift Container Platform CA or a Red Hat default CA

IBM Storage Scale container native automatically includes the OpenShift Container Platform CA and the default Red Hat CA bundle for storage cluster GUI communication. If the storage cluster uses the OpenShift Container Platform CA or a Red Hat trusted CA, a ConfigMap, as described in Option 1, does **not** need to be created for the CA certificate and the **cacert** field should be deleted from the RemoteCluster Custom Resource. For more information, see RemoteClusters.

Option 3 - Skip verification

Storage cluster verification may be skipped if desired, however, TLS is susceptible to machine-in-the-middle attacks. To skip verification, the **insecureSkipVerify** option must be set to **true**, when configuring the RemoteCluster Custom Resource. For more information, see RemoteClusters.

Storage cluster verification

Events are posted onto the **RemoteCluster** resource if configuration is missing. For example, if secrets and ConfigMaps are missing, you may see events similar to the following sample:

```
$ oc describe remotecluster.scale remotecluster-sample
...
Events:
  Type     Reason           Age                From           Message
  ----     ------           ----               ----           -------
  Warning  RemoteConnError  6m3s               RemoteCluster  Secret "cnsa-remote-mount-storage-cluster-1"
not found
  Warning  RemoteConnError  3s (x6 over 5m3s)  RemoteCluster  ConfigMap "cacert-storage-cluster-1" not found
```

Filesystem

The **Filesystem** custom resource supports the following file systems:

- Remote file systems are mounted from a remote IBM Storage Scale storage cluster.
- Local file systems (technology preview) use local disks or volumes of Red Hat OpenShift nodes as the storage.

Remote file system

To configure a file system in the IBM Storage Scale container native cluster, a **Filesystem** custom resource (CR) must be defined for each file system you want mounted.

An IBM Storage Scale remote storage cluster serves the file system. Create a **RemoteCluster** custom resource for that storage cluster before proceeding. For more information, see RemoteCluster.

The following steps guide you in creating a **Filesystem** resource.

1. Download a copy of the sample **filesystem.remote.yaml** from the GitHub repository.

   ```
   curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
   native/v5.2.0.x/generated/scale/cr/filesystem/filesystem.remote.yaml > filesystem.remote.yaml || echo
   "Failed to download Filesystem Sample CR"
   ```

2. Edit the **filesystem.remote.yaml** file and change the fields that are specific to your installation. For more information, see Filesystem Spec.

3. Use the following command to apply the changed yaml:

   ```
   oc apply -f filesystem.remote.yaml
   ```

4. View the **Filesystem** resources with the following command:

   ```
   oc get filesystem -n ibm-spectrum-scale
   ```

Use the command **oc edit filesystem <filesystem-name> -n ibm-spectrum-scale** to modify the properties of the resource.

Create an **EncryptionConfig** custom resource to encrypt the remote mounted file system for the IBM Storage Scale container native cluster. For more information, see EncryptionConfig.

Filesystem spec

The following table describes the properties for **Filesystem**:

Table 1. Filesystem property and description

Property	Required	Default	Description
`metadata.name`	Yes	None	The name of the CR.
`remote`	No	None	If specified, describes the file system to be a remote mounted file system.
`remote.fs`	Yes, if `remote` is specified	None	It is the name of the file system to mount, served by the remote cluster.
`remote.cluster`	Yes, if `remote` is specified	None	It is the name of the `RemoteCluster` resource.

You can define 1 or more `Filesystem` custom resources, one for each file system that you want to mount from the storage cluster.

All file systems are mounted under `/mnt`. The mount path cannot be changed.

remote

The `spec.remote` section defines the remote file system properties and consists of two fields:

- `remote.cluster`: This field specifies the name of the `RemoteCluster` CR that is defined that is serving the file system.
- `remote.fs`: This field specifies the file system name on the remote storage cluster that is mounted into the container native cluster.

In the following example:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Filesystem
...
spec:
  remote:
    cluster: remotecluster-sample
    fs: fs1
```

The file system `fs1` provided by the remote cluster, which is defined in `remotecluster-sample` is made available in the container.

If a `Filesystem` custom resource is deleted, the file system configuration from the IBM Storage Scale Cluster **is not** deleted.

Enter the `oc explain filesystem.spec.remote` command to view more details.

File system status

Status `Conditions` can be viewed as a snapshot of the current and most up-to-date status of a `Filesystem` instance.

- The `Success` condition is set to `True` if the `Filesystem` is created and mounted.

Examples

More file systems

To create more file systems int the container native cluster, define a custom resource for each file system and apply it to the Kubernetes cluster.

Following the prior examples results in a single file system resource:

```
$ oc get filesystem -n ibm-spectrum-scale
NAME            ESTABLISHED   AGE
remote-sample   True          25h
```

1. To create another file system that is served by the **same** remote cluster, `remotecluster-sample`, define a yaml as follows:

   ```
   apiVersion: scale.spectrum.ibm.com/v1beta1
   kind: Filesystem
   metadata:
     labels:
       app.kubernetes.io/instance: ibm-spectrum-scale
       app.kubernetes.io/name: cluster
     name: c1-fs2
     namespace: ibm-spectrum-scale
   spec:
     remote:
       cluster: remotecluster-sample
       fs: fs2
   ```

   When applied, the file system `fs2` hosted by the remote cluster `remotecluster-sample` is made available to the container native cluster.

   Displaying the `Filesystem` custom resources shows the following output:

   ```
   $ oc get filesystem -n ibm-spectrum-scale -o wide
   NAME            ESTABLISHED   REMOTE CLUSTER          MAINTENANCE MODE   AGE
   remote-sample   True          remotecluster-sample    not supported      25h
   c1-fs2          True          remotecluster-sample    not supported      10m
   ```

2. To create another file system that is served by a **different** remote cluster than **remotecluster-sample-2**, define a yaml as follows:

   As a prerequisite a RemoteCluster resource that is called **remotecluster-sample-2** must exist. For more information, see RemoteClusters.

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Filesystem
metadata:
  labels:
    app.kubernetes.io/instance: ibm-spectrum-scale
    app.kubernetes.io/name: cluster
  name: xyz-fs2
  namespace: ibm-spectrum-scale
spec:
  remote:
    cluster: remotecluster-sample-2
    fs: fs2
```

   When applied, the file system **fs2** hosted by the remote cluster **remotecluster-sample-2** is made available to the container native cluster.

   Displaying the **Filesystem** custom resources shows the following output:

```
$ oc get filesystem -n ibm-spectrum-scale -o wide
NAME            ESTABLISHED   REMOTE CLUSTER            MAINTENANCE MODE   AGE
remote-sample   True          remotecluster-sample      not supported      25h
c1-fs2          True          remotecluster-sample      not supported      20m
xyz-fs2         True          remotecluster-sample-2    not supported      10m
```

Local file system

This feature is available as a technology preview. Technology preview features are not supported for use within production environments. Use with nonproduction workloads, in demo or proof of concept environments only. IBM production service level agreements (SLAs) are not supported. Technology preview features may not be functionally complete. The purpose of technology preview features is to give access to new and exciting technologies, enabling customers to test functionality and provide feedback during the development process. The existence of a technology preview feature does not mean a future release is guaranteed. Feedback is welcome and encouraged.

To configure a local file system in the IBM Storage Scale container native cluster, a **Filesystem** custom resource (CR) must be defined for each local file system.

This type of file system uses disks or volumes of the Red Hat OpenShift nodes. A **LocalDisk** custom resource must be created for each disk or volume before proceeding. For more information, see LocalDisk.

The following steps describe how to create a **Filesystem** resource.

1. Download a copy of the sample **filesystem.local.yaml** from the GitHub repository.

```
curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/cr/filesystem/filesystem.local.yaml > filesystem.local.yaml || echo
"Failed to download Filesystem Sample CR"
```

2. Edit the **filesystem.local.yaml** file to change the fields that are specific to your installation. For more information, see Filesystem Spec.

3. After you complete the changes, use the following command to apply the yaml:

```
oc apply -f filesystem.local.yaml
```

4. Use the following command to view the **Filesystem** resources:

```
oc get filesystem -n ibm-spectrum-scale -o wide
```

   To view the detailed properties of a local filesystem named **local-sample**:

```
oc describe filesystem local-sample -n ibm-spectrum-scale
```

After the **Filesystem** resource is deployed, use the command **oc edit filesystem <filesystem-name> -n ibm-spectrum-scale** to modify the properties of the resource.

Create an **EncryptionConfig** custom resource to encrypt the remote mounted file system for the IBM Storage Scale container native cluster. For more information, see EncryptionConfig.

Filesystem spec

The following table describes the properties for **Filesystem**:

Table 1. LocalDisk property and description

Property	Required	Default	Description
metadata.name	Yes	None	The name of the CR.
local	No	None	If specified, describes the file system to be a local file system.
local.pools	Yes, if **local** is specified	None	List of file system pools. A local file system must have at least one pool with name **system**.
local.pools.name	Yes, if **local** is specified	None	It is the name of the pool. One pool with name **system** is mandatory.
local.pools.disks	Yes, if **local** is specified	None	The names of **LocalDisk** resources that must provide storage to this local file system pool.
local.pools.capacity	No	None	This field is unused.
local.blockSize	No	**4M**	A data block size of 4 MiB provides good sequential performance, makes efficient use of disk space, and provides good performance for small files. It works well for the widest variety of workloads.
local.replication	No	**3-way**	Replication is the number of replicas to create for each data/metadata block that is written to the file system. Only **3-way** is supported.

You can define 1 or more **Filesystem** custom resources, each file system must use its own local disks.

In the following example:

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: Filesystem
  name: local-sample
spec:
  local:
    pools:
    - disks:
      - worker0-sdb
      - worker0-sdc
      - worker1-sdb
      - worker1-sdc
      - worker2-sdb
      - worker2-sdc
      name: system
---
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: LocalDisk
metadata:
  name: worker0-sdb
spec:
  node: worker0.example.com
  device: /dev/sdb
---
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: LocalDisk
metadata:
  name: worker0-sdc
spec:
  node: worker0.example.com
  device: /dev/sdc
---
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: LocalDisk
metadata:
  name: worker1-sdb
spec:
  node: worker1.example.com
  device: /dev/sdb
---
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: LocalDisk
metadata:
  name: worker1-sdc
spec:
  node: worker1.example.com
  device: /dev/sdc
---
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: LocalDisk
metadata:
  name: worker2-sdb
spec:
  node: worker2.example.com
```

```
  device: /dev/sdb
---
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: LocalDisk
metadata:
  name: worker2-sdc
spec:
  node: worker2.example.com
  device: /dev/sdc
```

The local file system **local-sample** uses the **/dev/sdb** and **/dev/sdc** disks of 3 Red Hat OpenShift nodes.

Enter the **oc explain filesystem.spec.local** command to view more details.

Maintenance Mode

The maintenance mode for a local file system can be enabled by adding the **scale.spectrum.ibm.com/maintenanceMode=true** label to the **Filesystem** CR:

```
oc label filesystem local-sample scale.spectrum.ibm.com/maintenanceMode=true
```

Remove the **scale.spectrum.ibm.com/maintenanceMode** label from the **Filesystem** custom resource to disable the maintenance mode.

```
oc label filesystem local-sample scale.spectrum.ibm.com/maintenanceMode-
```

Alternatively the label value can be set to **false**:

```
oc label filesystem local-sample scale.spectrum.ibm.com/maintenanceMode=false
```

All I/O on the file system must be stopped before the maintenance mode is enabled.

Filesystem status

Status **Conditions** can be viewed as a snapshot of the current and most up-to-date status of a **Filesystem** instance.

- The **Success** condition is set to **True** if the file system is created and mounted.

The following table describes the status fields for **Filesystem**:

Table 2. LocalDisk status fields and description

Status field	Description
pools	The pools of the file system.
pools.name	The name of the file system pool.
pools.failureGroups	When data blocks are written to the local file system, one copy of each block is written to a disk within a different failure group. A file system pool must have disks in at least 3 failure groups to fulfill requirements for 3-way replication.
pools.failureGroups.diskCount	The number of **LocalDisk** objects that are part of the failure group.
pools.failureGroups.disks	The names of the **LocalDisk** objects that are part of the failure group.
pools.failureGroups.failureGroup	A number that represents the failure group.
pools.failureGroups.failureGroupMapping	A description of how the disks for the failure group are selected.
pools.failureGroups.totalDiskSize	Total capacity of all disks within this failure group. Ideally, each failure group within a file system pool has the same total capacity.
uid	The file system UID.
version	The product version for which the file system is compatible.
maintenanceMode	Indicates whether the maintenance mode for this file system is enabled or disabled.

LocalDisks

This feature is available as a technology preview. Technology preview features are not supported for use within production environments. Use with nonproduction workloads, in demo or proof of concept environments only. IBM production service level agreements (SLAs) are not supported. Technology preview features may not be functionally complete. The purpose of technology preview features is to give access to new and exciting technologies, enabling customers to test functionality and provide feedback during the development process. The existence of a technology preview feature does not mean a future release is guaranteed. Feedback is welcome and encouraged.

Labels

Storage nodes

Red Hat OpenShift nodes that have the disks or volumes to be used for local file systems must be labeled with **scale.spectrum.ibm.com/role=storage**:

```
oc label node worker0.example.com scale.spectrum.ibm.com/role=storage
```

These nodes have the storage role, while other nodes have the client role. The node roles are used by the Operator for applying different configurations.

Failure groups

The following mechanisms are supported to assign IBM Storage Scale container native nodes to failure groups:

- Automatic Failure Group Assignment - Allows the Operator to assign storage nodes to failure groups automatically.
- Manual Failure Group Assignment - Allows administrators to have more control of the placement of storage nodes to failure groups.

Automatic failure group assignment

If the user does not label any storage nodes as failure group (see Manual Failure Group Assignment), the Operator automatically assigns storage nodes to failure groups. Kubernetes zones are considered whether they are configured in the Red Hat OpenShift cluster. The operator defines a separate failure group for all storage nodes within the same zone. The storage nodes must be assigned to at least three Kubernetes zones. If no zones or fewer than three zones are defined, the Operator automatically defines a separate failure group per storage node.

Manual failure group assignment

Some times Automatic Failure Group Assignment does not match to the failure topology of the cluster. In this case, the storage nodes can be labeled to create failure groups.

To apply a label to a node, enter the **oc label node <node name> scale.spectrum.ibm.com/nsdFailureGroup=<failure group number>** command as follows:

