IBM Spectrum Scale Container Native Storage Access Guide 5.1.2.1

SC28-3168-14

Tables of Contents

License information

The License information of IBM Spectrum® Scale container native includes the following topic:

- Edition notice

Edition notice

Note: Before using this information and the product it supports, read the information in Notices.

This edition applies to Version 5 release 1 modification 1 of the following products, and to all subsequent releases and modifications until otherwise indicated in new editions:

• IBM Spectrum Scale Data Management Edition ordered through Passport Advantage® (product number 5737-F34)

• IBM Spectrum Scale Data Access Edition ordered through Passport Advantage (product number 5737-I39)

• IBM Spectrum Scale Erasure Code Edition ordered through Passport Advantage (product number 5737-J34)

Overview

The overview of IBM Spectrum Scale® container native includes the following topics:

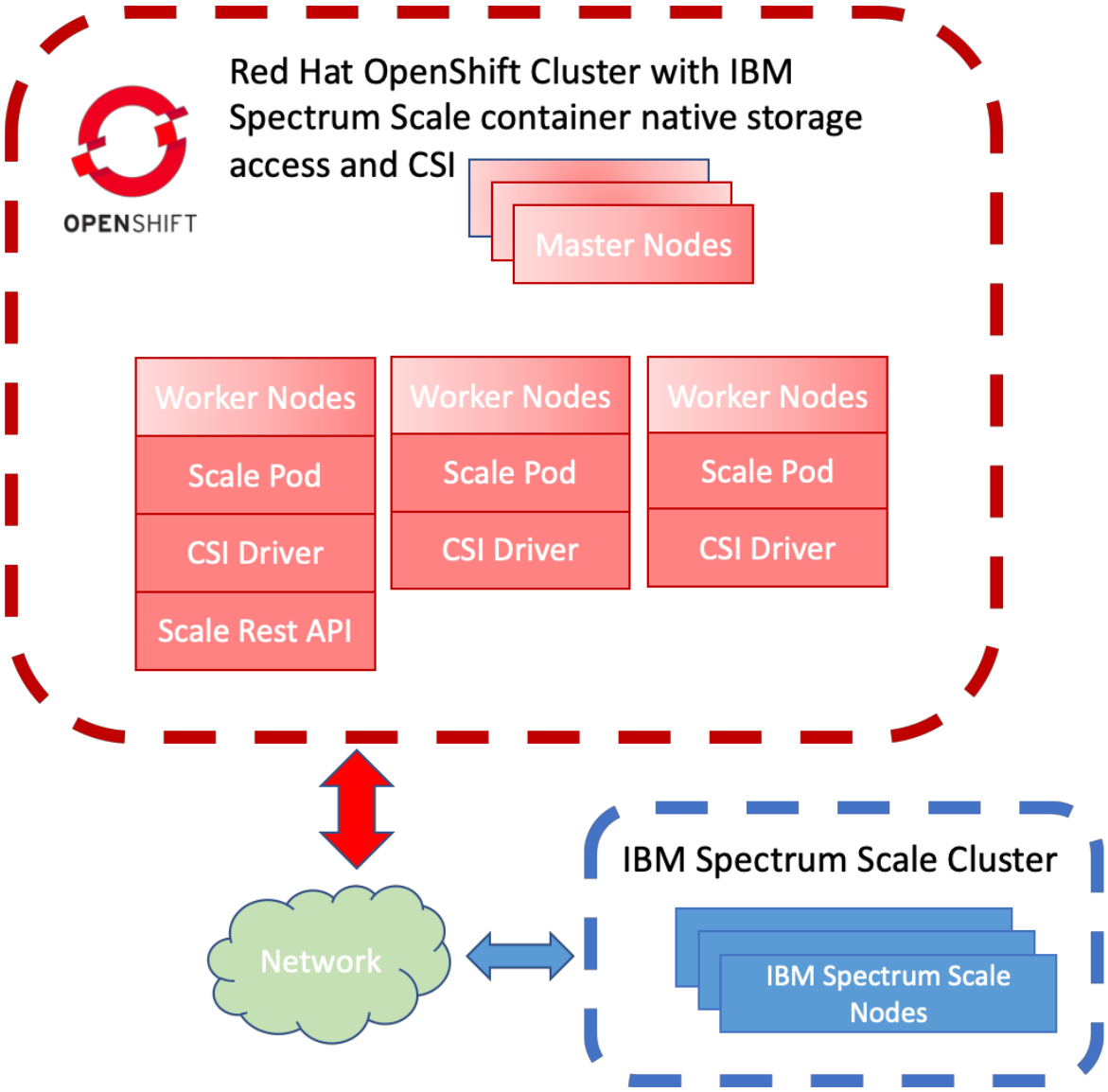
- Introduction
- What's new?
- Supported features
- Limitations

Introduction

IBM Spectrum® Scale container native is a containerized version of IBM Spectrum Scale.

IBM Spectrum 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. This 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 Spectrum Scale features, see Product overview in IBM Spectrum Scale documentation.

IBM Spectrum Scale container native allows the deployment of the cluster file system in a Red Hat® OpenShift® cluster. Using a remote mount attached file system, the container native deployment provides a persistent data store to be accessed by the applications through the IBM Spectrum Scale CSI driver by using Persistent Volumes (PVs). For more information, see IBM Spectrum Scale Container Storage Interface Driver in IBM Spectrum Scale CSI documentation.



What's new?

The following enhancements are made in this release:

- Direct storage attachment on x86 and power servers.
- Ability to upgrade IBM Spectrum® Scale container native from 5.1.1.4 to 5.1.2.1. For more information, see Upgrade IBM Spectrum Scale container native.
- Automatic quorum selection is Kubernetes topology aware.

Supported features

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

- IBM Spectrum Scale node labels to establish node affinity
- Automated client-only cluster creation

- Automated remote file system mount for IBM Spectrum Scale Storage cluster
- Integrated IBM Spectrum Scale Container Storage Interface (CSI) driver for application persistent storage
- Automated deployment of IBM Spectrum Scale Container Storage Interface (CSI) driver
- IBM Spectrum 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 Spectrum 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, and power servers
- Automatic quorum selection is Kubernetes topology aware

Limitations

- IBM Spectrum Scale container native currently supports only remote mount of the file system. It does not support local disks and NSD nodes.

Description	Max supported
Number of worker nodes	128
Number of remote clusters	4
Number of remote file systems	16

Planning

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

- Prerequisites
- Hardware requirements
- Software requirements
- Deployment considerations
- Container image list for IBM Spectrum Scale container native
- Roles and personas

Prerequisites

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

These steps are built on top of each other, so it is critical to follow the sequence defined in the following sections. Before you begin installation, there are several things that needs to be considered. The list of questions provided helps you to be prepared for the procedure.

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

Preparations for deploying the IBM Spectrum Scale container native cluster

Complete the following steps:

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

   Note: 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.

2. Validate and apply the configuration to the Red Hat OpenShift installation settings.

3. Obtain IBM Cloud Container Registry entitlement key in order to access the container images of IBM Spectrum Scale container native.

4. If you are in a restricted network environment, then mirror the container images of IBM Spectrum Scale container native into a site-managed private image registry.

5. Create an OpenShift global pull secret for the image registry that the cluster uses (either IBM Cloud Container Registry or private image registry).

Deploying the IBM Spectrum Scale container native cluster

To deploy a cluster, complete the following steps:

1. Create the IBM Spectrum Scale container native and IBM Spectrum Scale CSI operators by deploying the operator installer file.

2. Download the sample YAML file from the GitHub repository.

   Note: The sample file **`scale_v1beta1_cluster_cr.yaml`** is a collection of multiple custom resources and kinds.

a. Configure the Cluster custom resource that is used for deployment of the Operator.

`i. Specify the IBM Spectrum Scale Edition in the license field.`

`ii. Configure appropriate node selectors for the IBM Spectrum Scale container native deployment.`

`iii. Configure host aliases (or ensure that proper DNS is configured for your environment) to allow for communication to storage cluster.`

`iv. Configure Ephemeral Port Range, if necessary.`

v. Enable the optional Grafana Bridge.

b. Configure the Callhome custom resource.

c. Configure the Filesystem custom resource.

i. Define the RemoteCluster resource.

ii. Define the file system on the RemoteCluster to mount.

d. Configure the RemoteCluster custom resource.

i. Populate the details of the storage cluster GUI.

3.Create the IBM Spectrum Scale container native cluster by deploying the configured **scale_v1beta1_cluster_cr.yaml** file.

4.If accessing encrypted data on the storage cluster, download and configure the **EncryptionConfig** custom resource YAML file from the GitHub repository.

5.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 storage cluster.

6.Create a secret using the storage cluster GUI user credentials for **ContainerOperator** GUI user in the **ibm-spectrum-scale** namespace.

7.Create a secret using the storage cluster GUI user credentials for **CsiAdmin** GUI user in the **ibm-spectrum-scale-csi** namespace.

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

Hardware requirements

Note: IBM Spectrum Scale container native 5.1.2.1 supports only on-premises environments (customer infrastructure) and does not support Cloud environments.

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.
- RDMA for InfiniBand or RoCE for Ethernet is not supported.

Worker node requirements

The following table lists the minimum and recommended worker node requirements per each OpenShift Container Platform worker node.

IBM Spectrum Scale container native running on OpenShift Container Platform recommends a minimum of 3 worker nodes for the cluster. It supports a maximum of 128 worker nodes.