```
oc label node worker0.example.com scale.spectrum.ibm.com/nsdFailureGroup=0
```

All nodes with the same failure group number belong to the same failure group.

Custom resource

A **LocalDisk** custom resource must be created for each disk or volume of an Red Hat OpenShift node that is used to provide storage for a local file system.

The following steps help guide you in creating a **LocalDisk** resource.

1. Download a copy of the sample **localdisk.yaml** from the GitHub repository.

   ```
   curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.2.0.x/generated/scale/cr/localdisk/localdisk.yaml > localdisk.yaml || echo "Failed to download LocalDisk Sample CR"
   ```

   The sample **localdisk.yaml** provides a list of sample disks that must be changed. Each LocalDisk resource must have a unique **metadata.name**.

2. Edit the **localdisk.yaml** file and change the fields that are specific to your installation. For more information, see LocalDisk Spec.

3. Apply the resource by using the following command:

   ```
   oc apply -f localdisk.yaml
   ```

4. View the localdisk resources by using the following command:

   ```
   oc get localdisk -n ibm-spectrum-scale
   ```

After the resource is defined to Kubernetes, use **oc edit localdisk <localdisk-name> -n ibm-spectrum-scale** to modify the properties of the resource.

LocalDisk spec

The following table describes the properties for **LocalDisk**:

Table 1. LocalDisk property and description

Property	Required	Default	Description

Property	Required	Default	Description
`metadata.name`	Yes	None	The name of the CR.
`device`	Yes	None	The device path that is used at creation time. The device path of a disk can be changed if required.
`existingDataSkipVerify`	No	False	This property controls whether existing Storage Scale data structure is overwritten when the disk is created. A disk can have Storage Scale data structures on if it was used in a file system before and if it was not cleaned up. If false, a "DiskHasFilesystemData" event is displayed if the disk still has Storage Scale data on it, and the disk is not used. If true, the disk is formatted, no matter if it still has data on it.
`node`	Yes	None	The Kubernetes node name of the node where the local disk lives.
`thinDiskType`	No	no	The space reclaim disk type of IBM Storage Scale disks.

LocalDisk status

Status `Conditions` can be viewed as a snapshot of the current and most up-to-date status of a `LocalDisk` instance.

- The `DiskCreated` condition is set to `True` if the `LocalDisk` successfully created a disk within the IBM Storage Scale cluster.
- The `DiskUsed` condition is set to `True` if the disk is used within a local file system.
- The `DiskUp` condition is set to `True` if the disk within the IBM Storage Scale cluster is operational.

Encryption

IBM Storage Scale container native supports remote mount of an encrypted filesystem. Encryption is managed through use of encryption keys stored on key server.

The following key servers are supported:

- IBM Security Guardium Key Lifecycle Manager (SKLM)

EncryptionConfig spec

The following table describe the properties for `EncryptionConfig`:

Table 1. EncryptionConfig property and description

Property	Required	Default	Description
`metadata.name`	Yes	None	The name of the CR.
`server`	Yes	None	The key server host name or IP in which encryption keys are stored.
`backupServers`	No	None	The backup key servers configured for high availability. This field is optional.
`port`	No	None	It can be used to override the default port for the key server.
`cacert`	No	None	The ConfigMap storing CA and endpoint certificates used while adding/renewing key server certificate chain.
`secret`	Yes	None	The name of the basic-auth secret containing the username and password to the key server.
`tenant`	Yes	None	The tenant name on the key server that contains encryption keys. This has to be the same tenant name that is used to store the encryption keys of the remote storage file system.
`client`	Yes	None	The key client to communicate with the key Server.
`remoteRKM`	Yes	None	The RKM ID from the storage cluster corresponding to given key server and tenant.

Deleting a `EncryptionConfig` custom resource does **not delete** the encryption configuration from the IBM Storage Scale container native cluster.

Updating `client` and `tenant` is not recommended as it causes loss of master encryption keys for that tenant.

For more information, enter the `oc explain encryptionconfig.spec` command.

EncryptionConfig status

Status `Conditions` can be viewed as a snapshot of the current and most up-to-date status of a `EncryptionConfig` instance.

- The `Success` condition is set to `True` if the `EncryptionConfig` is successfully configured.

Examples

To give IBM Storage Scale container native access to the encryption keyserver, an **EncryptionConfig** custom resource must be created. The configuration must add the same key server and tenant as configured on storage cluster hosting the filesystem. You can define more than one **EncryptionConfig** custom resource.

For more information, see Encryption in IBM Storage Scale documentation.

Pre-requisites

- Create a secret containing the administrator username and password credentials to the key server.

```
oc create secret generic keyserver-credentials -n ibm-spectrum-scale \
--from-literal=username=<keyserver_admin_name> \
--from-literal=password=<keyserver_admin_password>
```

- If using CA certificates, create the **ConfigMap** holding the CA certificate chain.

  1. Obtain CA certificates and endpoint/server certificates. Separate the root certificate and the intermediate certificates into the following **.crt** files:

     - Root certificate: **root.crt**
     - Server or Endpoint certificate: **endpoint.crt**
     - Intermediate certificates: **intermediate<numeric_index>.crt**

  2. Create the ConfigMap with the following command:

```
oc create ConfigMap sample-ca-cert \
--from-file=/path/to/root.crt \
--from-file=/path/to/intermediate1.crt \
--from-file=/path/to/intermediate2.crt \
--from-file=/path/to/intermediate3.crt \
--from-file=/path/to/endpoint.crt \
-n ibm-spectrum-scale
```

- Encryption details from storage cluster, specifically, **tenant** and **RKMID**.

If the core pods are unable to resolve the IP address of the IBM SKLM server, you can add **hostAliases** entries in the **Cluster** custom resource. For more information, see Cluster.

Configure EncryptionConfig custom resource

The following steps describe creating a **EncryptionConfig** CR.

1. Download a copy of the sample **encryptionconfig.remote.yaml** from the GitHub repository.

```
curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/cr/encryptionconfig/encryptionconfig.remote.yaml >
encryptionconfig.remote.yaml || echo "Failed to download EncryptionConfig Sample CR"
```

2. Make changes specific to your installation. For more information about the **EncryptionConfig** specification, see EncryptionConfig spec.

   - Replace **keyserver.example.com** with your keyserver hostname
   - Replace **keyserver1.example.com, keyserver2.example.com**, etc with your backup keyserver hostnames
   - Replace **sampleTenant** with your tenant name
   - Replace **sampleClient** with your client name
   - Replace **sampleRKM** with your RKMID
   - If using self-signed certificates, comment out the **cacert** field in the spec

3. Apply the cluster yaml using the following command:

```
oc apply -f encryptionconfig.remote.yaml
```

Once deployed, use the command **oc edit encryptionconfig <encryptionconfig-name> -n ibm-spectrum-scale** to modify properties of the resource.

Validating installation

The following sections will help validate the installation.

- Verifying the IBM Storage Scale container native cluster
- Status and events

Verifying the IBM Storage Scale container native cluster

Verify whether the deployment of the IBM Storage Scale container native cluster is done correctly.

Complete the following steps:

For more information, see Debugging IBM Storage Scale deployment.

1. Verify that the Operator has created the cluster by checking the pods.

   ```
   oc get pods -n ibm-spectrum-scale
   ```

   A sample output is shown:

   ```
   # oc get pods -n ibm-spectrum-scale
   NAME                                 READY   STATUS    RESTARTS   AGE
   ibm-spectrum-scale-gui-0             4/4     Running   0          5m45s
   ibm-spectrum-scale-gui-1             4/4     Running   0          2m9s
   ibm-spectrum-scale-pmcollector-0     2/2     Running   0          5m15s
   ibm-spectrum-scale-pmcollector-1     2/2     Running   0          4m11s
   worker0                              2/2     Running   0          5m43s
   worker1                              2/2     Running   0          5m43s
   worker3                              2/2     Running   0          5m45s
   ```

   The following list includes considerations about the IBM Storage Scale cluster creation and its pods:

   - The cluster takes some time to create.
   - One core pod per node gets created on nodes matching the `nodeSelector`.
   - Core pods can take several minutes to move to Running status.
   - GUI pods do not achieve the Running status until all the core pods are in a Running status.
   - Two GUI pods are created, where the second is created after the first is moved to Running status.
   - Two pmcollector pods are created, where the second is created after the first is moved to Running status.
   - Resulting cluster should have one core pod per node as specified by the `nodeSelector`, two GUI pods, and two pmcollector pods.

2. Verify that the IBM Storage Scale cluster is created correctly:

   a) Enter the `mmlscluster` command:

   ```
   oc exec $(oc get pods -lapp.kubernetes.io/name=core \
   -ojsonpath="{.items[0].metadata.name}" -n ibm-spectrum-scale)  \
   -c gpfs -n ibm-spectrum-scale -- mmlscluster
   ```

   The output from the command should show that an IBM Storage Scale cluster has been created, and all nodes as specified by the `nodeSelector` are present.

   ```
   GPFS cluster information
   ========================
   GPFS cluster name:         ibm-spectrum-scale.mycluster.example.com
   GPFS cluster id:           835278197609441888
   GPFS UID domain:           ibm-spectrum-scale.mycluster.example.com
   Remote shell command:      /usr/bin/ssh
   Remote file copy command:  /usr/bin/scp
   Repository type:           CCR

   Node   Daemon node name                                                IP address    Admin node name
Designation
   ------------------------------------------------------------------------------------------------------------
------------------------------------------------------------------
      1   worker2.daemon.ibm-spectrum-scale.stg.mycluster.example.com.  172.29.0.145
worker2.admin.ibm-spectrum-scale.stg.mycluster.example.com.  quorum-manager-perfmon
      2   worker1.daemon.ibm-spectrum-scale.stg.mycluster.example.com.  172.29.0.146
worker1.admin.ibm-spectrum-scale.stg.mycluster.example.com.  quorum-manager-perfmon
      3   worker3.daemon.ibm-spectrum-scale.stg.mycluster.example.com.  172.29.0.148
worker3.admin.ibm-spectrum-scale.stg.mycluster.example.com.  quorum-manager-perfmon
   ```

   b) Enter the `mmgetstate` command:

   ```
   oc exec $(oc get pods -lapp.kubernetes.io/name=core \
   -ojsonpath="{.items[0].metadata.name}" -n ibm-spectrum-scale)  \
   -c gpfs -n ibm-spectrum-scale -- mmgetstate -a
   ```

   The output from the command should show that the `GPFS state` for all nodes are listed as `active`.

   ```
   Node number  Node name        GPFS state
   -----------------------------------------
         1      worker0          active
         2      worker1          active
         3      worker3          active
   ```

3. Verify that the Remote Cluster authentication is successfully created.

   a. Get a list of the remote clusters.

   ```
   oc get remotecluster.scale -n ibm-spectrum-scale
   ```

   b. Inspect the remote clusters and ensure that the value for **READY** is **True**.

   Example:

   ```
   # oc get remotecluster.scale -n ibm-spectrum-scale
   NAME                    READY   AGE
   remotecluster-sample    True    30h
   ```

4. Verify that the storage cluster file system is configured:

   a. Get a list of the file systems:

   ```
   oc get filesystem.scale -n ibm-spectrum-scale
   ```

   b. Inspect the file systems and ensure that the value for **ESTABLISHED** is **True**.

   ```
   $ oc get filesystem.scale -n ibm-spectrum-scale
   NAME            ESTABLISHED   AGE
   remote-sample   True          30h
   ```

5. Manually verify that the file system is mounted using the **mmlsmount** command.

   ```
   oc exec $(oc get pods -lapp.kubernetes.io/name=core \
   -ojsonpath="{.items[0].metadata.name}" -n ibm-spectrum-scale)  \
   -c gpfs -n ibm-spectrum-scale -- mmlsmount remote-sample -L
   ```

   Example output:

   ```
   File system remote-sample (gpfs1.local:fs1) is mounted on ...
   ...
   172.29.0.148    worker3.daemon.ibm-spectrum-scale.stg.mycluster.example.com. ibm-spectrum-
   scale.mycluster.example.com
   172.29.0.146    worker1.daemon.ibm-spectrum-scale.stg.mycluster.example.com. ibm-spectrum-
   scale.mycluster.example.com
   172.29.0.145    worker2.daemon.ibm-spectrum-scale.stg.mycluster.example.com. ibm-spectrum-
   scale.mycluster.example.com
   ```

6. Verify that there are no problems reported in the operator status and events. For more information, see Status and events.

7. Verify that the CSI pods are up and running.

   ```
   oc get pods -n ibm-spectrum-scale-csi
   ```

8. Verify that the Core DNS pods are up and running. There will be at least one Core DNS pod per core pod.

   ```
   oc get pods -n ibm-spectrum-scale-dns
   ```

Status and events

The custom resource (CR) objects contain helpful information which can be retrieved by entering the **oc describe** command. For each object, a **Status** attribute provides the last observed state of the resource. In the retrieved information, a log of recent **Events** pertaining to the resource is also shown. This information can be helpful to check the desired state of the resource or when debugging with the IBM Storage Scale container native cluster. For more information, see Application Introspection and Debugging in Kubernetes documentation.

The **oc describe <CR> -n ibm-spectrum-scale** command is used to view the *status* and *events* of the custom resources, such as **cluster**, **daemon**, **filesystem**, **remotecluster**, **callhome** and others.

The *Status* can be seen in the **Conditions** section:

```
$ oc describe callhome.scale -n ibm-spectrum-scale
...
Status:
  Conditions:
    Last Transition Time:  2021-08-31T12:54:05Z
    Message:               Callhome is enabled.
    Reason:                Enabled
    Status:                True
    Type:                  Enabled
    Last Transition Time:  2021-08-31T12:54:07Z
    Message:               Successfully tested connection to the IBM Callhome Server.
    Reason:                TestPassed
```

```
    Status:                True
    Type:                  Success
  Mode:                    test
...
```

A *Condition* has the following fields:

- *Type:* Type of condition.
- *Status:* Status of the condition, one of `True`, `False` or `Unknown`.
- *Reason:* The reason contains a programmatic identifier indicating the reason for the condition's last transition.
- *Message:* Message is a human readable message indicating details about the transition.
- *Last Transition Time:* This is the last time the condition transitioned from one status to another (For example, from `False` to `True`).

The `Events` section of `oc describe` output lists the *Events*:

```
$ oc describe callhome.scale -n ibm-spectrum-scale
...
Events:
  Type    Reason       Age   From       Message
  ----    ------       ----  ----       -------
  Normal  NodeUpdate   44m   Callhome   Callhome was enabled on 0 nodes before, but now it's enabled on all 5
nodes.
  Normal  Configured   44m   Callhome   Successfully updated callhome configuration. Customer=IBM,
CustomerID=123456, Email=sroth@de.ibm.de, Country=DE, Type=test
  Normal  Enabled      44m   Callhome   Callhome has been enabled.
```

Enter the `oc get crd | grep ibm` command to see a full list of CRs that can be checked for status and events with the `oc describe` command.

- The *Events* disappear after they are created.
- The *Status* and *Events* shown in the documentation are examples and may look slightly different on your system.

Using IBM Storage Scale GUI

The following section describes how to use functionality in the IBM Storage Scale container native cluster.

- IBM Storage Scale container native GUI

IBM Storage Scale container native GUI

You can manage and monitor cluster and node information through the IBM Storage Scale container native GUI.

Red Hat OpenShift users

Red Hat OpenShift roles are mapped to two IBM Storage Scale GUI user groups. Details are provided in the following table:

Table 1. Roles and privileges

	Roles			Privileges	
OpenShift Container Platform role	GUI role	View	Download snap[1]	Manage events[2]	Test connection for call home
cluster-admin	Maintenance	Yes	Yes	Yes	Yes
ibm-spectrum-scale-maintenance	Maintenance	Yes	Yes	Yes	Yes
ibm-spectrum-scale-monitor	Monitor	Yes	No	No	No

Add a role to a user or a group

Users who are created on the Red Hat OpenShift Container Platform having a role of `ibm-spectrum-scale-maintenance` or `ibm-spectrum-scale-monitor` can log in to the IBM Storage Scale container native GUI through Single Sign On using the OAuth implementation.

```
  oc adm policy add-cluster-role-to-user <role> <user>
```

```
  oc adm policy add-cluster-role-to-group <role> <group>
```

Accessing the IBM Storage Scale GUI

To access the IBM Storage Scale GUI, complete the following steps:

1. In a browser, go to **`https://ibm-spectrum-scale-gui-ibm-spectrum-scale.apps.<ocp-domain>/`**, where **`<ocp-domain>`** is the domain of your Red Hat OpenShift cluster. There, you should see the login page for IBM Storage Scale GUI.

   If your Red Hat OpenShift domain is: **`ocp4.example.com`**, the URL would be **`https://ibm-spectrum-scale-gui-ibm-spectrum-scale.apps.ocp4.example.com`**.

2. Click **`Sign in`** and you are redirected to the Red Hat OpenShift login page.

3. Authenticate by using your OpenShift Container Platform user credentials. On success, you are redirected to the IBM Storage Scale GUI.

---

1. Ability to download master and non-master snaps.↩

2. Ability to mark events as resolved, hiding resolved tips and notifications.↩

Upgrading

When upgrading an IBM Storage Scale container native cluster that is less than v5.1.5.0, first upgrade to v5.1.5.0. For more information, see Supported upgrade paths.

IBM Storage Scale container native upgrade consists of the following topics:

- Supported upgrade paths
- Upgrading IBM Storage Scale container native
- Post upgrade tasks

Supported upgrade paths

Use the following table to understand the supported upgrade paths for IBM Storage Scale container native.

Upgrade from	To: 5.1.5.x	To: 5.1.6.x	To: 5.1.7.x	To: 5.1.9.x	To: 5.2.0.x
5.1.5.x	Yes	Yes	Yes	No	No
5.1.6.x	-	Yes	Yes	No	No
5.1.7.x	-	-	Yes	Yes	No
5.1.9.x	-	-	-	Yes	Yes
5.2.0.x	-	-	-	-	Yes

When upgrading an IBM Storage Scale container native cluster that is less than v5.1.5.0, first upgrade to v5.1.5.0.

Upgrading IBM Storage Scale container native

The following section describes how to upgrade the IBM Storage Scale container native cluster.

While an upgrade is in progress, do not perform the following:

- Do not make changes to the Cluster custom resource.
- Do not attempt to add a node to the cluster.

During an upgrade, the IBM Storage Scale operator orchestrates the upgrade procedure in a rolling node-by-node fashion. Each node will be:

- Cordoned (tainted unschedulable)
- Drained (pods are safely evicted and rescheduled to other available nodes)
- Rebooted, if necessary
- Uncordoned (returning it to normal service)

After the node is schedulable, IBM Storage Scale and IBM Storage Scale Container Storage Interface (CSI) pods will start. Applications may fail to attach storage until the system is started.

Prerequisites

- All the core pods need to be in running status.

  Use the following command to check the status of the core pods:

```
oc get daemons ibm-spectrum-scale -n ibm-spectrum-scale -ojson | jq -r '.status.podsStatus'
```

Ensure that there are no pods in any of the following states:

- starting
- terminating
- unknown
- waitingForDelete

In the following example, the output shows 1 pod in "waitingForDelete", so the upgrade should **not** be done at this time.

```
$ oc get daemons ibm-spectrum-scale -n ibm-spectrum-scale -ojson | jq -r '.status.podsStatus'
{
"running": "4",
"starting": "0",
"terminating": "0",
"unknown": "0",
"waitingForDelete": "1"
}
```

Upgrade steps

Complete the following steps to upgrade:

1. Stop the running operator pod by setting the **replicas** in the deployment to 0.

   ```
   oc scale deployment ibm-spectrum-scale-controller-manager -n ibm-spectrum-scale-operator --replicas=0
   ```

2. Delete the old security context constraint.

   ```
   oc delete scc ibm-spectrum-scale-privileged
   ```

3. Delete the old role binding for privilege.

   ```
   oc delete rolebinding -n ibm-spectrum-scale ibm-spectrum-scale-privileged --ignore-not-found
   ```

4. Delete the **MutatingWebhookConfiguration** and **ValidatingWebhookConfiguration**. These will be created in later steps.

   ```
   oc delete MutatingWebhookConfiguration ibm-spectrum-scale-mutating-webhook-configuration
   oc delete ValidatingWebhookConfiguration ibm-spectrum-scale-validating-webhook-configuration
   ```

5. Apply the new manifests.

   ```
   oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
   native/v5.2.0.x/generated/scale/install.yaml
   ```

Verification

After the new IBM Storage Scale container native operator is deployed, the upgrade process will begin. It will take some time to complete as the new container images are rolled out into the cluster.

Validate the **app.kubernetes.io/version** on the operator deployment:

```
oc get deployment ibm-spectrum-scale-controller-manager \
-n ibm-spectrum-scale-operator  -ojson | jq -r .metadata.labels
```

To check the progress of the pod restarts and node reboots, query the daemon CR using the following command:

```
oc describe daemon ibm-spectrum-scale  -n ibm-spectrum-scale
```

Information will be available under the "Status Details" and "Events" sections.

Code version updated

The version details will be listed under **.status.versions** in the Daemon CR and will be updated as the pods roll. The following command will show the versions that core pods currently have on them. Wait until all the pods are reporting the same new version.

```
oc get daemon ibm-spectrum-scale -n ibm-spectrum-scale -ojson  | jq -r .status.versions
```

Post upgrade tasks

The following section describes actions that should be performed on the cluster to complete the upgrade of IBM Storage Scale container native to the new code levels.

Apply the patch to the cluster CR to enable network policies

Issue the following command to enable network policies to your IBM Storage Scale deployment:

```
oc patch cluster.scale.spectrum.ibm.com ibm-spectrum-scale --type='json' \
-p='[{"op": "add", "path": "/spec/networkPolicy", "value":{}}]'
```

For more information, see Network policies.

Approve the new release level

It is recommended to first use the cluster with the new code of IBM Storage Scale installed, until you are sure to permanently upgrade the cluster to the new level. When you are ready to enable the new functionality of the installed release and lock in the new level, you need to approve an **UpgradeApproval** resource. An **UpgradeApproval** resource is automatically created by the operator if a release level change is detected after the upgrade.

For more information, see File system format changes between versions of IBM Storage Scale in the IBM Storage Scale documentation.

Complete the following steps:

1. Check to see if any cluster upgrade approvals are present that require action.

   An upgrade approval that shows nothing under the **COMPLETED** field are ones that require some action.

   ```
   oc get upgradeapproval -n ibm-spectrum-scale
   ```

   If an upgrade approval does not appear, check the Daemon CR status to ensure that all pods are on the new version using: **oc get daemon -n ibm-spectrum-scale -ojson | jq -r .items[].status.versions**

2. Check the **minReleaseLevel** of the cluster:

   ```
   oc exec $(oc get pods -lapp.kubernetes.io/name=core -ojsonpath="{.items[0].metadata.name}" \
   -n ibm-spectrum-scale) -c gpfs -n ibm-spectrum-scale -- mmlsconfig release
   ```

3. To approve the upgrade approval job, execute the following command:

   ```
   oc patch upgradeapproval <upgradeapproval-name> -n ibm-spectrum-scale --type='json' \
   -p='[{"op": "replace", "path": "/spec/approved", "value":true}]'
   ```

Full Example:

```
# Check for any upgrade approvals for TYPE=cluster
$ oc get upgradeapproval -n ibm-spectrum-scale
NAME            TYPE      FILESYSTEM   LAST SCHEDULE TIME   LAST SUCCESSFUL TIME   RUNNING   COMPLETED
upgrade-rmlp4   cluster

# check the daemon status for the versions deployed on each core pod
$ oc get daemon -n ibm-spectrum-scale -ojson | jq  .items[].status.versions
[
  {
    "count": "3",
    "version": "5.1.9.1"
  }
]

# check the current cluster release version
$ oc exec $(oc get pods -lapp.kubernetes.io/name=core -ojsonpath="{.items[0].metadata.name}" -n ibm-
spectrum-scale) -c gpfs -n ibm-spectrum-scale -- mmlsconfig release
minReleaseLevel 5.1.5.0

# patch the upgrade approval
$ oc patch upgradeapproval upgrade-rmlp4  -n ibm-spectrum-scale \
> --type='json'  -p='[{"op": "replace", "path": "/spec/approved", "value":true}]'
upgradeapproval.scale.spectrum.ibm.com/upgrade-rmlp4 patched

# query the upgrade approval to see it running
$ oc get upgradeapproval -n ibm-spectrum-scale
NAME            TYPE      FILESYSTEM   LAST SCHEDULE TIME   LAST SUCCESSFUL TIME   RUNNING
COMPLETED
upgrade-rmlp4   cluster               23s                                         ibm-spectrum-
scale/worker2/gpfs/upgradeCluster_p6Jhz9

# upgrade approval job completed
$ oc get upgradeapproval -n ibm-spectrum-scale
NAME            TYPE      FILESYSTEM   LAST SCHEDULE TIME   LAST SUCCESSFUL TIME   RUNNING   COMPLETED
upgrade-rmlp4   cluster               38s                  6s                                  Successful

# verify that the cluster release level has been updated
```

```
$ oc exec $(oc get pods -lapp.kubernetes.io/name=core -ojsonpath="{.items[0].metadata.name}" -n ibm-spectrum-scale) -c gpfs -n ibm-spectrum-scale -- mmlsconfig release
minReleaseLevel 5.1.9.1
```

For more information, see Completing the upgrade to a new level of IBM Storage Scale in IBM Storage Scale documentation.

Remote storage cluster considerations

The storage cluster is supported to be down-level from the IBM Storage Scale container native cluster, but it is strongly recommended that the versions match. CSI functionality is highly dependent upon the IBM Storage Scale release, filesystem level, and version, installed on the storage cluster. If the storage cluster is running an earlier version, some functionality may not be available. For more information about CSI features and required levels, see Hardware and software requirements in IBM Storage Scale CSI documentation. For more information about compatibility and software matrix, see Section 17.3 in IBM Storage Scale FAQ documentation.

- Run the following step for each filesystem to upgrade to the latest metadata format:

  If the storage cluster is being mounted by other GPFS client clusters that are running a prior version of code, performing the followin step will make those client cluster unable to mount filesystems from this storage cluster.

  ```
  mmchfs <Filesystem> -V full
  ```

  This step is optional but recommended for enabling the functionality provided at the latest levels of code.

- Enable **auto-inode-limit** of the file system.

  ```
  mmchfs <Filesystem> --auto-inode-limit
  ```

  The **--auto-inode-limit** option is available only at filesystem format level of 28.00 or later. Enable this option as soon as the filesystem is updated to 28.00 or later.

Remove deprecated resources

Some resources are deprecated in version 5.2.0, and they should be deleted.

- Run the following steps to delete the deprecated resources:

  ```bash oc delete ClusterRoleBinding ibm-spectrum-scale-dns oc delete ClusterRole ibm-spectrum-scale-dns
  ```

Configuring IBM Storage Scale Container Storage Interface (CSI) driver

Use the following sections to help with deploying IBM Storage Scale CSI with IBM Storage Scale container native:

- Configuring storage class to use CSI driver
- Managed CSI fields

Configuring storage class to use CSI driver

Storage class is used for creating lightweight volumes and fileset based volumes.

Lightweight (directory) based volumes

A Storage class example for creating directory (lightweight) based volumes is provided.

Adjust the parameters according to your environment.

```
# cat storageClass_Lightweight.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
    name: ibm-spectrum-scale-csi-lt
provisioner: spectrumscale.csi.ibm.com
parameters:
    volBackendFs: "fs1"
    volDirBasePath: "pvfileset/lwdir" # relative path from filesystem mount point for creating lightweight volume
reclaimPolicy: Delete
```

Enter the command to create the lightweight storage class:

```
oc create -f storageClass_Lightweight.yaml
```

Fileset based volumes

A Storage class example for creating fileset-based volumes is provided.

Adjust the parameters according to your environment.

```
# cat storageClass_fileset.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
    name: ibm-spectrum-scale-csi-fileset
provisioner: spectrumscale.csi.ibm.com
parameters:
    volBackendFs: fs1
reclaimPolicy: Delete
```

A sample fileset-based Storage Class is created that uses the primary file system as the **volBackendFs**. It can be used to create other storage classes with a remote file system. Enter the **oc get storageclass -oyaml > storageClass_fileset.yaml** command to create a copy of this storage class. Then configure parameters as needed and create the configured storage class by using the following command:

```
oc create -f storageClass_fileset.yaml
```

For more information, see Storage class in IBM Storage Scale CSI documentation.

Managed CSI fields

The CSI controller creates the CSI Custom Resource (CR), and manages some fields. If you change the managed fields, the controller overrides those fields. If needed, you can change the fields other than the managed fields.

Managed fields

The following fields are populated with default values by the CSI controller. Though any new values are accepted, any values that are manually removed are repopulated upon the next controller reconciliation cycle.

Table 1. Managed fields description

Field	Default Value
clusters	Two entries are created by default (local and remote clusters).
clusters.id	Local Cluster ID or Cluster ID of Remote cluster.
clusters.secrets	**ibm-spectrum-scale-gui-csiadmin** for local cluster entry, and the secret that is specified in remote cluster CR for the remote cluster entry.
clusters.secureSSLMode	**false**
clusters.primary.primaryFs	The name of the first file system created (only applicable in local. cluster entry)
clusters.restApi.guiHost	The GUI route for local cluster entry and one or more GUI hosts that are specified in the remote cluster CR for the remote cluster entry.
tolerations	**NoSchedule**, **NoExecute**, and **CriticalAddonsOnly**
attacherNodeSelector	**scale=true**
provisionerNodeSelector	**scale=true**
pluginNodeSelector	**scale=true**
snapshotterNodeSelector	**scale=true**

Editing the CSI CR

To edit the CSI CR, enter this command and update the needed field:

```
oc edit csiscaleoperator -n ibm-spectrum-scale-csi
```

Maintenance for a deployed cluster

The maintenance of a deployed IBM Storage Scalecontainer native cluster includes certain procedures.

- Shutting down a cluster
- IBM Storage Scale container native cluster and node maintenance
- Red Hat OpenShift maintenance
- Starting the cluster after shutdown
- Adding a new node to an existing cluster
- IBM Storage Scale storage cluster

Shutting down a cluster

Before you begin the maintenance procedure, the IBM Storage Scale container native cluster must be shut down to avoid any issues.

For more information, see On the nodes running CSI sidecars in IBM Storage Scale CSI documentation.

Complete the following steps to shut down a cluster:

1. Stop the running operator pod by setting the **replicas** in the deployment to 0.

   ```
   oc scale deployment ibm-spectrum-scale-controller-manager -n ibm-spectrum-scale-operator --replicas=0
   ```

2. Enter the following command to remove the CSI label:

   ```
   oc label node --all scale-
   ```

3. Enter the following command to delete the running core pods:

   ```
   oc delete pods -lapp.kubernetes.io/name=core -n ibm-spectrum-scale
   ```

IBM Storage Scale container native cluster and node maintenance

The following section provides information on changing the IBM Storage Scale cluster configuration.

Limitations

While a configuration change is in progress, do not perform the following actions:

- Do not upgrade IBM Storage Scale container native.
- Do not attempt to add a node to the cluster.
- Do not upgrade Red Hat OpenShift.

Updating existing cluster configuration

The **Cluster** resource controls IBM Storage Scale cluster configuration.

To edit the configuration for an existing cluster:

```
oc edit cluster.scale
```

When a core pod configuration requires an update, the IBM Storage Scale operator cordons, drains, reboots (if necessary), and uncordons the node, which is performed one node at a time. When the drain is complete, the core pod is updated with the new configuration.

If the node was previously cordoned before the update, the operator does not uncordon the node.

Updating cluster configuration by upgrade

An upgrade of IBM Storage Scale container native is also considered an update of the cluster configuration, as the upgrade likely introduces changes to pod specification, for example update of container images.

For more information about how IBM Storage Scale Operator orchestrates an upgrade, see Upgrading.

Red Hat OpenShift maintenance

When a Red Hat OpenShift Administrator needs to perform maintenance on a node that involves a drain, the IBM Storage Scale operator intercepts and handles the updates safely. If the operator is not running, the interception and drain fail.

Limitations

While Red Hat OpenShift configuration or maintenance is in progress, do not perform the following actions:

- Do not update IBM Storage Scale container native configuration.
- Do not upgrade IBM Storage Scale container native.
- Do not attempt to add a node to the cluster.

In addition, it is possible to have a deadlock due to the applications that are waiting to be safely evicted from one or more nodes under maintenance. The deadlock can occur if too many nodes are under maintenance concurrently, and the application workload gets disrupted if further action is taken. In these cases, the Red Hat OpenShift Administrator should safely evict and reschedule impacted applications.

For more information about troubleshooting cluster maintenance issues, see Identifying applications preventing cluster maintenance.

Red Hat OpenShift cluster configuration update

The Machine Config Operator (MCO) manages the Red Hat OpenShift machine configuration. When configuration changes impact node operation, MCO cordons and drains nodes to perform the maintenance action. Existing pods on the node are evicted and rescheduled to another available node. The IBM Storage Scale operator intercepts requests from the Kubernetes scheduler to remove the applications that are running IBM Storage Scale storage workloads before the IBM Storage Scale core pod on the corresponding node. Therefore, the application is shutdown gracefully before storage access on the node is disrupted.

When the core pod is safely removed, the MCO update continues and reboots the node, if necessary. After the MCO update is complete, the node will be uncordoned and schedulable.

Red Hat OpenShift node administrator maintenance

When a Red Hat OpenShift Administrator needs to perform manual maintenance on a node, it should be cordoned and safely drained of pods. The cordon taints the node as unschedulable and safely evicts applications. Therefore, the application is rescheduled to other available nodes, and the IBM Storage Scale operator is notified of the node maintenance needed.

1. Drain the node that needs maintenance. The drain command also cordons the node.

   ```
   oc adm drain <node name>
   ```

2. When the drain completes (without error), maintenance can be performed on the node. For example, powering it down.
3. When maintenance is complete, uncordon the node to allow it to resume normal operation and scheduling.

   ```
   oc adm uncordon <node name>
   ```

Starting the cluster after shutdown

If the IBM Storage Scale cluster was shut down, start the cluster by using the following steps:

Ensure that the worker nodes are in Ready state before restart of the IBM Storage Scale cluster by entering the **oc get nodes** command. If any of the worker nodes are in a state other than Ready, the IBM Storage Scale cluster fails to restore.

1. Scale up the operator pods by setting the **replicas** in the deployment to 1.

   ```
   oc scale deployment ibm-spectrum-scale-controller-manager -n ibm-spectrum-scale-operator --replicas=1
   ```

   After the operator pod comes back up, the core pods are rescheduled and the default CSI label is reapplied.

Adding a node to an existing cluster

To add a node:

- Ensure the node that belongs to the existing Machine Config Pool configured to include the kernel-devel extensions. This was configured as a prerequisite to IBM Storage Scale container native installation, and defaults to Red Hat OpenShift **worker** nodes. For more information, see Red Hat OpenShift Container Platform.
- Ensure the node has a node selector that matches the existing cluster's node selector. For more information, see Node selectors.

When the node selector is recognized by the IBM Storage Scale Operator, a pod will be created on the new node and will go to running state within a few minutes.

Check the progress of the creation of the new pod by entering the following command:

```
oc get pods -n ibm-spectrum-scale
```

Ensure the new pod is ready by entering the following command:

```
oc exec -n ibm-spectrum-scale \
$(oc get pods -lapp.kubernetes.io/name=core -n ibm-spectrum-scale -ojsonpath="{.items[0].metadata.name}") \
-- mmgetstate -a
```

The output appears as shown:

```
 Node number  Node name        GPFS state
-------------------------------------------
       1      worker1          active
       2      new node         arbitrating
       3      worker0          active
```

Once the pod has finished arbitrating and enters the active state, CSI will automatically be configured for use by the pod by the IBM Storage Scale and CSI Operators. Once CSI has completed configuration, then the newly added node can now be used for running applications.

When adding nodes it may make sense to select additional quorum nodes. For more information, see Labels and annotations.

IBM Storage Scale storage cluster

The following sections describe maintenance tasks for the remote IBM Storage Scale storage cluster.

- Updating remote mount access key on storage cluster
- Updating authentication to the storage cluster

Updating remote mount access key on storage cluster

An access key is used in the IBM Storage Scale storage cluster to grant file system access to the IBM Storage Scale container native cluster. The access key can be changed by following these steps:

1. Identify the **RemoteCluster** resource that you intend to change the access key.

   ```
   oc get remotecluster -n ibm-spectrum-scale
   ```

The **NAME** of the **RemoteCluster** resource is used in the following steps as **<remote-cluster>**.

1. On the IBM Storage Scale storage cluster, generate a new access key by entering the following command:

   ```
   mmauth genkey new
   ```

   Now, the storage cluster temporarily has two access keys.

2. Begin the update of the new generated key from the IBM Storage Scale container native cluster by labeling the **RemoteCluster** resource:

   ```
   oc label remotecluster <remote-cluster> scale.spectrum.ibm.com/update= -n ibm-spectrum-scale
   ```

3. Check the success by watching status and events of the remote cluster resource:

   ```
   oc describe remotecluster <remote-cluster> -n ibm-spectrum-scale
   ```

4. Verify that all your client clusters remote mounting file systems from this storage cluster is using the new key.

5. Commit the new access key by using the following command:

   ```
   mmauth genkey commit
   ```

   When the new key is committed, the old key is removed.

For more information, see mmauth command.

Updating authentication to the storage cluster

Use the following documentation as a guide for updating authentication on the remote storage cluster. This procedure involves two parts:

1. Updating user passwords on the storage cluster
2. Updating user secrets for the storage cluster on Red Hat OpenShift

Updating user passwords on the storage cluster

To change the user passwords on the storage cluster, run the following actions on the storage cluster. For more information about users, see Creating Container Operator and CSI users.

The administrator can change the password by using the following commands:

1. To change the container native operator GUI user, enter the following command on the storage cluster:

   ```
   /usr/lpp/mmfs/gui/cli/chuser cnsa_storage_gui_user -p new_cnsa_storage_gui_password
   ```

2. To change the CSI GUI user, enter the following commands on the storage cluster:

   ```
   /usr/lpp/mmfs/gui/cli/chuser csi_storage_gui_user -p new_csi_storage_gui_password
   ```

The **chuser** command needs to be run on a node where the GUI is installed in the storage cluster.

Updating user secrets for the storage cluster on Red Hat OpenShift

When user passwords on the remote IBM Storage Scale storage cluster changes, the corresponding Kubernetes secrets with the credentials must be updated for the operator to maintain communication. For more information about these secrets, see Creating secrets for storage cluster GUI users.

1. To update the secret named **cnsa-remote-mount-storage-cluster-1** in the **ibm-spectrum-scale** namespace, enter the following command:

   ```
   oc patch secret cnsa-remote-mount-storage-cluster-1 -n ibm-spectrum-scale \
   -p="{\"data\":{\"password\": \"`echo -n 'new_cnsa_storage_gui_password' | base64`\"}}"
   ```

   Verify that the secret is updated:

   ```
   echo -n `oc get secret cnsa-remote-mount-storage-cluster-1 -n ibm-spectrum-scale \
   -ojson | jq -r .data.password` | base64 -d && echo
   ```

2. To update the secret named **csi-remote-mount-storage-cluster-1** in the **ibm-spectrum-scale-csi** namespace, enter the following command:

   ```
   oc patch secret csi-remote-mount-storage-cluster-1 -n ibm-spectrum-scale-csi \
   -p="{\"data\":{\"password\": \"`echo -n 'new_csi_storage_gui_password' | base64`\"}}"
   ```

   Verify that the secret is updated:

   ```
   echo -n `oc get secret csi-remote-mount-storage-cluster-1 -n ibm-spectrum-scale-csi \
   -ojson | jq -r .data.password` | base64 -d && echo
   ```

Cleanup

The following procedures outline the steps to cleanup various pieces of the IBM Storage Scale container native solution. To delete the cluster and uninstall the product, run through each of the following sections.

- Removing applications
- Cleanup Kubernetes resources
  - Remote file systems
  - Remote clusters
  - Local file systems
  - Cluster
- Cleanup IBM Storage Scale CSI
- Cleanup IBM Storage Scale container native
- Cleanup Red Hat OpenShift nodes
- Cleanup storage cluster
- Cleanup on AWS ROSA

Removing applications

If there are applications that are accessing the storage filesystem through IBM Storage Scale CSI, they must be stopped before continuing.

1. Stop all the applications that are accessing storage via the IBM Storage Scale CSI driver.
2. Delete all the Persistent Volume Claims (PVCs) and Persistent Volumes (PVs) provisioned by IBM Storage Scale CSI Driver.

Cleanup Kubernetes resources

The following sections describe how to clean up Kubernetes custom resources that are defined in your IBM Storage Scale container native cluster.

- Remote file systems
- RemoteCluster
- Local file systems
- Cluster

Remote file systems

After a **`Filesystem`** custom resource of a remote file system is deleted, the operator does not delete the remote mount file system configuration on the IBM Storage Scale container native cluster.

Make sure that no applications use Persistent Volumes that the file system serves.

In this example, the sample **`Filesystem`** is used:

```
kind: Filesystem
metadata:
  ...
  name: remote-sample
spec:
  remote:
    cluster: remotecluster-sample
    fs: fs1
```

Complete the following steps:

1. Enter the following command to delete the file system from Red Hat OpenShift.

   ```
   oc delete filesystem remote-sample -n ibm-spectrum-scale
   ```

2. Log in to a core pod by using the following command and remove the file system from IBM Storage Scale.

   ```
   oc rsh -n ibm-spectrum-scale worker0
   ```

   - Unmount the file system on all the container native pods.

     ```
     mmunmount remote-sample -a
     ```

   - Delete the remote file system.

     ```
     mmremotefs delete remote-sample
     ```

3. You can delete the remote cluster definition if the remote storage cluster is only configured to mount and serve the single **`remote-sample`** file system. Otherwise, the other file systems must be deleted by using the same process that is mentioned in the prior step.

   - Find the remote clusters.

     ```
     mmremotecluster show all
     ```

   - Delete the remote cluster that serves the remote file system. For example, to delete a remote cluster named **`gpfs.storage`**.

     ```
     mmremotecluster delete gpfs.storage
     ```

Remote clusters

Deleting a **`RemoteCluster`** custom resource **does not** result in the operator deleting the access permission of the IBM Storage Scale container native cluster to the file systems on the remote storage cluster. The **`RemoteCluster`** controller only handles creating the access permissions.

Before removing the remote cluster credentials, ensure that no file systems are using this credential. For information on removing file system resources, see Filesystem.

For this example, the sample **RemoteCluster** is used:

```
kind: RemoteCluster
metadata:
   name: remotecluster-sample
spec:
  ...
```

Perform the following steps:

1. Delete the **RemoteCluster** definition from OpenShift by entering the following command:

   ```
   oc delete remotecluster remotecluster-sample -n ibm-spectrum-scale
   ```

2. Delete the secure credentials on the storage cluster. For more information, see Cleaning up on the storage cluster page.

Local file systems

This feature is available as a technology preview. Technology preview features are not supported for use within production environments. Use with nonproduction workloads, in demo or proof of concept environments only. IBM production service level agreements (SLAs) are not supported. Technology preview features may not be functionally complete. The purpose of technology preview features is to give access to new and exciting technologies, enabling customers to test functionality and provide feedback during the development process. The existence of a technology preview feature does not mean a future release is guaranteed. Feedback is welcome and encouraged.

After a **Filesystem** custom resource of a local file system is deleted, the operator deletes the local file system on the IBM Storage Scale container native cluster. Therefore, all file system data is erased.

Make sure that no Persistent Volumes use storage of this file system.

Complete the following steps:

1. To prevent deletion of a file system by mistake, a label must be added to the **Filesystem** custom resource to confirm deletion. Enter the following command to label the file system for deletion.

   ```
   oc label filesystem local-sample -n ibm-spectrum-scale scale.spectrum.ibm.com/allowDelete=
   ```

2. Enter the following command to delete the file system from Red Hat OpenShift and erase the data.

   ```
   oc delete filesystem local-sample -n ibm-spectrum-scale
   ```

Local disks

Deleting a **LocalDisk** custom resource causes that the related disk or volume can no longer be used within a local file system.

1. The disk to delete must not be used by a file system. A **LocalDisk** custom resource indicates the file system usage by the **USED** and **FILESYSTEM** columns.

   ```
    $ oc get localdisk worker0-vdb -n ibm-spectrum-scale
    NAME          NODE                  DEVICE     READY   USED    AVAILABLE   FILESYSTEM
   FAILUREGROUP   SIZE      AGE
    worker0-vdb   worker0.example.com   /dev/vdb   True    True    True        local-sample    1
   3.5 TiB   16h
   ```

2. If the disk is not used by any file system, enter the following command to delete it.

   ```
    oc delete localdisk worker0-vdb -n ibm-spectrum-scale
   ```

Cluster

Complete the following steps to delete a cluster:

1. Record the GPFS cluster name before deleting the cluster.

   ```
    oc get daemon -n ibm-spectrum-scale -o json | jq -r '.items[].status.clusterName'
   ```

   This GPFS cluster name is required in subsequent steps when removing the authorization on the storage cluster.

2. Enter the following command to delete the IBM Storage Scale cluster.

```
oc delete cluster.scale ibm-spectrum-scale
```

Cleanup IBM Storage Scale CSI

For steps to remove CSI from your cluster, see Cleaning up IBM Storage Scale Container Storage Interface driver and Operator by using CLIs.

Cleanup IBM Storage Scale container native

Complete the following steps:

1. Change to a default namespace:

```
oc project default
```

2. Enter the following command to uninstall the operator, kubernetes objects, namespaces, and more.

```
oc delete -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.2.0.x/generated/scale/install.yaml --ignore-not-found=true
```

3. Enter the following command to clean up the performance monitoring and IBM Storage Scale CSI artifacts.

   a. Enter the following command to list the PVs with claim of **datadir-ibm-spectrum-scale-scale-pmcollector**. Two PVs are returned.

```
oc get pv -lapp.kubernetes.io/instance=ibm-spectrum-scale,app.kubernetes.io/name=pmcollector
oc delete pv -lapp.kubernetes.io/instance=ibm-spectrum-scale,app.kubernetes.io/name=pmcollector
```

   b. Enter the following command to delete the Storage Classes created by performance monitoring and IBM Storage Scale CSI artifacts:

```
oc delete sc -lapp.kubernetes.io/instance=ibm-spectrum-scale,app.kubernetes.io/name=pmcollector
oc delete sc ibm-spectrum-scale-sample
```

Cleanup Red Hat OpenShift nodes

IBM Storage Scale container native requires host path volume mounts and creates persistent directories and files on each of the Red Hat OpenShift nodes where core pods are deployed. Removal of these persistent files are needed during clean up and before reinstallation of the cluster.

Complete the following steps to remove the persistent files on the Red Hat OpenShift nodes.

1. Switch to the default project.

```
oc project default
```

2. Enter the following command to list the nodes.