Architecture	Minimum		Recommended	
	CPU (cores)	Memory (GB)	CPU (cores)	Memory (GB)
x86_64	8	16	16	64
PPC64LE	8	16	16	64
s390x	4	8	8	16

IBM Spectrum Scale storage cluster (remote cluster)

The IBM Spectrum Scale storage cluster that is used as the remote cluster must run IBM Spectrum Scale 5.1.2.1 or later.

Software requirements

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

IBM Spectrum Scale container native	OpenShift Container Platform	IBM Spectrum Scale Container Storage Interface	IBM Spectrum Scale remote cluster	File system version cannot be newer than
5.1.2.1	4.7, 4.8, 4.9	2.4.0	5.1.2.1+	26.00
5.1.1.4	4.7, 4.8	2.3.1	5.1.1.4+	25.00
5.1.1.3	4.7, 4.8	2.3.0	5.1.1.3+	25.00
5.1.1.1	4.6.6+, 4.7 ,4.8	2.2.0	5.1.0.1+, 5.1.1.0+*	25.00
5.1.0.3	4.6.6+, <=4.7.23	2.2.0	5.1.0.1+, 5.1.1.0+*	24.00
5.1.0.3	4.5, 4.6.6+	2.1.0	5.1.0.1+	24.00

Note: * indicates pre-req of this code level or higher for CSI snapshots.

For more information about compatibility and software matrix, see Section 17.3 in IBM Spectrum Scale FAQ documentation.

Red Hat® OpenShift® Container Platform

Following are the supported Red Hat OpenShift Container Platform versions:

- 4.7
- 4.8
- 4.9

IBM Spectrum Scale Container Storage Interface (CSI)

- CSI 2.4.0 is installed in conjunction with IBM Spectrum Scale container native 5.1.2.1, and requires IBM Spectrum Scale storage cluster running 5.1.2.1 or later.

Remote cluster

- The remotely mounted file system should be at file system format level 26.00 or earlier. (26.00 was introduced with 5.1.2).
- When you are upgrading the remote cluster to any version of 5.1.2, run `mmchfs -V compat` to retain accessibility of the file system from the container native client cluster.

External container images

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

Auxiliary helper applications

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

Deployment considerations

Before deployment, ensure that you are aware of the Red Hat® OpenShift® version, Red Hat OpenShift cluster persistent storage, and storage cluster considerations.

The following list includes the Red Hat OpenShift cluster considerations:

- The IBM Spectrum Scale pods use the host network. They do not use the Container Network Interface (CNI) overlay network.
- The DNS must be configured properly so that the worker nodes can resolve the storage cluster nodes. For more information, see Host aliases.
- All worker nodes must be able to communicate with each other through the host network.
- Red Hat Enterprise Linux® CoreOS (RHCOS) restricts new file system mounts to the `/mnt` subtree. IBM Spectrum® Scale can mount any file system under `/mnt` on the Red Hat OpenShift cluster regardless of the default mount point that is defined on the storage cluster.
- A minimum configuration of three master nodes and three worker nodes, with a maximum of 128 worker nodes is required.

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

- The IBM Spectrum Scale pods use host path mounts to store IBM Spectrum Scale cluster metadata and various logs.
- The IBM Spectrum Scale container native operator creates two local PersistentVolumes (PVs) on two eligible worker nodes. At least 25 GB 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 Spectrum 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 Spectrum Scale operator.
- IBM Spectrum Scale container native does not support the use of dynamically created or pre-created PVs.

The following list includes the storage cluster considerations:

- The storage cluster must be at IBM Spectrum Scale 5.1.2.1 or later to support remote mount through the Operator.

- The remotely mounted file system should be at file system format level 26.00 or earlier. (26.00 was introduced with 5.1.2).
- Encrypted file systems are supported. Configure the EncryptionConfig custom resource with the necessary key client and key server information. For more information, see EncryptionConfig.

  Note: The remote mount file system level must be at a maximum 5.1.2 file system format level 26.00. Any file system exceeding this maximum results in remote mount failure. When you are upgrading the remote cluster to any version of IBM Spectrum Scale 5.1.2 or later, ensure to run the **`mmchfs -V compat`** command to retain access of the file system from the container native client cluster.

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

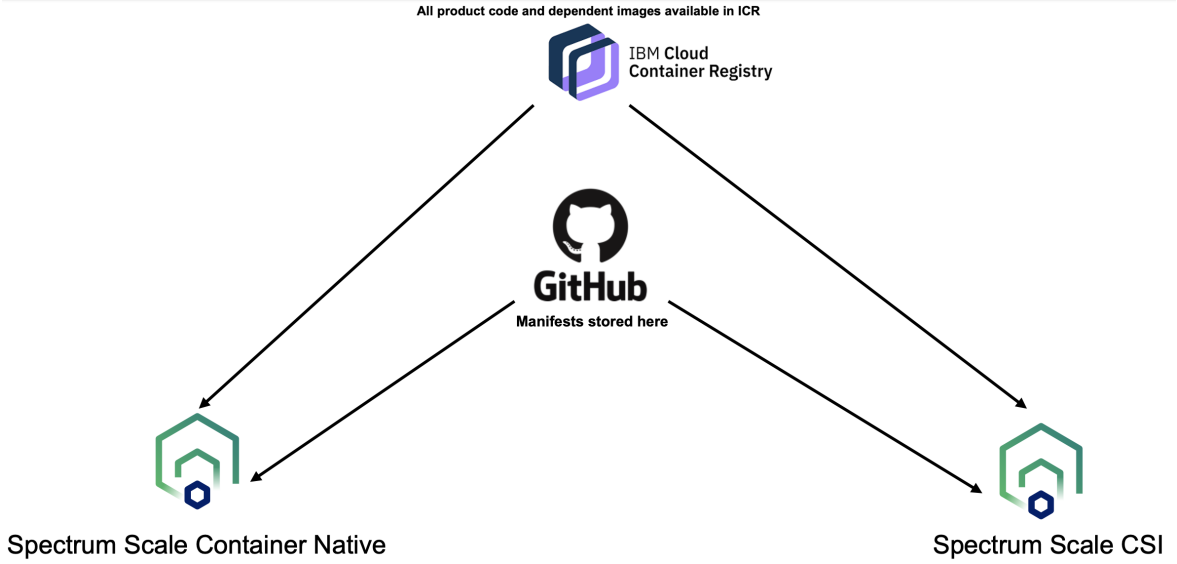
- In a restricted network environment where the Red Hat OpenShift Container Platform cluster cannot pull IBM Spectrum Scale images from the IBM Container Repository, images must be mirrored to a production grade enterprise image registry that the Red Hat OpenShift Container Platform cluster can access.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

The following list includes the considerations for direct storage attachment on IBM Z:

- Support for direct storage attachment on x86 and power servers.
- The virtualization layers of an IBM Z server allow the physical connection of the disks containing the IBM Spectrum Scale file system data to both the storage cluster and the IBM Spectrum Scale container native cluster.
- In direct storage attachment configuration, the worker nodes use the SAN fabric instead of the IBM Spectrum Scale NSD protocol for I/O traffic. For more information about setting up a direct storage attachment, see Attaching direct storage on IBM Z in IBM Spectrum Scale documentation.

Container image list for IBM Spectrum Scale container native

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



IBM Spectrum Scale images acquired from non-entitled IBM Container Repository

The images listed in the following table are the container images that are obtained through the IBM Container Repository that do not require entitlement. These images can be anonymously pulled.

Pod	Container	Repository	Image
ibm-spectrum-scale-controller-manager-XXXXXXXXX-XXXXX	manager	icr.io/cpopen	ibm-spectrum-scale-operator@sha256:e986a2664fa450c8b1ef028deb5fc8a27662802fc2cd804211cdd5184c50486b
ibm-spectrum-scale-csi-operator	operator	icr.io/cpopen	ibm-spectrum-scale-csi-operator@sha256:38751e2b7a4624e588747ed427c2c2146bee320bab74b0bb288f38c2c5d2bddd

IBM Spectrum Scale images acquired from entitled IBM Container Repository

The images listed in the following table are the container images that are obtained through entitlement to the IBM Container Repository.

Pod	Container	Repository	Image
workerX/masterX*	mmbuildgpl	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-core-init@sha256:1d1f2e1d2c035d1c58aac20961071c8cd011852acfaf977928f0de2ee536279c
workerX/masterX*	config	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-core-init@sha256:1d1f2e1d2c035d1c58aac20961071c8cd011852acfaf977928f0de2ee536279c
workerX/masterX*	gpfs (if using Data Access Edition)	cp.icr.io/cp/spectrum/scale/data-access	ibm-spectrum-scale-daemon@sha256:27d65e1edeaa656ea091f410d673ca8af2701375d254b181a726c63c96141e5b

Pod	Container	Repository	Image
workerX/masterX*	gpfs (if using Data Management Edition)	cp.icr.io/cp/spectrum/scale/data-management	ibm-spectrum-scale-daemon@sha256:6d477c4115ab84777f199e0dd2cf4ba40bd3825d903498667de414b890b66bff
workerX/masterX*	logs	cp.icr.io/cp/spectrum/scale	ubi-minimal@sha256:d9b92ea78e76300968f5c9a4a04c2cf220a0bbfac667f77e5e7287692163d898
ibm-spectrum-scale-gui-X	liberty	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-gui@sha256:8f6a4df6f3906ddac08d25dec206a4f89ac6a8d5529518013a7301132241f7c2
ibm-spectrum-scale-gui-X	sysmon	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-monitor@sha256:48a1bebb89804c11593d3577107bedba1a6a40157298d81bfef4983c9744a9b6
ibm-spectrum-scale-gui-X	postgres	cp.icr.io/cp/spectrum/scale	postgres@sha256:a2da8071b8eba341c08577b13b41527eab3968bf1c8d28123b5b07a493a26862
ibm-spectrum-scale-gui-X	logs	cp.icr.io/cp/spectrum/scale	ubi-minimal@sha256:d9b92ea78e76300968f5c9a4a04c2cf220a0bbfac667f77e5e7287692163d898
ibm-spectrum-scale-pmcollector-X	pmcollector	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-pmcollector@sha256:9f21c31541a1e85c7f9eae2a695b19c57b85a7731286a277f9cd5184e8c87323
ibm-spectrum-scale-pmcollector-X	sysmon	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-monitor@sha256:48a1bebb89804c11593d3577107bedba1a6a40157298d81bfef4983c9744a9b6
ibm-spectrum-scale-csi-snapshotter	csi-snapshotter	cp.icr.io/cp/spectrum/scale/csi	csi-snapshotter@sha256:818f35653f2e214db81d655063e81995de9073328a3430498624c140881026a3
ibm-spectrum-scale-csi-attacher	ibm-spectrum-scale-csi-attacher	cp.icr.io/cp/spectrum/scale/csi	csi-attacher@sha256:80dec81b679a733fda448be92a2331150d99095947d04003ecff3dbd7f2a476a
ibm-spectrum-scale-csi-provisioner	csi-provisioner	cp.icr.io/cp/spectrum/scale/csi	csi-provisioner@sha256:6477988532358148d2e98f7c747db4e9250bbc7ad2664bf666348abf9ee1f5aa
ibm-spectrum-scale-csi-resizer-X	ibm-spectrum-scale-csi-resizer	cp.icr.io/cp/spectrum/scale/csi	csi-resizer@sha256:6e0546563b18872b0aa0cad7255a26bb9a87cb879b7fc3e2383c867ef4f706fb
ibm-spectrum-scale-csi-driver-XXXXX	driver-registrar	cp.icr.io/cp/spectrum/scale/csi	csi-node-driver-registrar@sha256:f9bcee63734b7b01555ee8fc8fb01ac2922478b2c8934bf8d468dd2916edc405
ibm-spectrum-scale-csi-driver-XXXXX	ibm-spectrum-scale-csi	cp.icr.io/cp/spectrum/scale/csi	ibm-spectrum-scale-csi-driver@sha256:4d8c41138f2fddac351f82db19c32fe5ad1282e7886f78fe2669f0c30ea5badb
ibm-spectrum-scale-csi-driver-XXXXX	liveness-probe	cp.icr.io/cp/spectrum/scale/csi	livenessprobe@sha256:529be2c9770add0cdd0c989115222ea9fc1be430c11095eb9f6dafcf98a36e2b
must-gather-XXXXX	must-gather	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-must-gather@sha256:4e344cf5c694e8d17d532b1e709095bd319644a02165f825d949a43ffbf50622

Pod	Container	Repository	Image
ibm-spectrum-scale-grafana-bridge-X	grafanabridge	cp.icr.io/cp/spectrum/scale	ibm-spectrum-scale-grafana-bridge@sha256:a0bfc1ef649070fdee0ad23746e4f3c438d6798d7c3148ead7149496110e1801

**Pod names that contain the mmbuildgpl, config, and gpfs containers may vary. The pod name is based on the shortname of the node it was scheduled to.*

> **NOTE:** This list is for information only. No user action is required to obtain or define this list of images when in a non-airgapped environment. There are instructions to mirror the list of images in an air gap environment. For more information, see Air gap setup for network restricted Red Hat OpenShift Container Platform clusters.

Roles and persona

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

Persona

Red Hat OpenShift Cluster administrator must deploy the IBM Spectrum Scale container native cluster.

Operator permissions

The IBM Spectrum Scale container native operator is a namespace-scoped operator. The operator watches the namespace that it is deployed into. As part of the operator installation, you can deploy various role-based access control (RBAC) related YAML files that control the operator's access to resources within the namespace it is watching. While the operator is running with a namespace scope, it requires access to cluster level resources to successfully deploy. Access to cluster level resources is handled through a cluster role that is deployed during the deployment of RBAC YAML files. The role and cluster role are bound to the custom **`ibm-spectrum-scale-operator`** ServiceAccount, which the operator uses to create the IBM Spectrum Scale container native cluster.

ibm-spectrum-scale-operator role

Resources	Verbs	API Groups
pods, pods/exec, services, serviceaccounts, configmaps, secrets, services/finalizers	*	-
roles, rolebindings	*	rbac.authorization.k8s.io
leases	get, create, update	coordination.k8s.io
daemonsets, replicasets, statefulsets	*	apps
servicemonitors	get, create	monitoring.coreos.com
deployments, deployments\finalizers (resourceName=ibm-spectrum-scale-operator only)	get, update	apps
*	*	ibm.com
scaleclusters/status	get, patch, update	scale.ibm.com
scaleclusters, scaleclusters/finalizers	create, delete, get, list, patch, update, watch	scale.ibm.com

ibm-spectrum-scale-operator cluster role

Resources	Verbs	API Groups
nodes, services, events	get, list, create, patch, watch	-
persistentvolumes, persistentvolumes/finalizers, persistentvolumeclaims	get, list, create, patch, delete	-
statefulsets	get	apps
securitycontextconstraints	get, list, watch, create, update, patch, delete	security.openshift.io
storageclasses	get, list, patch, create	storage.k8s.com
clusterroles, clusterrolebindings	get, list, watch, create, update, patch, delete	rbac.authorization.k8s.io

Core pod permissions

You can collect a gpfs.snap from any running Spectrum Scale core pod for diagnostic log collection when seeking problem determination. The gpfs.snap contains both gpfs logs and captured output relevant to kubernetes and OpenShift resources. In order to successfully query Kubernetes and OpenShift resources, the daemonset must be given permission to access said resources. This permission is given by a role that is bound to the **`ibm-spectrum-scale-core`** service account, which is used exclusively by the daemonset.

Resources	Verbs	API Groups
pods, services	get, list	-
deployments, statefulsets	get, list	apps

Installation prerequisites

Prior to the installation of IBM Spectrum® Scale container native, the following are the prerequisites:

- Red Hat OpenShift Container Platform configuration
  - Compact clusters support
- IBM Cloud container registry
  - IBM Cloud Container Registry (ICR) entitlement
  - Adding IBM Cloud container registry credentials
  - Air gap setup for network restricted Red Hat OpenShift Container Platform clusters

Red Hat OpenShift Container Platform configuration

You must modify the Red Hat OpenShift Container Platform installation for IBM Spectrum® Scale container native to operate correctly.

For more information, see Installing in Red Hat OpenShift documentation.

For the instructions that follow, it is assumed that the Red Hat OpenShift Container Platform is already installed.

> **Note:** The configuration tasks shown 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.

Applying the machine configuration provided drives a rolling update of the OpenShift nodes and could take several minutes to complete. For the new configuration to take effect, the nodes within the pool must be rebooted. On applying the supplied YAML files, you can complete the following tasks:

- **Increase pids_limit**: Increase the `pids_limit` to 4096. Without this change, the GPFS daemon crashes during I/O by running out of PID resources.
- **Kernel Devel/Header Packages**: Install the kernel related packages for IBM Spectrum Scale to successfully build its portability layer.
- **Increase vmalloc kernel parameter**: Modify the kernel parameters that are required to operate properly with Red Hat CoreOS. It applies only to the IBM Spectrum Scale running on Linux® on Z.

Applying Machine Config Operator (MCO) Settings

Apply the following set of MCO settings depending on your OCP version and machine's architecture:

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

  For 4.7:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.7/mco_x86_64.yaml
  ```

  For 4.8:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.8/mco_x86_64.yaml
  ```

  For 4.9:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.9/mco_x86_64.yaml
  ```

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

  For 4.7:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.7/mco_ppc64le.yaml
  ```

  For 4.8:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.8/mco_ppc64le.yaml
  ```

  For 4.9:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.9/mco_ppc64le.yaml
  ```

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

  For 4.7:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.7/mco_s390x.yaml
  ```

  For 4.8:

  ```
  oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.8/mco_s390x.yaml
  ```

For 4.9:

```
 oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.9/mco_s390x.yaml
```

Verifying Machine Config Operator (MCO) settings

Complete the following steps:

1. Check the status of the update by entering the following command:

   ```
   oc get MachineConfigPool
   ```

2. Verify that the **pids_limit** is increased on the worker nodes by entering the following command:

   ```
   oc get nodes -lnode-role.kubernetes.io/worker= \
   -ojsonpath="{range .items[*]}{.metadata.name}{'\n'}" |\
   xargs -I{} oc debug node/{} -T -- chroot /host crio-status config | grep pids_limit
   ```

   Note: This command runs through all the worker nodes. Use it with discretion if you have a large system.

3. Enter the following command to validate that the **Kernel-devel** package is successfully applied on the Red Hat OpenShift container worker nodes.

   ```
   oc get nodes -lnode-role.kubernetes.io/worker= \
   -ojsonpath="{range .items[*]}{.metadata.name}{'\n'}" |\
   xargs -I{} oc debug node/{} -T -- chroot /host sh -c "rpm -q kernel-devel"
   ```

4. If running s390x, validate that the machine config has the vmalloc kernel parameter set by entering the following command:

   ```
   oc describe machineconfig | grep vmalloc
   ```

5. If running s390x, 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 -lnode-role.kubernetes.io/worker= \
   -ojsonpath="{range .items[*]}{.metadata.name}{'\n'}" |\
   xargs -I{} oc debug node/{} -T -- cat /proc/cmdline
   ```

   You can see **vmalloc=4096G** in the following output:

   ```
   # 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
   ```

Compact clusters support

You can deploy compact-3-node clusters on resource constrained environments in 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.