```
oc get nodes -o jsonpath="{range .items[*]}{.metadata.name}{'\n'}"
```

   Use the **nodeSelector** to reduce the set of nodes to cleanup, or run cleanup against all the nodes.

3. From the list of nodes, use the **oc debug node** command to create a debug pod accessing the host and allowing for the ability to remove the kernel modules and the host path volume-mounted directories that are used by IBM Storage Scale container native:

```
oc debug node/<openshift_node> -T -- chroot /host sh -c "rm -rf /var/mmfs; rm -rf /var/adm/ras; rmmod tracedev mmfs26 mmfslinux;"
```

   Example:

```
oc debug node/worker0.example.com -T -- chroot /host sh -c "rm -rf /var/mmfs; rm -rf /var/adm/ras; rmmod tracedev mmfs26 mmfslinux;"
Starting pod/worker0examplecom-debug ...
To use host binaries, run `chroot /host`
Removing debug pod ...
```

4. Check that none of the artifacts remain by entering the following validation command:

```
oc debug node/<openshift_node> -T -- chroot /host sh -c "ls /var/mmfs; ls /var/adm/ras; rmmod tracedev
mmfs26 mmfslinux;"
```

Example:

```
oc debug node/worker0.example.com -T -- chroot /host sh -c "ls /var/mmfs; ls /var/adm/ras; rmmod
tracedev mmfs26 mmfslinux;"
Starting pod/worker0examplecom-debug ...
To use host binaries, run `chroot /host`
ls: cannot access '/var/mmfs': No such file or directory
ls: cannot access '/var/adm/ras': No such file or directory
rmmod: ERROR: Module tracedev is not currently loaded
rmmod: ERROR: Module mmfs26 is not currently loaded
rmmod: ERROR: Module mmfslinux is not currently loaded
Removing debug pod ...
error: non-zero exit code from debug container
```

5. Remove any node labels that are created by the IBM Storage Scale container native operator:

```
# clean up all 'scale.spectrum.ibm.com' labels on the nodes
oc get nodes -ojson | jq -r '.items[].metadata.labels'  | \
grep scale.spectrum.ibm.com | sort | cut -d: -f1 | \
tr -d "\""| uniq | xargs -I{} oc label nodes --all {}\-

# clean up 'scale' label on the nodes
oc label node --all scale-
```

Cleanup storage cluster

If your storage cluster is on AWS ROSA, skip this section and see Revoke filesystem access from the storage cluster.

Delete the access permission that is granted to the IBM Storage Scale client cluster for mounting the remote file system by running the following steps on the storage cluster.

1. Enter the following command to query the name of the GPFS clusters that have established authorization to this storage cluster.

   ```
   mmauth show all
   ```

2. If the name of your GPFS cluster is **ibm-spectrum-scale.clustername.example.com**, run the following command to remove the client cluster authorization.

   ```
   mmauth delete ibm-spectrum-scale.clustername.example.com
   ```

Cleanup on AWS ROSA

This section provides cleanup procedures that needs to be done on an IBM Storage Scale container native deployment on Red Hat OpenShift Service on AWS (ROSA).

Cleaning up the ROSA worker security group

Complete the following steps to clean up ROSA worker security group:

1. In your AWS Management Console, navigate to the "EC2 Dashboard" in the region where ROSA is installed and select the "Security Groups" option from the navigation pane. Locate the Security Group for ROSA worker security group id and export the following variable.

   ```
   export AWS_ROSA_WORKER_SECURITY_GROUP=<security_group_id>
   ```

2. Use the following commands to revoke the IBM Storage Scale container native ingress traffic rules.

   ```
    aws ec2 revoke-security-group-ingress --group-id ${AWS_ROSA_WORKER_SECURITY_GROUP} --protocol tcp --
   port 12345  --source-group ${AWS_ROSA_WORKER_SECURITY_GROUP}
    aws ec2 revoke-security-group-ingress --group-id ${AWS_ROSA_WORKER_SECURITY_GROUP} --protocol tcp --
   port 1191  --source-group ${AWS_ROSA_WORKER_SECURITY_GROUP}
    aws ec2 revoke-security-group-ingress --group-id ${AWS_ROSA_WORKER_SECURITY_GROUP} --protocol tcp --
   port 60000-61000  --source-group ${AWS_ROSA_WORKER_SECURITY_GROUP}
   ```

Revoke filesystem access from the storage cluster

Before revoking access from the storage cluster, ensure that you have fully cleaned up the IBM Storage Scale container native cluster and no longer have any applications utilizing the filesystems.

After IBM Storage Scale container native has been uninstalled, use the following command to revoke the filesystem access:

```
cloudkit revoke filesystem
```

The following block shows an example of what you can expect to see when revoking the filesystem:

```
# ./cloudkit revoke filesystem
I: Logging at /root/scale-cloudkit/logs/cloudkit-16-2-2023_0-16-37.log
? Cloud platform name:  AWS
================================================================================
|                              ! Note !                                        |
================================================================================|
|  Revoke cluster involves unmount of filesystem. Make sure the I/O is         |
|  stopped and data is flushed before proceeding further.                      |
================================================================================
? Select remotemount name:  scale-strg-cls-rosa-scale
? IBM Storage Scale Container Native cluster name:  ibm-spectrum-scale.stg.rosa-scale.example.com
? Storage cluster management GUI username:  administrator
? Storage cluster management GUI password:  ********
? Connectivity method to cloud:  JumpHost
? Bastion/JumpHost instance login username:  ec2-user
? Bastion/JumpHost instance public ip address:  xxx.xxx.xxx.195
? Bastion/JumpHost SSH private key file path (will be used only for configuration):  /root/bastion_pvt_key
I: Obtaining IBM Storage Scale storage cluster definition.
I: Initiating remote mount revoke configuration.
 100% |████████████████████████████████████████████████████████████████████████████████|
(10/10, 1 it/min)
I: IBM Storage Scale cluster configuration completed.
I: Updating storage security group '<sg-storage-hash>' to revoke traffic from OCP security group '<sg-ocp-
group-hash>'.
I: Access to IBM Storage Scale cluster 'scale-strg-cls' has been revoked.
```

Monitoring

The IBM Storage Scale container native cluster is monitored by sending the health status and events between its pods.

- System monitor and Kubernetes readiness probe
- Viewing and analyzing the performance data with the IBM Storage Scale bridge for Grafana

System monitor and Kubernetes readiness probe

The scale-monitor sidecar container has the following objectives:

- Runs the containermon service which is monitoring the service (GUI, pmcollector) in the same pod.
- Provides a readiness probe API (HTTPS).
- Sends the health status and events back to the core pod on the same worker node.
- Core pod is forwarding the events to GUI or mmhealth.
- Provides an API for call home data collection.
- Has several debug tools installed and can be used for problem determination.

For more information, see Container probes in Kubernetes documentation.

If the monitoring status is HEALTHY, the probe returns success 200. When the "unreadyOnFailed" option is enabled in containermon.conf (default=true), any FAILED state causes the probe to return 500. When a critical event occurred which has the "container_unready=True" flag, the probe returns 501. When the service faces an issue, for example, no service found, it returns 502.

Viewing and analyzing the performance data with the IBM Storage Scale bridge for Grafana

IBM Storage Scale has built-in performance monitoring tool that collects metrics from various GPFS components. These metrics can provide you with a status overview and trends of the key performance indicators. You can view and analyze the collected performance data with Grafana, a third-party visualization software.

For using Grafana, you need a running Grafana instance and the IBM Storage Scale performance monitoring bridge for Grafana deployed on your IBM Storage Scale container native cluster. For more information, see IBM Storage Scale bridge for grafana repository in GitHub.

The IBM Storage Scale bridge for Grafana is an open source tool, available for free usage on IBM Storage Scale devices. It translates the IBM Storage Scale metadata and performance data collected by the IBM Storage Scale performance monitoring tool to query requests acceptable by the Grafana-integrated openTSDB plugin.

The IBM Storage Scale performance monitoring bridge for Grafana can be deployed automatically through the operator. For more information, see Grafana bridge.

For more information about setting up a Grafana instance for monitoring the IBM Storage Scale container native cluster, see Setup Grafana for monitoring a IBM Storage Scale container native cluster in a k8s OCP environment in GitHub documentation.

Support

There may be occasions where you need to troubleshoot or open a support case to IBM for further assistance.

- Known issues
- Troubleshooting
- Gathering data about your cluster

Known Issues

The following sections describe issues and solutions.

- Pod Issues
  - Error: daemon and kernel extension do not match
  - MountVolume.SetUp failed for volume "ssh-keys"
  - GUI or Grafana bridge pods fails to start, no data returned from pmcollector to frontend applications
  - pmcollector pod is in pending state during OpenShift Container Platform upgrade or reboot
  - pmsensors that shows null after failure of pmcollector node
  - pid_limits set higher than podPidLimits, but not being honored
- Remote cluster Issues
  - RestError: failed to get storage cluster information. errmsg: 401 Unauthorized GET
  - Failed to establish remote cluster connection when cacert ConfigMap does not exist
  - Adding a Remote Cluster to an existing IBM Storage Scale container native cluster taking a long time to appear
- File system issues
  - Remote file systems are defined but not mounted on all nodes
  - Remote file systems unable to mount successfully
- Upgrade Issues
  - tls errors found in the operator log
- Red Hat OpenShift Node issues
  - Adding a node fails - the node appears to already belong to a GPFS cluster
- Red Hat OpenShift Service on AWS
  - Error: InvalidGroup.NotFound

Pod Issues

- Error: daemon and kernel extension do not match
- MountVolume.SetUp failed for volume "ssh-keys"
- GUI or Grafana bridge pods fails to start, no data returned from pmcollector to front end applications
- pmcollector pod is in pending state during OpenShift Container Platform upgrade or reboot
- pmsensors that shows null after failure of pmcollector node
- pid_limits set higher than podPidLimits, but not being honored

Error: daemon and kernel extension do not match

This error occurs when an unintentional upgrade of GPFS code happens and the issue presents itself as the GPFS state is "Down". The following error can be found in the GPFS logs in **/var/adm/ras** on the core pods.

```
Error: daemon and kernel extension do not match
```

To prevent this issue, follow proper upgrade procedures as the kernel module cannot be unloaded when a file system is in use.

To resolve this problem, restart the node or follow procedures to remove application workloads and use the following command to unload the kernel module. For more information, see Removing applications.

```
rmmod tracedev mmfs26 mmfslinux
```

MountVolume.SetUp failed for volume "ssh-keys"

```
Warning FailedMount 83m (x5 over 83m) kubelet, worker-0.example.ibm.com  MountVolume.SetUp failed for volume
"ssh-keys" : secret "ibm-spectrum-scale-ssh-key-secret" not found
```

Check the pod creation time and the secret creation times. It is common that the **ssh-key-secret** is created after the deployment of the core pods. This means that the pods cannot find the secret as it does not exist yet.

The message can be misleading as it does take some time for the operator to create all the resources and this error is transient and will resolved itself after the secret is present.

If the core pods are not in **Running** state and the secret is created and present for some time, deleting the core pods should also resolve the issue. This action causes the pods to be re-created and should successfully mount the secret.

GUI or Grafana bridge pods fails to start, no data returned from pmcollector to front end applications

There exists an issue where no data is returned to front end applications that are actively consuming performance metrics from IBM Storage Scale pmcollector. This also has a signature of Grafana Bridge pod failing to start. If this is experienced, apply the following workaround.

1. Check NodeNetworkConfigurationPolicy's to determine which network interfaces are configured for a node network.

   - List the NodeNetworkConfigurationPolicies

     ```
     oc get nnce
     ```

     Example:

     ```
     # oc get nnce
     NAME                                                    STATUS
     compute-0.mycluster.example.com.bond1-ru5-policy   SuccessfullyConfigured
     compute-1.mycluster.example.com.bond1-ru6-policy   SuccessfullyConfigured
     compute-2.mycluster.example.com.bond1-ru7-policy   SuccessfullyConfigured
     control-0.mycluster.example.com.bond1-ru2-policy   SuccessfullyConfigured
     control-1.mycluster.example.com.bond1-ru3-policy   SuccessfullyConfigured
     control-2.mycluster.example.com.bond1-ru4-policy   SuccessfullyConfigured
     ```

   - Describe the **NodeNetworkConfigurationPolicy** to identify the network interface being used.

     Example:

     ```
     # oc describe nnce compute-0.mycluster.example.com.bond1-ru5-policy | grep Name
     Name:         compute-0.mycluster.example.com.bond1-ru5-policy
     Namespace:
     Name:           bond1-ru5-policy
       Name:         bond1
       Name:         bond1.3201
     ```

     In this particular example, the bond interfaces are configured for the node network traffic.

2. Change the Performance Data Collection rules to limit the discovery of the Network adapters to only to the configured interfaces.

   - Stop the sensors activities on all Core nodes

     ```
     oc get pods -lapp.kubernetes.io/name=core -n ibm-spectrum-scale \
     -ojsonpath="{range .items[*]}{.metadata.name}{'\n'}" | \
     xargs -I{} oc exec {} -n ibm-spectrum-scale -c gpfs -- \
     kill $(pgrep -fx '/opt/IBM/zimon/sbin/pmsensors -C /etc/scale-pmsensors-
     configuration/ZIMonSensors.cfg -R /var/run/perfmon')
     ```

   - Review the current filter settings for the Network sensor in the Performance Data Collection rules. These are stored in the **ibm-spectrum-scale-pmsensors-config** configmap.

     ```
     oc describe cm ibm-spectrum-scale-pmsensors-config -n ibm-spectrum-scale | grep filter | grep
     netdev
     ```

     Example output:

     ```
     # oc describe cm ibm-spectrum-scale-pmsensors-config -n ibm-spectrum-scale | grep filter | grep
     netdev
     filter = "netdev_name=veth.*|docker.*|flannel.*|cali.*|cbr.*"
     ```

The `filter =` output is used for exclusion logic.

- Edit the `ibm-spectrum-scale-pmsensors-config` configmap with the following command:

  ```
  oc edit ibm-spectrum-scale-pmsensors-config -n ibm-spectrum-scale
  ```

  Replace the substring `netdev_name=veth.*|docker.*|flannel.*|cali.*|cbr.*` with `netdev_name=^((?!bond).)*`

  Bond interface is being used in this example. Replace the *bond* with the respective adapter name that is used by the customer's network interface.

- Verify that the `ibm-spectrum-scale-pmsensors-config` configmap now reflects the wanted adapter.

  ```
  oc describe cm ibm-spectrum-scale-pmsensors-config -n ibm-spectrum-scale|grep filter | grep netdev
  ```

3. Clean up the metadata keys in the pmcollector database not related to the configured node network interfaces. Remote the shell into each pmcollector pod and issue the following commands.

   ```
   oc rsh -cpmcollector ibm-spectrum-scale-pmcollector-0
   ```

   ```
   echo "delete key .*|Network|[a-f0-9]{15}|.*" | /opt/IBM/zimon/zc 0
   ```

   ```
   echo "topo -c -d 6" | /opt/IBM/zimon/zc 0| grep Network | cut -d'|' -f2-3 | sort | uniq -c | sort -n |
   tail -50
   ```

   Then, exit the container.

   Example:

   ```
   # oc rsh -cpmcollector ibm-spectrum-scale-pmcollector-0
   sh-4.4$ echo "delete key .*|Network|[a-f0-9]{15}|.*" | /opt/IBM/zimon/zc 0
   sh-4.4$ echo "topo -c -d 6" | /opt/IBM/zimon/zc 0| grep Network | cut -d'|' -f2-3 | sort | uniq -c |
   sort -n | tail -50
        96 Network|bond0
        96 Network|bond1
        96 Network|bond1.3201
        96 Network|lo
   sh-4.4$ exit

   # oc rsh -cpmcollector ibm-spectrum-scale-pmcollector-1
   sh-4.4$ echo "delete key .*|Network|[a-f0-9]{15}|.*" | /opt/IBM/zimon/zc 0
   sh-4.4$ echo "topo -c -d 6" | /opt/IBM/zimon/zc 0| grep Network | cut -d'|' -f2-3 | sort | uniq -c |
   sort -n | tail -50
        96 Network|bond0
        96 Network|bond1
        96 Network|bond1.3201
        96 Network|lo
   sh-4.4$ exit
   ```

4. Start the sensors jobs on all Core nodes

   ```
   oc get pods -lapp.kubernetes.io/name=core -n ibm-spectrum-scale \
   -ojsonpath="{range .items[*]}{.metadata.name}{'\n'}" | \
   xargs -I{} oc exec {} -n ibm-spectrum-scale -c gpfs -- \
   /opt/IBM/zimon/sbin/pmsensors -C /etc/scale-pmsensors-configuration/ZIMonSensors.cfg -R
   /var/run/perfmon
   ```

5. Delete the pmcollector and Grafana bridge pods to update the configuration changes.

   ```
   oc delete pod -lapp.kubernetes.io/instance=ibm-spectrum-scale,app.kubernetes.io/name=pmcollector
   oc delete pod -lapp.kubernetes.io/instance=ibm-spectrum-scale,app.kubernetes.io/name=grafanabridge
   ```

   After some time, the pmcollector and Grafana bridge pods are redeployed by the ibm-spectrum-scale-operator.

pmcollector pod is in pending state during OpenShift Container Platform upgrade or reboot

```
Events:
  Type     Reason            Age                     From               Message
  ----     ------            ----                    ----               -------
  Warning  FailedScheduling  65s (x202 over 4h43m)  default-scheduler  0/6 nodes are available: 1 node(s)
were unschedulable, 2 node(s) had volume node affinity conflict, 3 node(s) had taint {node-
role.kubernetes.io/master:}, that the pod didn't tolerate.
```

This issue is caused by a problem during OpenShift Container Platform Upgrade or when a worker node has not been reset to schedulable after reboot. The pmcollector remains in a `Pending` state until the pod itself and its respective Persistent Volume can be bound to a worker node.