While not advised, IBM Spectrum Scale container native can be configured to be scheduled in an OpenShift Container Platform compact cluster environment. You must not run user container workload on the Control Plane. For more information, see Control Plane Components in Kubernetes documentation.

Schedulable control plane nodes

To allow pod placement for master nodes (also known as control plane nodes), ensure that they are configured as Schedulable. By default, control plane nodes are not schedulable.

Verify that **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 Red Hat OpenShift documentation.

Applying Machine Config Operator (MCO) settings

Similar to the configuration tasks that are required for the workers nodes, these MCO settings must also be applied to the master nodes in a compact-cluster environment. For more information, see Red Hat OpenShift Container Platform configuration.

You can take the sample mco **yaml** files as a base template that can be modified and applied to your cluster.

1. Download the correct sample file based on your OCP version and machine architecture and save it as **master_mco.yaml**.

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

     For 4.7:

     ```
     curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-
     container-native/v5.1.2.1/generated/mco/ocp4.7/mco_x86_64.yaml >
     master_mco.yaml
     ```

     For 4.8:

     ```
     curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-
     container-native/v5.1.2.1/generated/mco/ocp4.8/mco_x86_64.yaml >
     master_mco.yaml
     ```

     For 4.9:

     ```
     curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-
     container-native/v5.1.2.1/generated/mco/ocp4.9/mco_x86_64.yaml >
     master_mco.yaml
     ```

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

     For 4.7:

     ```
     curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-
     container-native/v5.1.2.1/generated/mco/ocp4.7/mco_ppc64le.yaml >
     master_mco.yaml
     ```

     For 4.8:

```
curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.8/mco_ppc64le.yaml > master_mco.yaml
```

For 4.9:

```
curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.9/mco_ppc64le.yaml > master_mco.yaml
```

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

  For 4.7:

  ```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.7/mco_s390x.yaml > master_mco.yaml
  ```

  For 4.8:

  ```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.8/mco_s390x.yaml > master_mco.yaml
  ```

  For 4.9:

  ```
  curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/mco/ocp4.9/mco_s390x.yaml > master_mco.yaml
  ```

2. Modify the sample file for **master** role and apply to your cluster:

   ```
   cat master_mco.yaml | sed 's/worker/master/g' | oc apply -f -
   ```

3. Validate MCO settings against the **master** pool. For more information, see Verifying Machine Config Operator (MCO) Settings.

4. Remove the **node-role.kubernetes.io/worker: ""** selector from the default **Cluster CR** node selector. Removing this selector enables the deployment of IBM Spectrum Scale **core** pods on master and worker nodes. For more information, see Node Selectors.

Obtaining deployment image from IBM Cloud Container Registry

Starting with IBM Spectrum Scale container native 5.1.1.1, the container images have moved from Fix Central to the IBM Cloud Container Registry.

Note: If your cluster is already configured with IBM Cloud Container Registry, you do not need to create an entitlement key nor create the global pull secret since they already exist there.

- IBM Cloud Container Registry (ICR) entitlement
- Adding IBM Cloud container registry credentials
- Air gap setup for network restricted Red Hat OpenShift Container Platform clusters

IBM Cloud Container Registry (ICR) entitlement

To obtain an entitlement key, complete the following steps:

1. Log in to the IBM container software library with an IBM id and a password that is associated with the entitled software.
2. Click **Get entitlement key** on the left navigation bar.
3. On the **Access your container software** page, click **Copy key** to copy the generated entitlement key.
4. Save the key to a secure location for future use.

   Note: Entitlement keys determine whether the IBM Spectrum Scale operator can automatically pull the required IBM Spectrum Scale container native images. During installation, image pull failures may occur due to an invalid entitlement key or a key belonging to an account that does not have entitlement to either IBM Spectrum Scale Data Access Edition or IBM Spectrum Scale Data Management Edition. It is therefore important to generate a key from an account that already has entitlement to the desired edition of IBM Spectrum Scale software.

Adding IBM Cloud Container Registry credentials

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

Note: The following steps are for users whose OpenShift cluster is accessing the IBM Cloud Container Registry. For more information, see Air gap setup for network restricted Red Hat OpenShift Container Platform clusters.

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

   - For using IBM Cloud Container Registry, the credentials are the fixed **cp** user and the generated entitlement key. For more information, see IBM Cloud Container Registry (ICR) entitlement.

     ```
     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 **cp.icr.io** repository), and generated 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. The following step takes the **authority.json** and include it as a new authority in your **.dockerconfigjson**, stored as a **temp_config.json**.

   Note: 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 -d - | \
   jq '.["cp.icr.io" += input' - authority.json > temp_config.json
   ```

   Note: This command is supported with **jq** 1.5.

- To verify that your authority credentials were created in the resulting file:

```
# cat temp_config.json
{
   "auths": {
      "quay.io": {
        "auth": "",
        "email": ""
      },
      "registry.connect.redhat.com": {
         "auth": "",
         "email": ""
      },
      "registry.redhat.io": {
         "auth": "",
         "email": ""
      },
      "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 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. The updated config is now rolled out to all the nodes in the OpenShift cluster. Nodes are cycled through one at a time and are not schedulable before rebooting. Enter the **watch oc get nodes** command to observe nodes.

```
 # oc get nodes

 NAME                                STATUS                          ROLES    AGE
VERSION
 master0.example.com            NotReady,SchedulingDisabled   master   99d
v1.19.0+43983cd
 master1.example.com            Ready                         master   99d
v1.19.0+43983cd
 master2.example.com            Ready                         master   99d
v1.19.0+43983cd
 worker0.example.com            NotReady,SchedulingDisabled   worker   99d
v1.19.0+43983cd
 worker1.example.com            Ready                         worker   99d
v1.19.0+43983cd
 worker2.example.com            Ready                         worker   99d
v1.19.0+43983cd
```

   Note: Red Hat OpenShift Container Platform 4.7 and above versions do not reboot the nodes. For more information, see Updating the global cluster pull secret in Red Hat OpenShift documentation.

6. When the global pull secret is updated, enter the following command to remove the temporary files that were created.

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

Air gap setup for network restricted Red Hat OpenShift Container Platform clusters (optional)

Air gap setup is done for Red Hat OpenShift Container Platform clusters that are in a restricted network environment.

Note: 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

Following are the prerequisites before setting up the air gap environment:

- A production grade Docker V2 compatible registry, such as Quay Enterprise, JFrog Artifactory, or Docker Registry. The Red Hat® OpenShift® Internal Registry is not supported.
- An online node that can copy images from the source image registry to the production grade internal image registry.
- The online node must have **skopeo** installed.
- Access to the Red Hat OpenShift Container Platform cluster as a user with the **cluster-admin** role.

Note: For Red Hat OpenShift Container Platform clusters that are in a restricted network environment, the obtained files must be transferred to a bastion/infrastructure node that can communicate with the target cluster before applying the **yaml** files. This is likely the same node in your Red Hat OpenShift Container Platform cluster where the **oc** command is executed.

Configuring the registry mirror

Create a new **ImageContentSourcePolicy** 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 **inf** node of your Red Hat OpenShift cluster:

1. Paste the following in a file (example: registrymirror.yaml) and replace your internal image registry repository with **example.io/subdir**:

```
apiVersion: operator.openshift.io/v1alpha1
kind: ImageContentSourcePolicy
metadata:
  name: icr-mirror
spec:
  repositoryDigestMirrors:
  - mirrors:
    - example.io/subdir
    source: cp.icr.io/cp/spectrum/scale
  - mirrors:
    - example.io/subdir
    source: icr.io/cpopen
```

Note: Do not prefix mirrors with **http://** or **https://** and ensure that they do not have trailing **/** characters as this causes issue while resolving them correctly.

2. Create the icr-mirror **ImageContentSourcePolicy** by entering the following command:

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

The mirror gets rolled out to all nodes in the OpenShift cluster. Nodes are cycled one at a time and are made unschedulable before rebooting.

3. Enter the following command to observe the nodes:

```
watch oc get nodes
```

Note: Red Hat OpenShift Container Platform 4.7 and later do not reboot the nodes.

4. Once all nodes have finished updating and rebooting, verify that the **ImageContentSourcePolicy** is applied by entering the **oc debug** command to query the mirrors on the host nodes.

```
$  oc debug node/worker0.subdomain
Starting pod/worker0examplecom-debug ...
To use host binaries, run `chroot /host`
Pod IP: 12.34.56.789
If you don't see a command prompt, try pressing enter.

# chroot /host
# cat /etc/containers/registries.conf
unqualified-search-registries = ["registry.access.redhat.com", "docker.io"]

[[registry]]
  prefix = ""
  location = "cp.icr.io/cp/spectrum/scale"
  mirror-by-digest-only = true

   [[registry.mirror]]
     location = "example.io/subdir"

[[registry]]
  prefix = ""
  location = "icr.io/cpopen"
  mirror-by-digest-only = true

   [[registry.mirror]]
     location = "example.io/subdir"
```

Note: For more information, see Configuring image registry repository mirroring in Red Hat OpenShift documentation.

Copying images from source image registry to target internal image registry

The OpenShift cluster is configured to redirect external image registry requests to an internal registry through the **ImageContentSourcePolicy**. Now, the internal registry must be populated with the images from the source image registry.

Complete the following steps from the **online** node described in the prerequisites:

1. Log in to the IBM® Entitled Container Registry with the credentials by entering the **skopeo** command.

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

2. Log in to your internal production grade image registry with the credentials by entering the **skopeo** command.

```
skopeo login example.io
```

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

```
icr.io/cpopen/ibm-spectrum-scale-
operator@sha256:e986a2664fa450c8b1ef028deb5fc8a27662802fc2cd804211cdd5184c50486
b
cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-core-
init@sha256:1d1f2e1d2c035d1c58aac20961071c8cd011852acfaf977928f0de2ee536279c
cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-
pmcollector@sha256:9f21c31541a1e85c7f9eae2a695b19c57b85a7731286a277f9cd5184e8c8
7323
cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-
monitor@sha256:48a1bebb89804c11593d3577107bedba1a6a40157298d81bfef4983c9744a9b6
cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-
gui@sha256:8f6a4df6f3906ddac08d25dec206a4f89ac6a8d5529518013a7301132241f7c2
cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-grafana-
bridge@sha256:a0bfc1ef649070fdee0ad23746e4f3c438d6798d7c3148ead7149496110e1801
cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-must-
gather@sha256:4e344cf5c694e8d17d532b1e709095bd319644a02165f825d949a43ffbf50622
cp.icr.io/cp/spectrum/scale/ubi-
minimal@sha256:d9b92ea78e76300968f5c9a4a04c2cf220a0bbfac667f77e5e7287692163d898

cp.icr.io/cp/spectrum/scale/postgres@sha256:a2da8071b8eba341c08577b13b41527eab3
968bf1c8d28123b5b07a493a26862
icr.io/cpopen/ibm-spectrum-scale-csi-
operator@sha256:38751e2b7a4624e588747ed427c2c2146bee320bab74b0bb288f38c2c5d2bdd
d
cp.icr.io/cp/spectrum/scale/csi/ibm-spectrum-scale-csi-
driver@sha256:4d8c41138f2fddac351f82db19c32fe5ad1282e7886f78fe2669f0c30ea5badb
cp.icr.io/cp/spectrum/scale/csi/csi-
snapshotter@sha256:818f35653f2e214db81d655063e81995de9073328a3430498624c1408810
26a3
cp.icr.io/cp/spectrum/scale/csi/csi-
provisioner@sha256:6477988532358148d2e98f7c747db4e9250bbc7ad2664bf666348abf9ee1
f5aa
cp.icr.io/cp/spectrum/scale/csi/csi-node-driver-
registrar@sha256:f9bcee63734b7b01555ee8fc8fb01ac2922478b2c8934bf8d468dd2916edc4
05
cp.icr.io/cp/spectrum/scale/csi/csi-
attacher@sha256:80dec81b679a733fda448be92a2331150d99095947d04003ecff3dbd7f2a476
a

cp.icr.io/cp/spectrum/scale/csi/livenessprobe@sha256:529be2c9770add0cdd0c989115
222ea9fc1be430c11095eb9f6dafcf98a36e2b
cp.icr.io/cp/spectrum/scale/csi/csi-
resizer@sha256:6e0546563b18872b0aa0cad7255a26bb9a87cb879b7fc3e2383c867ef4f706fb
```

To deploy a cluster using the Data Access edition of IBM Spectrum Scale container native, copy the following image:

```
cp.icr.io/cp/spectrum/scale/data-access/ibm-spectrum-scale-
daemon@sha256:27d65e1edeaa656ea091f410d673ca8af2701375d254b181a726c63c96141e5b
```

To deploy a cluster using the Data Management edition of IBM Spectrum Scale container native, copy the following image:

```
cp.icr.io/cp/spectrum/scale/data-management/ibm-spectrum-scale-
daemon@sha256:6d477c4115ab84777f199e0dd2cf4ba40bd3825d903498667de414b890b66bff
```

> **Note:** The destination is up to the user and depends on how the registry mirror was configured in the first section. Using the same **example.io/subdir** repository, a sample **skopeo copy** command is shown:

```
skopeo copy --all docker://cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-
gui@sha256:8f6a4df6f3906ddac08d25dec206a4f89ac6a8d5529518013a7301132241f7c2
docker://example.io/subdir/ibm-spectrum-scale-
gui@sha256:8f6a4df6f3906ddac08d25dec206a4f89ac6a8d5529518013a7301132241f7c2
```

> **Note:** The ibm-spectrum-scale-daemon image is edition specific. When copying it, you must put it in a folder that indicates its edition. The folder it resides in must be **data-access** or **data-management** depending on the image you are entitled to.

The sample command for copying the Data Access Edition **ibm-spectrum-scale-daemon** image is shown:

```
skopeo copy --all docker://cp.icr.io/cp/spectrum/scale/data-access/ibm-
spectrum-scale-
daemon@sha256:27d65e1edeaa656ea091f410d673ca8af2701375d254b181a726c63c96141e5b
docker://example.io/subdir/data-access/ibm-spectrum-scale-
daemon@sha256:27d65e1edeaa656ea091f410d673ca8af2701375d254b181a726c63c96141e5b
```

The sample command for copying the Data Management Edition **ibm-spectrum-scale-daemon** image is:

```
skopeo copy --all docker://cp.icr.io/cp/spectrum/scale/data-management/ibm-
spectrum-scale-
daemon@sha256:6d477c4115ab84777f199e0dd2cf4ba40bd3825d903498667de414b890b66bff
docker://example.io/subdir/data-management/ibm-spectrum-scale-
daemon@sha256:6d477c4115ab84777f199e0dd2cf4ba40bd3825d903498667de414b890b66bff
```

A generic **skopeo copy** command is shown:

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

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 production grade image registry by entering the **skopeo** command.

```
skopeo logout example.io
```

Testing the pull of images from the mirrored registry

Complete the following steps from the **inf** node of your OpenShift cluster:

1. Pick a worker node from **oc get nodes** and start a node to debug it.

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

A command prompt must be presented.

2. Switch to 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: 12.34.56.789
If you don't see a command prompt, try pressing enter.
# chroot /host
```

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

```
# podman login example.io
Username: sampleemail@email.com
```

```
Password:
Login Succeeded!
```

4. Attempt to pull one of the images from the source image registry through podman. The 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@sha256:8f6a4df6f3906ddac08d25dec206a4f89ac6a8d5529518013a7301132241f7c2
Trying to pull cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-
gui@sha256:8f6a4df6f3906ddac08d25dec206a4f89ac6a8d5529518013a7301132241f7c2...
Getting image source signatures
Copying blob 45cc8b7f2b43 skipped: already exists
Copying blob 5f6bf015319e skipped: already exists
Copying blob 1e0d1e43bdb2 done
Copying blob f5f17c204ecc done
Copying blob b89ea354ae59 done
Copying blob edccb152016f done
Copying blob 87212cfd39ea done
Copying blob 5627e846e80f done
Copying blob e7f8612e0600 done
Copying blob 9456cfefd278 done
Copying blob 81377630e23b done
Copying blob a85ce2cde74f done
Copying blob 058915423c66 done
Copying blob 415bf2dea3d3 done
Copying blob c28b6e27c8e1 done
Copying blob 8e2f6f43f11e done
Copying blob ee4bd9648715 done
Copying blob fb76f893efb9 done
Copying config 665f4935f0 done
Writing manifest to image destination
Storing signatures
665f4935f090de796531883c2472d35c662e5d4f0fe9da9ebaf336636334412d
```

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
```

Red Hat OpenShift Container Registry pull secret

For images to be properly pulled at the pod level, the OpenShift global pull secrets must be modified to contain credentials to access your internal container registry.

Note: For more information, see Adding IBM Cloud Container Registry credentials.

Complete the following steps:

1. Create a base64 encoded string of the credentials used to access your internal container registry.

   Note: 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 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**.

   Note: For the example internal container registry of **example.io/subdir**, use **example.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 -d - | \
jq '.["example.io" += input' - authority.json > temp_config.json
```

   Note: This command is supported with **jq** 1.5.

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

```
# cat temp_config.json
{
   "auths": {
      "quay.io": {
        "auth": "",
        "email": ""
      },
      "registry.connect.redhat.com": {
         "auth": "",
         "email": ""
      },
      "registry.redhat.io": {
         "auth": "",
         "email": ""
      },
      "example.io": {
         "auth": "<base64 encoded string created in previous step>",
         "username": "<example.io username>",
         "password": "<example.io password>"
      }
   }
}
```

4. Use the contents of the **temp_config.json** file, and apply the updated configuration to the 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, enter the following command and confirm your authority is present.