```
# oc get nodes
NAME                  STATUS                     ROLES    AGE     VERSION
```

```
master0.example.com   Ready                      master   5d18h   v1.18.3+2fbd7c7
master1.example.com   Ready                      master   5d18h   v1.18.3+2fbd7c7
master2.example.com   Ready                      master   5d18h   v1.18.3+2fbd7c7
worker0.example.com   Ready                      worker   5d18h   v1.17.1+45f8ddb
worker1.example.com   Ready,SchedulingDisabled   worker   5d18h   v1.17.1+45f8ddb
worker2.example.com   Ready                      worker   5d18h   v1.17.1+45f8ddb
```

If the Persistent Volume has **Node Affinity** to the host that has **SchedulingDisabled**, the pmcollector pod remains in **Pending** state until the node associated with the PV becomes schedulable.

```
# oc describe pv worker1.example.com-pv
Name:              worker1.example.com-pv
Labels:            app=scale-pmcollector
Annotations:       pv.kubernetes.io/bound-by-controller: yes
Finalizers:        [kubernetes.io/pv-protection]
StorageClass:      ibm-spectrum-scale-internal
Status:            Bound
Claim:             example/datadir-ibm-spectrum-scale-pmcollector-1
Reclaim Policy:    Delete
Access Modes:      RWO
VolumeMode:        Filesystem
Capacity:          25Gi
Node Affinity:
  Required Terms:
    Term 0:        kubernetes.io/hostname in [worker1.example.com]
Message:
Source:
    Type:  LocalVolume (a persistent volume backed by local storage on a node)
    Path:  /var/mmfs/pmcollector
```

If the issue was with OpenShift Container Platform upgrade, fixing the upgrade issue should resolve the pending pod.

If the issue is due to worker node in **SchedulingDisabled** state and not due to a failed OpenShift Container Platform upgrade, re-enable scheduling for the worker with the **oc adm uncordon** command.

pmsensors that shows null after failure of pmcollector node

If a node that is running the pmcollector pod is drained, when the node is uncordoned, the pmcollector pods get new IPs assigned. This leads to the pmsensors process issue. It displays the following message:

```
Connection to scale-pmcollector-0.scale-pmcollector successfully established.
```

But an error is reported:

```
Error on socket to scale-pmcollector-0.scale-pmcollector: No route to host (113)
```

See **/var/log/zimon/ZIMonSensors.log**. This issue can also be seen on the pmcollector pod:

```
# echo "get metrics cpu_user bucket_size 5 last 10" | /opt/IBM/zimon/zc 0
1:      worker1
2:      worker2
Row Timestamp               cpu_user
1   2020-11-16 05:27:25     null
2   2020-11-16 05:27:30     null
3   2020-11-16 05:27:35     null
4   2020-11-16 05:27:40     null
5   2020-11-16 05:27:45     null
6   2020-11-16 05:27:50     null
7   2020-11-16 05:27:55     null
8   2020-11-16 05:28:00     null
9   2020-11-16 05:28:05     null
10  2020-11-16 05:28:10     null
```

If the scale-pmcollector pods get their IP addresses changed, the pmsensors process needs to be stopped and restarted manually on all scale-core pods, to get the performance metrics collection resumed.

To stop the pmsensor process, run these commands on all the ibm-spectrum-scale-core pods. The **PMSENSORPID** variable holds the results of the **oc exec** command. If this variable is empty, then no process is running, and you do not need to enter the following command to stop the process.

```
PMSENSORPID=`oc exec <ibm-spectrum-scale-core>  -n ibm-spectrum-scale -- pgrep -fx
'/opt/IBM/zimon/sbin/pmsensors -C /etc/scale-pmsensors-configuration/ZIMonSensors.cfg -R /var/run/perfmon'`
echo $PMSENSORPID
oc exec <scale-pod> -n ibm-spectrum-scale -- kill $PMSENSORPID
```

To start the service again, enter this command on all the scale pods.

```
oc exec <scale-pod> -n ibm-spectrum-scale -- /opt/IBM/zimon/sbin/pmsensors -C /etc/scale-pmsensors-
configuration/ZIMonSensors.cfg -R /var/run/perfmon
```

pid_limits set higher than podPidLimits, but not being honored

With Red Hat OpenShift Container Platform 4.11, certain CRI-O fields introduced before the support was in kubelet have been deprecated. One of those deprecated fields is **pids_limit**, that was configured in the **ContainerRuntimeConfig** CR. For more information, see CRI-O should deprecate log size max and pids limit options.

If you had applied a custom MCO configuration with a **pids_limit** value higher than 4096, the container limits are restricted by the default **podPidsLimit** value in **kubelet.conf**. This default is set to 4096 on OpenShift Container Platform 4.11, and later. In order to increase this value, do the following:

It is highly recommended that you are at IBM Storage Scale container native v5.1.5 or higher before making changes to **MachineConfig** as the IBM Storage Scale container native operator will orchestrate the updates to **MachineConfig** as an attempt to keep the IBM Storage Scale cluster operational.

1. Define the **podPidsLimit** in the **KubeletConfig** custom resource.

```
apiVersion: machineconfiguration.openshift.io/v1
kind: KubeletConfig
metadata:
  name: 01-worker-ibm-spectrum-scale-increase-pid-limit
spec:
  machineConfigPoolSelector:
    matchLabels:
      pools.operator.machineconfiguration.openshift.io/worker: ''
  kubeletConfig:
    podPidsLimit: 8192
```

2. Delete the IBM Storage Scale container native **ContainerRuntimeConfig** resource in order to set the default value for ContainerRuntime to 0, effective unlimited:

```
oc delete ContainerRuntimeConfig 01-worker-ibm-spectrum-scale-increase-pid-limit
```

Remote cluster Issues

- RestError: failed to get storage cluster information. errmsg: 401 Unauthorized GET
- Failed to establish remote cluster connection when cacert ConfigMap does not exist
- Adding a Remote Cluster to an existing IBM Storage Scale container native cluster taking a long time to appear

RestError: failed to get storage cluster information. errmsg: 401 Unauthorized GET

When describing the **RemoteCluster** CR using **oc describe remotecluster.scale** you may see Events that show errors indicating "401 Unauthorized". The IBM Storage Scale GUI REST credentials for storage clusters are stored in Kubernetes secrets. The "401 Unauthorized" error indicates that the credentials that are provided by the Kubernetes secrets do not match the GUI user credentials in the storage cluster. For more information, see Storage cluster.

The following lists a few possible root causes:

- The GUI user was never created as described in the procedure for creating operator user and group. For more information, see Creating Container Operator and CSI users.
- The GUI user password has expired in the storage cluster and must be changed.
- The GUI user password is changed in the storage cluster.
- The GUI user is deleted in the storage cluster.

Complete the following steps to solve this problem:

1. Get the name of the secret by entering **oc describe remotecluster.scale -n ibm-spectrum-scale** command and looking for **Secret Name**:

```
...
Spec:
  Contact Nodes:
    storagecluster1node1
    storagecluster1node2
  Gui:
    Cacert:                cacert-storage-cluster-1
    Csi Secret Name:       csi-remote-mount-storage-cluster-1
    Host:                  guihost.example.com
    Insecure Skip Verify:  false
    Port:                  443
```

```
    Secret Name:           cnsa-remote-mount-storage-cluster-1
 ...
```

2. Read the credentials from the Kubernetes secret for accessing the storage cluster IBM Storage Scale GUI REST API.

```
 oc get secret cnsa-remote-mount-storage-cluster-1 -n ibm-spectrum-scale -ojsonpath='{.data.username}'
| base64 --decode && echo
 oc get secret cnsa-remote-mount-storage-cluster-1 -n ibm-spectrum-scale -ojsonpath='{.data.password}'
| base64 --decode && echo
```

   In some shells, the end of the line has a highlighted `%`. This denotes that there is no new line and should not be included when updating the password.

3. If the password differs from the one that is set for the GUI user in the storage cluster, then delete and re-create the secret as configured during installation.

4. If GUI user does not exist in storage cluster, create the IBM Storage Scale GUI user in the **ContainerOperator** group by either using the GUI or by issuing the following command in the shell of the GUI node of the storage cluster:

```
/usr/lpp/mmfs/gui/cli/mkuser cnss_storage_gui_user -p cnss_storage_gui_password -g ContainerOperator
```

Failed to establish remote cluster connection when cacert ConfigMap does not exist

When describing the remote cluster objects, you may see an error: **Error: ConfigMap "cacert-storage-cluster-1" not found**.

This issue is caused by not configuring TLS verification of CA certificates for the remote storage GUI.

For more information and resolution options, see Configuring Certificate Authority (CA) certificates.

Adding a Remote Cluster to an existing IBM Storage Scale container native cluster taking a long time to appear

When adding a RemoteCluster CR after initial installation of an IBM Storage Scale container native, it can take some time for the operator to propagate this information to the CSI CR.

To resolve this, manually trigger a reconcile of the operator by deleting the operator pod and allowing it to be re-created.

```
oc delete pod -nibm-spectrum-scale-operator -lapp.kubernetes.io/name=operator
```

Once the operator reconciles, it updates the CSI CR with the new RemoteCluster CR.

File system issues

Remote file systems are defined but not mounted on all nodes

If the RemoteMount controller shows that the target storage cluster file system is established, but the remote file system is not mounted on all the nodes in the ibm-spectrum-scale-core pods, run the following command to mount the file system manually from one of the scale-core pods:

```
# Replace FILESYSTEM with the name of your filesystem
FILESYSTEM="fs1"
oc exec $(oc get pods -lapp=ibm-spectrum-scale-core -ojsonpath="{.items[0].metadata.name}") -- mmmount
$FILESYSTEM -a
```

Remote file systems unable to mount successfully

On the file system CR, if you see events, which show that the file system is unable to mount, check in the pod to see whether running **mmlsfs <filesystem>** results in **Operation not permitted** error message.

Starting with version IBM Storage Scale and IBM Storage Scale container native v5.1.3, the **tscCmdAllowRemoteConnections** configuration is recommended to be set to **no**. If the storage cluster and all client clusters (including IBM Storage Scale container native) are at versions >= v5.1.3, it is recommended to set this value to **no**. However, if any version is < v5.1.3, **tscCmdAllowRemoteConnections** needs to be set to **yes** on the storage cluster and client clusters to successfully communicate between the clusters.

Use the following table as a reference.

Table 1. tscCmdAllowRemoteConnections configuration

Storage cluster version	IBM Storage Scale container native version	`tscCmdAllowRemoteConnections`
< 5.1.3	< 5.1.3.0	yes
>= 5.1.3	< 5.1.3.0	yes
>= 5.1.3	>= 5.1.3.0	no

- To change this value on the storage cluster, run: **`mmchconfig tscCmdAllowRemoteConnections='yes|no'`**.
- To change this value on the IBM Storage Scale container native cluster, set **`tscCmdAllowRemoteConnections: yes|no`** in the clusterProfile section of the cluster spec:

```
kind: Cluster
metadata:
name: ibm-spectrum-scale
spec:
  ...
  daemon:
    ...
    ...
    clusterProfile:
      tscCmdAllowRemoteConnections: "yes"
```

For more information to configure the **`clusterProfile`** section of the cluster spec, see Cluster profile.

Upgrade Issues

- tls errors found in the operator log

tls errors found in the operator log

After upgrading the IBM Storage Scale container native, you can see many messages in the operator log similar to the following:

```
http: TLS handshake error from <IP>:<PORT> remote error: tls: bad certificate
```

Verify that the Webhooks have a **`caBundle`** injected for each defined **`path:`** You can use these commands to check this:

- To check the **`MutatingWebhookConfiguration`**:

```
oc get MutatingWebhookConfiguration ibm-spectrum-scale-mutating-webhook-configuration -oyaml | egrep
"caBundle:|path:"
```

- To check the **`ValidatingWebhookConfiguration`**:

```
oc get ValidatingWebhookConfiguration ibm-spectrum-scale-validating-webhook-configuration -oyaml |
egrep "caBundle:|path:"
```

For example, if running the command against the **`ibm-spectrum-scale-mutating-webhook-configuration`**, you can see an output similar to the following:

```
$ oc get MutatingWebhookConfiguration ibm-spectrum-scale-mutating-webhook-configuration -oyaml | egrep
"caBundle:|path:"
caBundle: <SOME CA BUNDLE DEFINED, LONG KEY>
  path: /mutate-scale-spectrum-ibm-com-v1beta1-cluster
  path: /mutate-scale-spectrum-ibm-com-v1beta1-filesystem
```

The output shows one **`caBundle:`** entry and two **`path:`** entries, meaning for the second path: **`/mutate-scale-spectrum-ibm-com-v1beta1-filesystem`**, a valid **`caBundle`** was NOT injected. This is caused by a bug in the Red Hat OpenShift service CA operator, OCPBUGS-8512.

If this has occurred in your cluster, resolve the problem by using the following steps:

1. Delete the **`MutatingWebhookConfiguration`** and **`ValidatingWebhookConfiguration`**.

```
oc delete MutatingWebhookConfiguration ibm-spectrum-scale-mutating-webhook-configuration
oc delete ValidatingWebhookConfiguration ibm-spectrum-scale-validating-webhook-configuration
```

2. Reapply the manifest for IBM Storage Scale container native. For example, if you are upgrading to v5.1.9, then run:

```
oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.2.0.x/generated/scale/install.yaml
```

3. Running the webhook queries again should result in each endpoint having a **caBundle** injected in each path:

```
caBundle: <SOME CA BUNDLE DEFINED, LONG KEY>
  path: /mutate-scale-spectrum-ibm-com-v1beta1-cluster
caBundle: <SOME CA BUNDLE DEFINED, LONG KEY>
  path: /mutate-scale-spectrum-ibm-com-v1beta1-filesystem
```

For more information, see Upgrading IBM Storage Scale container native.

Red Hat OpenShift Node issues

Adding a node fails - the node appears to already belong to a GPFS cluster

When adding a worker node into Red Hat OpenShift, and using the nodeSelector of **node-role.kubernetes.io/worker** in the Cluster CR, the IBM Storage Scale Container native operator deploys a core pod to the newly added node and attempt to add this node into the GPFS cluster. There can be a situation where the core pod is in "Init: 1/2" state with no sign of recovery.

The operator log contains entries matching **ERROR Failed to add node** and **mmaddnode** failing with the reason:

```
The node appears to already belong to a GPFS cluster.
```

To recover from this scenario, use the following steps:

1. Create a debug pod to the node where the pod is failing to start and delete the GPFS metadata.

   ```
   oc debug node/<openshift_worker_node> -T -- chroot /host sh -c "rm -rf /var/mmfs; rm -rf /var/adm/ras"
   ```

   Example:

   ```
   oc debug node/worker0.example.com -T -- chroot /host sh -c "rm -rf /var/mmfs; rm -rf /var/adm/ras"
   Starting pod/worker0examplecom-debug ...
   To use host binaries, run `chroot /host`
   Removing debug pod ...
   ```

2. Delete the core pod. If the core pod is called **worker3**, run: **oc delete pod worker3 -n ibm-spectrum-scale**.
3. The operator should reconcile and attempt to create the pod again and succeed.

Red Hat OpenShift Service on AWS

Error: InvalidGroup.NotFound

The following error is seen in the Cloudkit logs:

```
{"level":"debug","ts":"2024-02-19T03:11:56.724-
0500","caller":"commonhelpers/common_helpers.go:436","msg":"...InvalidGroup.NotFound: You have specified two
resources that belong to different networks."}
```

The jump host security group or the cloud-vm installer node security group belongs to different VPC than the VPC opted for IBM Storage Scale cluster deployment.

Cloudkit does not support VPC peering. It is required to use the cloud-vm installer node VPC for IBM Storage Scale deployment. If your installer node is in a different VPC, it needs to be migrated to the VPC where you want to deploy the IBM Storage Scale cluster.

Troubleshooting

Use the following sections to help troubleshoot and debug specific issues with IBM Storage Scale container native deployment.

Troubleshooting operator

operator pod is not successfully deployed

No operator pod exists in the **ibm-spectrum-scale-operator** namespace.

- Verify that all worker nodes in the Red Hat OpenShift Container Platform cluster are in a **Ready** state. Nodes that are **NotReady** might be preventing the operator pod from being scheduled.