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

     The updated configuration is now rolled out to all nodes in the OpenShift cluster. Nodes are cycled one at a time and are made unavailable for scheduling before rebooting.

5. Enter the **watch oc get nodes** command to observe the nodes.

```
 # oc get nodes
 NAME                                STATUS                        ROLES    AGE
VERSION
 master0.example.com                 NotReady,SchedulingDisabled   master   99d
v1.19.0+43983cd
 master1.example.com                 Ready                         master   99d
v1.19.0+43983cd
 master2.example.com                 Ready                         master   99d
v1.19.0+43983cd
 worker0.example.com                 NotReady,SchedulingDisabled   worker   99d
v1.19.0+43983cd
 worker1.example.com                 Ready                         worker   99d
v1.19.0+43983cd
 worker2.example.com                 Ready                         worker   99d
v1.19.0+43983cd
```

Note: Red Hat OpenShift Container Platform 4.7 and above versions do not reboot the nodes. For more information, see Updating the global cluster pull secret in Red Hat OpenShift documentation.

6. When the global pull secret is updated, remove the temporary files that were created.

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

Installing the IBM Spectrum Scale container native operator and cluster

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

- Node labels and annotations
- Firewall recommendations
- IBM Spectrum Scale storage cluster configuration
- Deploy the operator
- Configuring the IBM Spectrum Scale container native cluster custom resources
  - Cluster
  - Callhome
  - Filesystems
  - Encryption
- Creating the IBM Spectrum Scale container native cluster
- Creating secrets for storage cluster GUI
- Configuring Certificate Authority (CA) certificates
- Verifying the IBM Spectrum Scale container native cluster
- Status and events

Labels and Annotations

IBM Spectrum Scale container native assigns labels to worker nodes and allows to set memory and CPU limits on a per node basis using node annotation.

Designation Labels

IBM Spectrum 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 using node labels.

The following mechanisms are supported to designate IBM Spectrum 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 Spectrum Scale node designations (like the quorum designation) to pods on specific worker nodes.

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.

Manual

Supported designation label values are **quorum** and **manager**. The nodes designated as **quorum** nodes also automatically assume the role of **manager**. If nodes are left without a designation label and sufficient **quorum** nodes are designated, unlabeled nodes become client nodes within the cluster.

IBM Spectrum Scale quorum

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

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 deployed on those nodes 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-
```

Note: Quorum node designations cannot be changed after creation of the IBM Spectrum Scale container native cluster.

Requests and Limits Annotations

For IBM Spectrum Scale container native core pods, you can override default memory and CPU resource requests. For more information, see Managing Resources for Containers in Kubernetes documentation.

Memory and CPU limits are automatically set to the capacity of the node and requests default to 25% of the node's capacity. However, requests may be specified on a per-node basis using the following node annotation.

Memory

- Enter the following command to set the memory request to 50G on **worker1** node:

  ```
  oc annotate nodes worker1.example.com scale.spectrum.ibm.com/memory="50G"
  ```

CPU

- Enter the following command to set the CPU request to 5 vCPU/Core on **worker2** node:

  ```
  oc annotate node worker2.example.com scale.spectrum.ibm.com/cpu="5"
  ```

  Note: The recommendation is to set the resource requests/limits at the role level in the cluster CR at **cluster.spec.daemon.roles.resources**.

Firewall recommendations

Ensure that ports 1191, 443, and the ephemeral port ranges are open on the storage cluster and on any network switches between the storage and container native cluster. Otherwise, the container native cluster can not remotely mount the file system from the storage cluster.

For more information, see Ephemeral port range.

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

IBM Spectrum Scale storage cluster configuration

Some additional tasks need to be performed on the IBM Spectrum Scale storage cluster in order for the container native cluster operator to configure the remote mount of the storage cluster file systems.

Creating Operator User and Group

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

1. To verify whether the IBM Spectrum Scale GUI user group **ContainerOperator** exists, enter the following command:

   ```
   /usr/lpp/mmfs/gui/cli/lsusergrp ContainerOperator
   ```

2. To create the **ContainerOperator** GUI user group if it does not exist, enter the following command:

   ```
   /usr/lpp/mmfs/gui/cli/mkusergrp ContainerOperator --role containeroperator
   ```

3. To verify whether an IBM Spectrum Scale GUI user exists within the **ContainerOperator** group, enter the following command:

   ```
   /usr/lpp/mmfs/gui/cli/lsuser | grep ContainerOperator
   ```

4. To create the GUI user for the **ContainerOperator** group, 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, then enter the following command to create the user with a password that never expires:

   ```
   /usr/lpp/mmfs/gui/cli/mkuser cnsa_storage_gui_user -p cnsa_storage_gui_password
   -g ContainerOperator -e 1
   ```

   Note: The **-e 1** parameter is only available for IBM Spectrum Scale storage cluster 5.1.1.0 or later.

Container Storage Interface (CSI) configuration

Complete the following steps on the *storage cluster* to ensure the IBM Spectrum Scale CSI driver is deployed successfully.

1. Create an IBM Spectrum® Scale user group **CsiAdmin** by entering the following command:

   ```
   /usr/lpp/mmfs/gui/cli/mkusergrp CsiAdmin --role csiadmin
   ```

2. Create an IBM Spectrum Scale user in the **CsiAdmin** group. This user must be used on IBM Spectrum Scale Container Storage Interface driver configuration. Enter this command on the GUI node to create the user.

   ```
   /usr/lpp/mmfs/gui/cli/mkuser csi-storage-gui-user -p  csi-storage-gui-password
   -g CsiAdmin
   ```

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

   ```
   $ mmlsfs fs1 --perfileset-quota
   flag                value                    description
   ------------------- ------------------------ -----------------------------------
   --perfileset-quota No                       Per-fileset quota enforcement
   ```

4. Enter the following command to enable the Quota in the file systems that is used by IBM Spectrum Scale Container Storage Interface driver.

   ```
   mmchfs fs1 -Q yes
   ```

5. Enter the following command to verify that the quota is enabled.

```
$ mmlsfs fs1 -Q

flag                value                      description
------------------- ------------------------ -----------------------------------
-Q                  user;group;fileset       Quotas accounting enabled
                      user;group;fileset       Quotas enforced
                      none                     Default quotas enabled
```

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

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

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

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

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

```
mmchfs fs1 --filesetdf
```

Deploy the operator

Deploy the IBM Spectrum Scale container native operator by entering the following command:

```
oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.1.2.1/generated/installer/ibm-spectrum-scale-operator.yaml
```

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

- **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
```

- **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
```

Configuring the IBM Spectrum Scale container native cluster custom resources

Before deploying the cluster, you need to make changes to the sample **scale_v1beta1_cluster_cr.yaml** file. Save the sample YAML file from the GitHub by entering the following command:

```
curl -fs https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/scale_v1beta1_cluster_cr.yaml > scale_v1beta1_cluster_cr.yaml || echo "Failed to curl Cluster CR"
```

This file is used to set configuration. When deployed, it initiates the IBM Spectrum Scale cluster creation.

The table below describes the custom resource definitions (CRDs) managed by the IBM Spectrum Scale container native operator:

Resource	Short name	Description
`cluster`	`gpfs`	Set attributes for the IBM Spectrum Scale container native cluster.
`callhome`	none	Configures IBM Spectrum Scale callhome functionality.
`filesystem`	`fs`	Configures remote mounted filesystems for the container native cluster.
`remoteclusters`	`remotegpfs`	Provide configuration to the remote cluster and establishes the secure authorizations. For more information, see Filesystem section.
`encryptionconfig`	`ec`	Allows users to configure encryption functionality.

The following sections guides through this process.

- Cluster
- Callhome
- Filesystems
- Encryption

Cluster

The sample `Cluster` custom resource can be found under `kind: Cluster` in the `scale_v1beta1_cluster_cr.yaml` file. For more information, see Configuring the IBM Spectrum Scale container native cluster custom resources. Once deployed, enter the `oc edit cluster` command to modify the properties.

Cluster status

Status	Description
Success	The `Cluster` is configured successfully.

Cluster spec

License

The `spec.license` section allows you to accept and choose the IBM Spectrum Scale edition that needs to be deployed in the IBM Spectrum Scale container native cluster. You must complete the following activities:

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

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

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

```
  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 Spectrum Scale core pods.

Node selectors

The **spec.daemon.nodeSelector** section allows you to configure a nodeSelector to determine where IBM Spectrum Scale pods can be deployed. The default location in the sample is to deploy core pods to kubernetes worker roles:

```
spec:
  daemon:
    ...
    nodeSelector:
      node-role.kubernetes.io/worker: ""
```

You may configure multiple node selector values by adding labels to the nodeSelector list. The Operator checks that a node has all defined labels present in order to deem a node eligible to deploy IBM Spectrum Scale pods. In the following example, the Operator deploys IBM Spectrum Scale pods on nodes with both the worker label and "scale" component label.

```
spec:
  daemon:
    ...
    nodeSelector:
      node-role.kubernetes.io/worker: ""
      app.kubernetes.io/component: "scale"
```

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

Host aliases

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

The **cluster.spec.daemon.hostAliases** section allows for entries to be created by kubernetes into the IBM Spectrum Scale **core** pod's **/etc/hosts** file.

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

```
spec:
  daemon:
    ...
    hostAliases:
      - hostname: node1.example.com
        ip: 10.0.0.1
      - hostname: node2.example.com
        ip: 10.0.0.2
```

The **hostAliases** section **DOES NOT** handle creating entries in **/etc/hosts** in any pods except for the **core** pods. For **RemoteCluster** CR, the hostname 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 **cluster.spec.daemon.clusterProfile** allows users to set default IBM Spectrum Scale configuration parameters for the cluster at cluster creation time.

> **Note:** Changing the values in the **clusterProfile** is not supported and must be avoided unless advised by IBM Support.

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

Ephemeral port range

If the storage cluster has the ephemeral port range configured, 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, 60000-61000, set this value under the **clusterProfile** section in the **Cluster** CR.

```
spec:
  daemon:
    clusterProfile:
      ...
      tscCmdPortRange: "60000-61000"
```

Roles

The **cluster.spec.daemon.roles** allow users to more finely tune resource target on nodes that are part of a specific role.

Changing the values in the **roles.profile** is not supported and must be avoided unless advised by IBM Support.

For example, to set a **memory** and **cpu** request target for the **client** role:

```
spec:
  ...
  daemon:
    ...
    roles:
    - name: client
      resources:
        memory: "40G"
        cpu: "4"
```

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

Grafana bridge

The **cluster.spec.grafanaBridge** section allows users to enable the deployment of the IBM Spectrum Scale bridge for Grafana application. For more information, see IBM Spectrum Scale bridge for Grafana repository in Github.

Enter the following command to enable Grafana Bridge:

```
spec:
  daemon:
  ...
  grafanaBridge: {}
```

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

Callhome

The sample **Callhome** custom resource can be found under **kind: Callhome** in the **scale_v1beta1_cluster_cr.yaml** file. For more information, see Configuring the IBM Spectrum Scale container native cluster custom resources. Fill out the following details to enable call home functionality.

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

> **Note:** You must always configure call home. If you choose not to configure call home, delete or comment out the call home section in the custom resource.

Call home can be enabled, modified, or disabled at any time. Enter the **oc explain callhome** command to view more details.

Callhome status

Status	Description
Enabled	**Callhome** is enabled.
Success	**Callhome** is configured and is able to communicate with the IBM Callhome server.

Callhome spec

License agreement

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

```
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**.

> **Note:** The **countryCode** is a two-letter upper case 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:

```
kind: Callhome
...
spec:
  ...
  proxy:
    host: proxyserver.example.com
    port: 443
    secretName: proxyServerSecret
```

Filesystems

Remote filesystem

To configure a remote mounted file system for the container native cluster, you must create a **Filesystem** custom resource and a **RemoteCluster** custom resource.

Filesystem

Filesystem status

Status	Description
Success	The file system is created or mounted.

Filesystem spec

The sample **Filesystem** custom resource can be found under **kind: Filesystem** in **scale_v1beta1_cluster_cr.yaml** file. For more information, see Configuring the IBM Spectrum Scale container native cluster custom resources.

The name that you set for the Filesystem CR under **metadata.name** is the name of the custom resource and also becomes the name of the remote file system mount point at **/mnt/<metadata.name>**. In the sample, the name of the local file system is **remote-sample** and is mounted at **/mnt/remote-sample**. You can define more than 1 Filesystem CR.

Set the details under the **remote** section to reflect the storage cluster file system being mounted as **fs** and the name of the **RemoteCluster** created as **cluster**.

```
kind: Filesystem
metadata:
  ...
  name: remote-sample
spec:
  remote:
    cluster: remotecluster-sample
    fs: fs1
```

Field	Required	Purpose

Field	Required	Purpose
`spec.remote.fs`	yes	`fs` is the name of the file system on the `RemoteCluster` to mount.
`spec.remote.cluster`	yes	Cluster is the name of the `RemoteCluster` resource.

Limitations: Deleting a `Filesystem` custom resource **does not** un-mount or delete the file system configuration from the IBM Spectrum Scale Cluster.

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

RemoteCluster

The sample `RemoteCluster` custom resource can be found under `kind: RemoteCluster` in `scale_v1beta1_cluster_cr.yaml` file.

RemoteCluster status

Status	Description
Ready	If the remotecluster credentials are established.

RemoteCluster spec

The name that you set for the `RemoteCluster` CR under `metadata.name` identifies the remote storage cluster you want to create an authentication to. This name is used as a reference in the `Filesystem` CR `spec.remote.cluster` to identify the remote storage cluster serving the file system. You can define more than 1 RemoteCluster.

To create RemoteCluster Spec, complete the following steps:

1. Validate that the secret for the storage cluster is created. For more information, see Creating secrets for storage cluster GUI.
2. Set the GUI details to match your remote storage GUI in the `spec.gui` section:

```
kind: RemoteCluster
...
metadata:
name: remotecluster-sample
spec:
contactNodes:
- storagecluster1node1
- storagecluster1node2
gui:
    cacert: cacert-storage-cluster-1
    host: guihost.example.com
    insecureSkipVerify: false
    secretName: cnsa-remote-mount-storage-cluster-1
```

Field	Required	Purpose
`metadata.name`	yes	The name of the CR, that is used to identify the remote storage cluster in the `filesystem` CR.
`spec.contactNodes`	no	The list of storage nodes names used as the contact nodes list. The names must be the **daemon node name** in the IBM Spectrum Scale storage cluster. If nothing is specified, the operator uses 3 nodes obtained from the storage cluster. **Note**: To list the contact nodes, enter the `mmlscluster` command on the storage cluster and use the listed daemon node names.

Field	Required	Purpose
`spec.gui.host`	yes	The hostname for the GUI endpoint on the storage cluster.
`spec.gui.cacert`	no	The name of the Kubernetes ConfigMap containing the CA certificate for the storage cluster GUI.
`spec.gui.secretName`	yes	The name of the kubernetes secret created during the storage cluster configuration.
`spec.gui.insecureSkipVerify`	no	The parameter controls whether a client verifies the storage cluster's GUI certificate chain and host name. If set to `true`, TLS is susceptible to machine-in-the-middle attacks. The default value is `false`.

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

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

EncryptionConfig

To access encrypted data from a remote mounted file system the encryption key should be present in the GPFS cluster. IBM Security Guardium Key Lifecycle Manager provides a centralized and automated key management solution to protect keys that are used for encryption. To access the encryption key from the key-server, the key-client must be configured in the cluster.

To configure key-server and key-client, you need to create an `EncryptionConfig` custom resource.

EncryptionConfig status

Status	Description
Success	If `EncryptionConfig` is successfully configured.

EncryptionConfig spec

The sample `EncryptionConfig` custom resource file can be downloaded by entering the following command:

```
curl https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-
native/v5.1.2.1/generated/scale_v1beta1_encryptionconfig_sample.yaml \
> scale_v1beta1_encryptionconfig_sample.yaml
```

This file is used to set configuration. When deployed, it initiates the configuration of key-server in the cluster.

Here, `metadata.name` is the name of the CR with the configuration parameters. Under `spec`:

- `server` is the hostname of the key-server in which key is stored.
- `tenant` is the name of the tenant that contains the encryption keys. This has to be the same tenant name that is used to store the encryption keys of the remote storage file system.
- `remoteRKM` is the RKM ID that is used in the storage cluster to register the client with the tenant. In case the key-server uses a CA signed server certificate, `cacert` is used to pass the CA certificates to the client for validation of the endpoint certificate.

```
apiVersion: scale.spectrum.ibm.com/v1beta1
kind: EncryptionConfig
metadata:
  ...
```

```
  name: encryption-config-sample
  ...
spec:
  cacert: sample-ca-cert
  client: sampleClient
  port: 9443
  remoteRKM: sampleRKM
  secret: keyserver-credentials
  server: keyserver.example.com
  tenant: sampleTenant
```

Additional parameters under **spec.filesystems** is used to encrypt an unencrypted local file system. If not required, then this spec must be deleted or commented.

Required fields for accessing the encrypted data from an encrypted remote storage file system are listed in the following table.

Field	Required	Purpose
metadata.name	yes	The name of the CR.
spec.server	yes	The key server name to configure for encryption.
spec.tenant	yes	The default tenant name to the key server. This name can consist of any alphanumeric characters and non-alphanumeric characters, such as '_'.
spec.secret	yes	The name of the basic-auth secret containing the username and password for the key server.
spec.client	yes	The key client that communicates with the key Server.
spec.remoteRKM	yes	The RKM ID from the remote cluster corresponding to the given key server and tenant.
spec.port	no	The port can be used to provide a non-default port for key server.
spec.cacert	no	The name of the ConfigMap storing CA and endpoint certificates to be used while adding/renewing key server certificate chain.
spec.filesystems	no	The list of file systems to be encrypted.
spec.filesystems.name	no	The name of the file system.
spec.filesystems.algorithm	no	The algorithm to be used for encryption. Valid values are “DEFAULTNISTSP800131AFAST” and “DEFAULTNISTSP800131A”.

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

Creating the IBM Spectrum® Scale container native cluster

Deploy the cluster by applying the custom resource modified in the *Configuring the IBM Spectrum Scale container native cluster custom resources* procedure. For more information, see Configuring the IBM Spectrum Scale container native cluster custom resources.

Once the custom resources file is applied, IBM Spectrum Scale Operator creates all the pods that make up an IBM Spectrum Scale container native cluster. Enter the following command to apply the YAML file:

```
oc apply -f ./scale_v1beta1_cluster_cr.yaml
```

Creating secrets for storage cluster GUI

Create a secret on the Red Hat® OpenShift® cluster to store the **username** and **password** for the IBM Spectrum Scale Storage cluster GUI user and password.

This secret is used by the Operator to communicate with the storage cluster while configuring for remote mount.

Two new secrets must be added for each storage cluster being configured for remote mount on the IBM Spectrum Scale container native cluster.

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

   The username and password specified in this topic must match the GUI user that was created on the storage cluster in the *Creating Operator User and Group* procedure. For more information, see Creating Operator User and Group.

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

   > **Note:** The name of this secret must match the **secretName** field defined for the RemoteCluster CR. For more information, see Filesystems.

   ```
   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 storage cluster **CsiAdmin** GUI user.