  ```
  oc get nodes
  ```

  ```
  # oc get nodes
  NAME                   STATUS      ROLES    AGE   VERSION
  master0.example.com    Ready       master   65d   v1.18.3+6c42de8
  master1.example.com    Ready       master   65d   v1.18.3+6c42de8
  master2.example.com    Ready       master   65d   v1.18.3+6c42de8
  worker0.example.com    NotReady    worker   65d   v1.18.3+6c42de8
  worker1.example.com    NotReady    worker   65d   v1.18.3+6c42de8
  worker2.example.com    NotReady    worker   65d   v1.18.3+6c42de8
  ```

- Inspect the operator namespace and look for details that might point to problems.
  - Check the operator deployment: **oc get deployment -n ibm-spectrum-scale-operator**

    Describe the deployment for more details:

    ```
    oc describe deployment ibm-spectrum-scale-controller-manager -n ibm-spectrum-scale-operator
    ```

  - Check the operator replicaset: **oc get replicasets -n ibm-spectrum-scale-operator**

    Describe the replicaset for more details:

    ```
    oc describe replicaset $(oc get replicasets -n ibm-spectrum-scale-operator \
    -ojson | jq -r .items[0].metadata.name) -n ibm-spectrum-scale-operator
    ```

operator pod shows container restarts

Kubernetes keeps the logs of the current container and the previous container.

Look at the previous container logs and look for any issues that might be causing the container to restart.

```
oc logs -p $(oc get pods -n ibm-spectrum-scale-operator -ojson | \
jq -r .items[0].metadata.name) -n ibm-spectrum-scale-operator
```

Troubleshooting deployment

no endpoints available for service "ibm-spectrum-scale-webhook-service"

When creating or editing IBM Storage Scale container native custom resources, it is possible that validating or mutating webhooks fails when the operator pod is unavailable. If you receive an error: **no endpoints for service "ibm-spectrum-scale-webhook-service"**, check the state of the operator pod.

Example error:

```
# oc apply -f remotecluster.yaml
remotecluster.scale.spectrum.ibm.com/remotecluster-sample unchanged
Error from server (InternalError): error when creating "remotecluster.yaml": Internal error occurred: failed
calling webhook "mcluster.scale.spectrum.ibm.com": failed to call webhook: Post "https://ibm-spectrum-scale-
webhook-service.ibm-spectrum-scale-operator.svc:443/mutate-scale-spectrum-ibm-com-v1beta1-cluster?
timeout=10s": no endpoints available for service "ibm-spectrum-scale-webhook-service"
```

Checking the operator pod, the STATUS is **ImagePullBackOff**:

```
# oc get pods -n ibm-spectrum-scale-operator
NAME                                                  READY   STATUS             RESTARTS          AGE
pod/ibm-spectrum-scale-controller-manager-64bb4798df-rrj4j   0/1     ImagePullBackOff   10 (4m14s ago)    34m
```

In the example, the operator is unable to pull the image due to errors in the pull credentials. To remedy the **no endpoints available** issue, the operator problems must be resolved first, then retry the original commands.

core pods are stuck in init

If for some reason, the IBM Storage Scale container native cluster fails to create, the core pods on the worker nodes get stuck in the Init container.

```
# oc get pods
NAME                                  READY   STATUS     RESTARTS   AGE
...
worker0                               2/2     Init:1/2   0          2h
worker1                               2/2     Init:1/2   0          2h
worker2                               2/2     Init:1/2   0          2h
worker3                               2/2     Init:1/2   0          2h
```

There is no recovery from this problem. For more information about clean up, see Cleanup IBM Storage Scale container native and Cleanup Red Hat OpenShift nodes. For more information on redeploying, see Installing the IBM Storage Scale container native operator and cluster.

core, GUI, or collector pods are in ErrImgPull or ImagePullBackOff state

When viewing **oc get pods -n ibm-spectrum-scale**, if any of the pods are in **ErrImgPull** or **ImagePullBackOff** state, use **oc describe pod <podname>** to get more details on the pod and look for any errors that may be happening.

```
oc describe pod <pod-name> -n ibm-spectrum-scale
```

core, GUI, or collector pods are not up

- If the pods are not deployed in the **ibm-spectrum-scale** namespace, or the cluster is not created, examine operator pod logs:

  ```
  oc logs $(oc get pods -n ibm-spectrum-scale-operator -ojson  | jq -r ".items[0].metadata.name") -n
  ibm-spectrum-scale-operator
  ```

core, GUI, or collector pods show container restarts

- Kubernetes keeps the logs of the current container and the previous container. Look at the previous container's logs for any clues by using the following command:

  ```
  oc logs -p <scale pod> -n ibm-spectrum-scale
  ```

all pods are running but the GPFS cluster is stuck in the "arbitrating" state

If the cluster is stuck in an arbitrating state:

- Check the output of **mmlscluster**.

  ```
  oc exec $(oc get pods -lapp.kubernetes.io/name=core -n ibm-spectrum-scale -o json | jq -r
  ".items[0].metadata.name") -- mmlscluster
  ```

- Check the GPFS logs.

  ```
  oc logs $(oc get pods -lapp.kubernetes.io/name=core -n ibm-spectrum-scale  -o json | jq -r
  ".items[0].metadata.name") -c logs | grep mmfs.log.latest
  ```

remote mount file system not getting configured or mounted

- Check the **RemoteCluster** objects and the **Filesystem** objects. The **Filesystem** controller waits until the RemoteCluster object is Ready before attempting to configure the remote mount file system. Describe the objects and check **Status** or **Events** for any reasons for failures.
    - Remote Clusters

      ```
      oc get remotecluster.scale -n ibm-spectrum-scale
      oc describe remotecluster.scale <name> -n ibm-spectrum-scale
      ```

```
* File system

    ```bash
    oc get filesystem.scale -n ibm-spectrum-scale
    oc describe filesystem.scale <name> -n ibm-spectrum-scale
    ```


Check the `Status` and `Events` for any reasons of failure.

If nothing, check the operator logs for any errors:

```bash
oc logs $(oc get pods -n ibm-spectrum-scale-operator -ojson  | jq -r ".items[0].metadata.name") -n ibm-
spectrum-scale-operator
```
```

- Enter the **mmnetverify** command to verify the network between the clusters. For more information, see mmnetverify command in IBM Storage Scale documentation.

Troubleshooting custom resources

Troubleshooting remote cluster

- Failed to register remote storage cluster on local GUI REST-API.

Failed to register remote storage cluster on local GUI REST-API.

The remote cluster resource is not READY and describing the resource shows the status: **Failed to register remote storage cluster on local GUI REST-API. Check the operator logs for more details.**

This error might be caused from the inability to resolve the hostname of the contact nodes on the storage cluster from the core pods. Check the operator logs and look for a failure on **mmremotecluster add** command.

```
mmremotecluster: Incorrect node node1.example.com specified for command.
mmremotecluster: Command failed. Examine previous error messages to determine cause.
```

To resolve, double check that you are able to resolve the hostnames of the contact nodes by using **nslookup**. If you are unable to fix the site DNS, you can also resolve this using **hostAliases** field in the Cluster spec to add entries into the IBM Storage Scale CoreDNS host file. For more information, see **Host aliases**.

Troubleshooting IBM Storage Scale Container Storage Interface (CSI)

- CSI pods stuck in CrashLoopBackOff (Unauthorized GET request)
- CSI CR is never created

CSI pods stuck in CrashLoopBackOff (Unauthorized GET request)

The following output shows an example of the CSI pods in **CrashLoopBackOff**.

```
# oc get pods
NAME                                                  READY   STATUS             RESTARTS   AGE
ibm-spectrum-scale-csi-95661                          1/2     CrashLoopBackOff   9          26m
ibm-spectrum-scale-csi-attacher-0                     1/1     Running            0          85m
ibm-spectrum-scale-csi-klr7x                          1/2     CrashLoopBackOff   9          26m
ibm-spectrum-scale-csi-operator-56955949c4-mzn7g      1/1     Running            0          90m
ibm-spectrum-scale-csi-provisioner-0                  1/1     Running            0          85m
ibm-spectrum-scale-csi-xlxkl                          1/2     CrashLoopBackOff   9          26m
```

The logs of the CSI pods might reveal what is causing the problems.

```
# oc logs ibm-spectrum-scale-csi-9566l -c ibm-spectrum-scale-csi

...

I1218 17:27:33.875884       1 http_utils.go:60] http_utils FormatURL. url: https://ibm-spectrum-scale-gui-ibm-spectrum-scale.apps.example.com:443/
I1218 17:27:33.875894       1 rest_v2.go:586] rest_v2 doHTTP. endpoint: https://ibm-spectrum-scale-gui-ibm-spectrum-scale.apps.example.com:443/scalemgmt/v2/cluster, method: GET, param: <nil>
I1218 17:27:33.875900       1 http_utils.go:74] http_utils HttpExecuteUserAuth. type: GET, url: https://ibm-spectrum-scale-gui-ibm-spectrum-scale.apps.example.com:443/scalemgmt/v2/cluster, user: csi-cnsa-gui-user
```

- Check that the **csi-cnsa-gui-user** role was created.

  ```
  # oc exec ibm-spectrum-scale-gui-0 -n ibm-spectrum-scale -- /usr/lpp/mmfs/gui/cli/lsuser
  Defaulting container name to liberty.
  Use 'oc describe pod/ibm-spectrum-scale-gui-0 -n ibm-spectrum-scale' to see all of the containers in this pod.
  Name              Long name Password status Group names       Failed login attempts Target Feedback Date
  ContainerOperator           active          ContainerOperator 0
  EFSSG1000I The command completed successfully.
  ```

  In this case, the **csi-cnsa-gui-user** role was not created. To resolve the issue, enter the following command to create the GUI user:

  ```
  # oc exec -c liberty ibm-spectrum-scale-gui-0 -n ibm-spectrum-scale -- /usr/lpp/mmfs/gui/cli/mkuser csi-cnsa-gui-user -p csi-cnsa-gui-password -g CsiAdmin
  EFSSG0019I The user csi-cnsa-gui-user has been successfully created.
  EFSSG1000I The command completed successfully.
  ```

- Check that the **csi-remote-mount-storage-cluster-1** secret was created with correct credentials.

  ```
  # oc get secrets csi-remote-mount-storage-cluster-1 -n ibm-spectrum-scale-csi -ojsonpath='{.data.username}' | base64 --decode
  csi-cnsa-gui-user

  # oc get secrets csi-remote-mount-storage-cluster-1  -n ibm-spectrum-scale-csi -ojsonpath='{.data.password}' | base64 --decode
  this-is-a-bad-password
  ```

  In this case, the **csi-remote-mount-storage-cluster-1** secret was created without the correct password. To resolve the issue, enter the following command to delete the secret and re-create it with correct values:

  ```
  # oc delete secrets csi-remote-mount-storage-cluster-1  -n ibm-spectrum-scale-csi
  secret "csi-remote-mount-storage-cluster-1" deleted

  # oc create secret generic csi-remote-mount-storage-cluster-1 --from-literal=username=csi-cnsa-gui-user --from-literal=password=csi-cnsa-gui-password -n ibm-spectrum-scale-csi
  secret/csi-remote-mount-storage-cluster-1 created

  # oc label secret csi-remote-mount-storage-cluster-1 product=ibm-spectrum-scale-csi -n ibm-spectrum-scale-csi
  secret/csi-remote-mount-storage-cluster-1 labeled
  ```

CSI CR is never created

When all the core pods are running and the IBM Storage Scale container native cluster appears to be in a good state, the CSI CR is created automatically. In some error paths this does not happen and causes the driver pods to not be scheduled.

```
# oc get po,csiscaleoperator -n ibm-spectrum-scale-csi
NAME                                                  READY   STATUS    RESTARTS   AGE
pod/ibm-spectrum-scale-csi-operator-79bd756d58-ht6hf   1/1     Running   0          47h
```

Only the operator pod is listed and no results are found for csiscaleoperators

- Check that all the GUI pods are up and running.

  ```
  # oc get pods -n ibm-spectrum-scale
  NAME                                 READY   STATUS    RESTARTS   AGE
  ibm-spectrum-scale-gui-0             4/4     Running   0          3m58s
  ibm-spectrum-scale-gui-1             4/4     Running   0          95s
  ibm-spectrum-scale-pmcollector-0     2/2     Running   0          3m59s
  worker0                              2/2     Running   0          3m59s
  worker1                              2/2     Running   0          3m58s
  worker2                              2/2     Running   0          3m58s
  ```

  All GUI pods must be up and running before the CSI CR is created. Each pod can take a few minutes for all containers in the pod to enter the **Running** state.

- Check that the daemon status has a nonempty cluster ID.

  ```
  oc describe daemon -n ibm-spectrum-scale
  ```

  Find the status section and validate that the **Cluster ID** field exists and is not empty.

  ```
  Status:
    Cluster ID:    3004252500454687654
    Cluster Name:  example.cluster.com
  ```

  If those fields are missing, then the IBM Storage Scale container native cluster is experiencing an issue. Check the operator logs for more information.

Troubleshooting PVCs

- PVC creation issue: PVC binding in pending state

PVC creation issue: PVC binding in pending state

If there are problems when PVCs are created:

- Check whether multiple GUIs are installed at the remote storage cluster.
- SSH to one of the GUI nodes and run the following command:

  ```
  mmlsfileset <filesystem name>
  ```

- Check whether file set is unlinked.

If multiple GUIs are installed, then contact IBM support for further guidance.

Troubleshooting Red Hat OpenShift

- GUI mounts not getting refreshed
- Recovery of IBM Storage Scale container native cluster after Red Hat OpenShift node failure

GUI mounts not getting refreshed

During upgrading Red Hat OpenShift, a problem can occur where the GUI mount is not being refreshed when multiple Red Hat OpenShift clusters are remote mounting the same file system.

To resolve this issue, unmount the file system from the other Red Hat OpenShift clusters.

```
# /usr/lpp/mmfs/gui/cli/runtask FILESYSTEM_MOUNT
err: Batch entry 3 INSERT INTO FSCC.FILESYSTEM_MOUNTS (CLUSTER_ID, DEVICENAME, HOST_NAME, MOUNT_MODE,
LAST_UPDATE) VALUES ('5228226002706731921','fs1','worker1.example.com','RW','2021-07-28
19:06:15.111000+00'::timestamp) was aborted: ERROR: duplicate key value violates unique constraint
"filesystem_mounts_pk"
  Detail: Key (host_name, cluster_id, devicename)=(worker1.example.com, 5228226002706731921 , fs1) already
exists.  Call getNextException to see other errors in the batch.
EFSSG1150C Running specified task was unsuccessful.
# /usr/lpp/mmfs/gui/cli/runtask FILESYSTEM_MOUNT
EFSSG1000I The command completed successfully.
# exit
exit
```

Recovery of cluster after Red Hat OpenShift node failure

IBM Storage Scale container native operator does not have support for the removal of a node. There have been cases where a node fails to recover after the node suffers an outage, for example, after a Red Hat OpenShift update.

- Before you begin
- Collect configuration before servicing the OpenShift Container Platform cluster
- Delete one Scale node from IBM Storage Scale container native cluster

- Add new Red Hat OpenShift node for use in existing IBM Storage Scale container native cluster
- Validate the IBM Storage Scale container native cluster after removal and replacement of Red Hat OpenShift node

Before you begin

Before the maintenance of the failed node, ensure that the node selectors, quorum and manager designations, and network configuration annotations are fully understood. These configurations are applied in subsequent steps after removal of the node to ensure parity of the cluster.

- Log the node selector labels (required for the Scale core pods) that are being used by the Cluster CR.
- Understand if the node to be removed is a quorum node for the IBM Storage Scale container native cluster.
- Identify if a CNI network configuration is in use for the IBM Storage Scale container native cluster.

Collect configuration before servicing the OpenShift Container Platform cluster

Run an IBM Storage Scale container native must-gather to collect the state before servicing the Red Hat OpenShift cluster. For more information, see must-gather

Delete one Scale node from IBM Storage Scale container native cluster

To delete the Scale node from IBM Storage Scale container native cluster, complete the following steps:

1. Stop the running operator pod by setting the **replicas** in the deployment to 0.

   ```
   oc scale deployment ibm-spectrum-scale-controller-manager -n ibm-spectrum-scale-operator --replicas=0
   ```

2. If the failed Red Hat OpenShift node exists, delete the corresponding Scale core pod.

   ```
   oc delete pod REPLACE_WITH_CORE_POD_OF_FAILED_OCP_NODE -n ibm-spectrum-scale
   ```

3. Enter a currently running Scale core pod to run the remaining commands.

   ```
   oc -n ibm-spectrum-scale rsh REPLACE_WITH_RUNNING_CORE_POD
   ```

4. Note the quorum and manager designations as denoted in **mmlscluster**. This information is needed when we attempt to add back a new node.

   ```
   mmlscluster
   ```

5. Map the affected Red Hat OpenShift node to the GPFS admin interface. It should report a status of **Unknown**.

   The GPFS admin interface would have the shortname of the affected Red Hat OpenShift node.

   ```
   mmgetstate -a
   ```

6. Ensure that most Scale nodes are in the **Active** state. Then, force delete the desired GPFS admin interface from the Scale cluster.

   Exercise caution when shutting down GPFS on quorum nodes or deleting quorum nodes from the GPFS cluster. If the number of remaining quorum nodes falls under the requirement for a quorum, then you are unable to perform file system operations.

   ```
   mmdelnode -N REPLACE_WITH_FAILED_GPFS_ADMIN_NODE --force
   ```

7. Validate that the bad GPFS admin node is deleted from the Scale cluster.

   ```
   mmgetstate -a
   ```

8. Exit from the Scale core pod.

   ```
   exit
   ```

Add new Red Hat OpenShift node for use in existing IBM Storage Scale container native cluster

The previous node configurations of the failed node need to be applied to the new Red Hat OpenShift node. So that the IBM Storage Scale container native operator can successfully add it to the existing IBM Storage Scale container native cluster.

1. Ensure the quorum and manager designations are matching what the previous node had in place. Ensure that the GPFS cluster is not in minority of quorum. If the number of remaining quorum nodes falls under the requirement for a quorum, then you are unable to perform file system operations.

   Alternatively, if a node is removed but no replacement node is added, ensure that the appropriate number of nodes are assigned to satisfy the quorum.

2. If using CNI, ensure that the network configuration is valid according to CNI documentation. Ensure that the **scale.spectrum.ibm.com/daemon-network** annotation is present on the new Red Hat OpenShift node. For more information, see Container network interface (CNI) configuration.
3. Ensure the node selector labels (required for the Scale core pods) from the Cluster CR are present on the newly added Red Hat OpenShift node.
4. Scale up the operator pods by setting the **replicas** in the deployment to 1.

   ```
   oc scale deployment ibm-spectrum-scale-controller-manager -n ibm-spectrum-scale-operator --replicas=1
   ```

5. The new Scale core pod should be created and go to running in a short amount of time.

Validate IBM Storage Scale container native cluster after removal and replacement of Red Hat OpenShift node

After the removal and replacement of the failed Red Hat OpenShift node has been completed, perform the validation steps. For more information, see Validating installation. Ensure the following:

- All pods are in a running state. Make sure that the CSI pod on the replaced node is running. This validates that file systems are active and running.
- Validate that the IBM Storage Scale cluster has successfully added the node with the desired node designation.
- Validate that the nodes are active in the IBM Storage Scale cluster.

Troubleshooting cluster maintenance

Identifying applications that prevent cluster maintenance

Debug steps to determine when a cluster maintenance action is not being completed.

When do you see this problem?

A drain occurs when a core pod is selected for deletion. Many actions prompt a core pod deletion:

- Pod evictions
  - Red Hat OpenShift Machine Config Operator
  - User-initiated drains
- Pod spec updates
  - Resource requests, for example to change core pod requests for CPU or memory
  - Image updates prompted by a new release

What does this look like?

An update that is driven by pod spec updates does present as core pods that wait for deletion. An update that is driven by Red Hat OpenShift Machine Config Operator (MCO) ceases to update nodes. The signature does look similar between the two scenarios. Use the following steps to determine where to direct the support case.

When to open a support case?

To determine whether the MCO is stuck due to IBM Storage Scale container native, run the following command:

```
oc describe daemon -n ibm-spectrum-scale
```

Example output:

```
...
Status:
...
    Cordon And Drain:
        Nodes Cordoned:  worker0.example.ibm.com
        Nodes Draining:
            Node:  worker0.example.ibm.com
        Ongoing Pod Evictions:
        Pod Evictions Failed:
            myworkload/mypod-0
            myworkload/mypod-1
```

```
        Pod Eviction Requests:
            Pods: worker0, worker1, worker2
            Requestor: machine-config-controller
    ...
    Status Details:
        ...
        Pods Waiting To Be Deleted:
            Delete Reason:   Eviction of 3 pods requested by machine-config-controller
            Pods:            worker0, worker1, worker2
    ...
Events:
    Type     Reason                 Age    From    Message
    ----     ------                 ----   ----    -------
    Normal   PodEvictionRequested   7m9s   Daemon  Eviction of 3 pods requested by machine-config-
controller.
    Normal   NodeDraining           6m28s  Daemon  Node worker2.example.ibm.com drained by Storage Scale
operator. Evicting 6 pods.
    Warning  NodeDrainFailed        87s    Daemon  Failed to evict pod myworkload/mypod-0 on node
worker0.example.ibm.com. Reason: Cannot evict pod as it would violate the pod's disruption budget.
    Warning  NodeDrainFailed        87s    Daemon  Failed to evict pod myworkload/mypod-1 on node
worker0.example.ibm.com. Reason: Cannot evict pod as it would violate the pod's disruption budget.
```

If the **Pod Evictions Failed:** list is empty and no **NodeDrainFailed** events are present, then IBM Storage Scale container native is blocking the ongoing drain. To resolve it, raise a support ticket to IBM. For more information, see Gather data about the IBM Storage Scale container native cluster.

If pods are listed in **Pod Evictions Failed:**, check the message that is provided in the **NodeDrainFailed** events. The pods that are listed in **Pod Evictions Failed:** block the node drain. Try to solve the problem with the pods that are blocking the node drain. If you cannot solve this problem, raise a support ticket to Red Hat. For more information, see Gather data about the Red Hat OpenShift Container Platform cluster.

Identifying signatures of an ongoing update

Complete the following steps:

1. Check the Daemon status to verify whether any pods are awaiting deletion.

   ```
   oc describe daemon -n ibm-spectrum-scale
   ```

   Example output:

   ```
   Status Details:
       ...
       Pods Waiting To Be Deleted:
           Delete Reason:   Eviction of 3 pods requested by machine-config-controller.
           Pods:            worker0, worker1, worker2
           Delete Reason:   Pod worker1 will restart because image of containers config, gpfs, mmbuildgpl
   are updated. The node will be rebooted.
           Pods:            worker1
   ```

2. Check whether any nodes are cordoned.

   ```
   oc get nodes
   ```

   Example:

   ```
   NAME                       STATUS                     ROLES    AGE   VERSION
   master0.example.ibm.com    Ready                      master   23d   v1.24.0+3882f8f
   master1.example.ibm.com    Ready                      master   23d   v1.24.0+3882f8f
   master2.example.ibm.com    Ready                      master   23d   v1.24.0+3882f8f
   worker0.example.ibm.com    Ready,SchedulingDisabled   worker   23d   v1.24.0+3882f8f
   worker1.example.ibm.com    Ready                      worker   23d   v1.24.0+3882f8f
   worker2.example.ibm.com    Ready                      worker   23d   v1.24.0+3882f8f
   ```

   ```
   oc describe daemon -n ibm-spectrum-scale
   ```

   Example:

   ```
   Cordon And Drain:
       Nodes Cordoned:  worker0.example.ibm.com
   ```

3. Check events in the daemon.

   ```
   oc describe daemon -n ibm-spectrum-scale
   ```

   Example event:

   ```
   Warning  NodeDrainFailed  87s    Daemon  Failed to evict pod myworkload/mypod-0 on node
   worker0.example.ibm.com. Reason: Cannot evict pod as it would violate the pod's disruption budget.
   ```

Gather data about your cluster

When opening issue to IBM Support, you may be asked to provide information that will help assist in performing root cause analysis. The follow sections describe the process for gathering data on your cluster.

- must-gather
- Generating GPFS trace reports
- Kernel crash dumps

must-gather

The **`oc adm must-gather`** CLI command is used to collect information about the cluster.

Prerequisites

- Access to the cluster as a user with the **`cluster-admin`** role
- The OpenShift Container Platform CLI (oc) command installed

Gather data about the Red Hat OpenShift Container Platform cluster

For issues with the Red Hat OpenShift Container Platform cluster where a ticket must be opened with Red Hat Support, provide the debugging information about the cluster for problem determination. For more information, see Gathering data about your cluster in Red Hat OpenShift documentation.

Executing a default **`must-gather`** for OpenShift Container Platform debug does **not** collect information for IBM Storage Scale container native.

Gather data about the IBM Storage Scale container native cluster

To gather data specifically for the IBM Storage Scale container native cluster to provide to IBM Support, point the **`must-gather`** tool to the **`ibm-spectrum-scale-must-gather`** image. This will collect Kubernetes resources associated with the IBM Storage Scale container native cluster as well as retrieving a GPFS snap.

`oc adm must-gather --image` requires the image stored in a repository where it can be anonymously pulled (no credentials required). In an airgapped environment, the **`ibm-spectrum-scale-must-gather`** image must be pulled from the IBM Cloud Container Registry and then uploaded to an internal image registry allowing anonymous pull. For more information about air gap instructions, see Air-gapped installs.

1. In the directory where your must-gather contents are to be stored, enter the must-gather command using the **`ibm-spectrum-scale-must-gather`** image:

   ```
   oc adm must-gather --image=icr.io/cpopen/ibm-spectrum-scale-must-gather:v5.2.0.0
   ```

2. Once completed, a new directory with **`must-gather`** prefix is created in your working directory. For example:

   ```
   # ls -ltr
   drwxr-xr-x  3 root root       229 Jun 14 09:11 must-gather.local.681612165636007567
   ```

3. Create a compressed file from the must-gather directory that was just created in your working directory.

   ```
   tar cvaf must-gather.tar.gz must-gather.local.681612165636007567/
   ```

   Replace the directory name used in this example with your respective must-gather directory.

Generating GPFS trace reports

Some issues might require low-level system detail accessible only through the IBM Storage Scale daemon and the IBM Storage Scale Linux kernel trace facilities.

In such instances the IBM Support Center might request such GPFS trace reports to facilitate rapid problem determination of failures.

The level of detail that is gathered by the trace facility is controlled by setting the trace levels using the mmtracectl command. For more information, see mmtracectl command in IBM Storage Scale documentation.

The following steps must be performed under the direction of the IBM Support Center.

1. Enter the following command to access a running ibm-spectrum-scale-core pod:

   ```
   oc rsh -n ibm-spectrum-scale <ibm-spectrum-scale-core-pod>
   ```

   The pod must be in **Running** status to connect. It is best to pick a pod running on a node that is not exhibiting issues.

   The remaining steps should be completed while connected to this shell running inside the **gpfs** container of this running core pod.

2. Enter the **mmchconfig** command to change the **dataStructureDump** field to point to **/var/adm/ras**. This changes the default location where trace data is stored to a directory that persists on the host machine:

   ```
   mmchconfig dataStructureDump=/var/adm/ras/
   ```

3. Set desired trace classes and levels.

   This part of the process is identical to classic IBM Storage Scale installs. For more information, see Generating GPFS trace reports in IBM Storage Scale documentation.

   ```
   mmtracectl --set --trace={io | all | def | "Class Level [Class Level ...]"}
   ```

4. Start the trace facility on all nodes by entering the following command:

   ```
   mmtracectl --start
   ```

5. Re-create the problem.

6. Stop the trace generation as soon as the problem to be captured occurs, by entering the following command:

   ```
   mmtracectl --stop
   ```

7. Turn off trace generation by entering the following command:

   ```
   mmtracectl --off
   ```

Kernel crash dumps

Red Hat Enterprise Linux CoreOS (RHCOS) based machines do not support configuring kdump or generating kernel crash dumps for Red Hat OpenShift Container Platform 4.6 and earlier. For more information, see How to configure kdump in Red Hat CoreOS in Red Hat OpenShift documentation.

In some virtual machine installations, it may be possible to generate a vmcore crash dump from the hypervisor.

In lieu of kernel dumps, CoreOS currently recommends using pstore, even if only small snippets of diagnostic data can be collected. For more information, see Using pstore in CoreOS documentation on GitHub.

References

- IBM Storage Scale
- Red Hat OpenShift or Kubernetes

IBM Storage Scale

- Administration Guide
- For Linux on Z: Changing the kernel settings
- mmchconfig command
- mmnetverify command
- Accessing a remote GPFS file system
- Defining the cluster topology for the installation toolkit
- Node quorum
- Installing IBM Storage Scale Container Storage Interface driver by using CLIs

Red Hat OpenShift or Kubernetes

- Display which Pods have the PVC in use
- Red Hat OpenShift Container Platform 4 now defaults to CRI-O as underlying container engine
- How to configure kdump in Red Hat CoreOS?
- Installing and configuring OpenShift Container Platform clusters
- Installation configuration
- Configuring an htpasswd identity provider

Product documentation in guide format

Table 1. Product documentation in PDF format

Links
- IBM Storage Scale Container Native Storage Access 5.2.0
- IBM Storage Scale Container Native Storage Access 5.1.9
- IBM Storage Scale Container Native Storage Access 5.1.7
- IBM Storage Scale Container Native Storage Access 5.1.6
- IBM Storage Scale Container Native Storage Access 5.1.5
- IBM Storage Scale Container Native Storage Access 5.1.4
- IBM Storage Scale Container Native Storage Access 5.1.3
- IBM Storage Scale Container Native Storage Access 5.1.2