   The username and password specified in this topic must match the GUI user that was created on the storage cluster of the *Container Storage Interface (CSI) configuration* procedure. For more information, see Container Storage Interface (CSI) configuration.

   > **Note:** The name of this secret should match the **csiSecretName** field defined for the RemoteCluster CR. For more information, see Filesystems.

3. To create the storage cluster GUI user 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
   ```

4. To 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
   ```

Configuring Certificate Authority (CA) certificates for storage cluster

IBM Spectrum® 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 Spectrum 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 Spectrum 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
```

> **Note**: 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 Spectrum 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 Filesystem Custom Resource. For more information, see Filesystems.

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 Filesystem Custom Resource. For more information, see Filesystems.

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 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
```

Verifying the IBM Spectrum Scale container native cluster

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

Complete the following steps:

Note: For more information, see Debugging IBM Spectrum 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
   ```

   Note: The resulting cluster contains two **gui** pods, two **pmcollector** pods, and one core pod per node that matches the specified **nodeSelector**.

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

   - 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
     ```

     Output:

     ```
     GPFS cluster information
     ========================
     GPFS cluster name:          ibm-spectrum-scale.ocp4.example.com
     GPFS cluster id:            835278197609441888
     GPFS UID domain:            ibm-spectrum-scale.ocp4.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   worker0          172.29.0.145  worker0          quorum-manager-
     perfmon
        2   worker1          172.29.0.146  worker1          quorum-manager-
     perfmon
     ```

```
    3   worker3           172.29.0.148  worker3          quorum-manager-
perfmon
```

- 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
```

Example output:

```
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. Check the status of the **RemoteCluster** to ensure that the Status **Ready** is **True**.

- Get a list of the remote clusters.

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

- Describe the remote cluster to check its status.

```
oc describe remotecluster remotecluster-sample -n ibm-spectrum-scale
```

b. If the storage cluster authentication was created successfully, you should see a **Status** similar to the sample shown:

```
# oc get remoteclusters remotecluster-sample -n ibm-spectrum-scale
...
...
Status:
Conditions:
   ...
   Message:               The remote cluster has been configured successfully.
   Reason:                AuthCreated
   Status:                True
   Type:                  Ready
Events:
Type     Reason            Age                   From           Message
----     ------            ----                  ----           -------
...
...
Normal   Created           66s                   RemoteCluster  The remote
cluster has been configured successfully.
```

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

a. Check the status of the **Filesystem** to ensure that the Status **Success** is **True**.

- Get a list of the file systems:

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

- Describe the sample file systems to check status:

```
oc describe filesystems remote-sample -n ibm-spectrum-scale
```

b. If the file system was created successfully, you should see a **Status** similar to the sample shown:

```
Status:
Conditions:
   ...
   Message:               The remote filesystem has been created and mounted.
   Reason:                FilesystemEstablished
   Status:                True
   Type:                  Success
Events:
Type     Reason   Age                  From         Message
----     ------   ----                 ----         -------
Warning  Failed   16m (x20 over 28m)   Filesystem   Unable to register storage
cluster on client cluster.
Warning  Failed   15m                  Filesystem   Unable to create remote
filesystem on client cluster.
Normal   Created  14m (x4 over 15m)    Filesystem   Attempting to mount
filesystem on: [worker0 worker1]
Normal   Created  14m                  Filesystem   Attempting to mount
filesystem on: [worker1 worker0]
Normal   Created  13m (x2 over 13m)    Filesystem   Attempting to mount
filesystem on: [worker0]
Normal   Created  8m8s                 Filesystem   The filesystem has been
created and mounted.
```

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                    ibm-spectrum-scale.ocp4-
c4.example.com
172.29.0.146    worker1                    ibm-spectrum-scale.ocp4-
c4.example.com
172.29.0.145    worker0                    ibm-spectrum-scale.ocp4-
c4.example.com
```

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

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 Spectrum 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 -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 -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.

Note:

- The *Events* disappear after they are created.
- The *Status* and *Events* listed above are examples and they look different on your system.

Upgrade IBM Spectrum Scale container native

The following information defines the upgrade steps from IBM Spectrum Scale container native 5.1.1.4 to 5.1.2.1.

Complete the following steps:

1. Set the operator deployment replicas to 0.

   ```
   oc patch deploy ibm-spectrum-scale-controller-manager \
   --type='json' -n ibm-spectrum-scale-operator \
   -p='[{"op": "replace", "path": "/spec/replicas", "value": 0}]'
   ```

2. Ensure the operator does **not** exist.

```
oc get pods -n ibm-spectrum-scale-operator \
-l app.kubernetes.io/instance=ibm-spectrum-scale \
-l app.kubernetes.io/name=operator
```

3. Delete the old security context constraint.

```
oc delete scc ibm-spectrum-scale-privileged
```

4. Apply the new manifests (excluding operator).

```
oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/installer/ibm-spectrum-scale-excluding-operator.yaml
```

5. Patch the daemon CR with `edition` from cluster CR.

```
oc patch daemon ibm-spectrum-scale \
--type='json' -n ibm-spectrum-scale \
-p="[{'op': 'replace', 'path': '/spec/edition', 'value': $(oc get cluster ibm-spectrum-scale -ojsonpath='{.spec.license.license}')}]"
```

6. Apply the new manifests (including operator).

```
oc apply -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-container-native/v5.1.2.1/generated/installer/ibm-spectrum-scale-operator.yaml
```

Configuring IBM Spectrum® Scale Container Storage Interface (CSI) driver

Use the following sections to help with deploying IBM Spectrum Scale CSI with IBM Spectrum Scale container native:

- Configuring storage class to use CSI driver
- Managed CSI fields
- Setting primary file set

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.

Note: Adjust the parameters as per 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

oc create -f storageClass_Lightweight.yaml
```

Fileset based volumes

A Storage class example for creating fileset based volumes is provided.

Note: Adjust the parameters as per 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
    clusterId: "17797813605352210071" # cluster ID of storage cluster
reclaimPolicy: Delete
```

A sample fileset based Storage Class is created using the primary file system as the **volBackendFs**. It can be used to create other storage classes with the remote cluster ID that is provided. Enter the **oc get storageclass -oyaml > storageClass_fileset.yaml** command to create a copy of this storage class. Then configure parameters as desired and create the configured storage class using the command below:

```
oc create -f storageClass_fileset.yaml
```

Note: For more information, see Storage Class in IBM Spectrum Scale CSI documentation.

Managed CSI fields

In the CSI Custom Resource (CR) that is created by the CSI Controller, there are some fields that are managed by the controller. If these fields are changed, they are overridden by the controller. If required, you can change any field that is not managed by the controller.

Managed fields

Note: The following fields are populated with default values by the CSI Controller. Any new values are honored, however, any values that are manually removed are repopulated upon the next controller reconcile cycle.

Field	Default Value(s)
clusters	Two entries are created by default (local and remote clusters).
clusters.id	Local Cluster ID / Cluster ID of Remote cluster.
clusters.secrets	**ibm-spectrum-scale-gui-csiadmin**
clusters.secureSSLMode	**false**
clusters.primary.primaryFs	The name of the first file system created (only applicable in local. cluster entry)

Field	Default Value(s)
clusters.restApi.guiHost	`ibm-spectrum-scale.<container-native-namespace>` for local cluster entry and the `host` 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 fill the desired field:

```
oc edit csiscaleoperator -n ibm-spectrum-scale-csi
```

Setting primary file set

After the CSI CR is created by the CSI controller the primary file set needs to be set in order to avoid the naming conflict. Once this field is added the CSI driver pods are deleted and recreated one by one.

Enter the `oc edit csiscaleoperator -n ibm-spectrum-scale-csi` command and add the `primaryFset` field:

```
clusters:
  - id: "11171289193543683780"
    secrets: "secret-cnsa"
    secureSslMode: false
    primary:
      primaryFs: "fs5"
      primaryFset: "cluster1-fset" #<---- example
      remoteCluster: "2303539379337927879"
    restApi:
      - guiHost: "ibm-spectrum-scale-gui.ibm-spectrum-scale"

  - id: "2303539379337927879"
    secrets: "secret-storage"
    restApi:
    - guiHost: "koopa-gui-1.fyre.ibm.com"
```

Using IBM Spectrum Scale GUI

You can refer to the mapping of OpenShift users to IBM Spectrum Scale GUI user groups for accessing the IBM Spectrum Scale GUI.

- IBM Spectrum Scale container native GUI

IBM Spectrum Scale container native GUI

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

OpenShift Users

All OpenShift users are mapped to two IBM Spectrum Scale GUI user groups. Details are provided in the following table:

	Roles			**Privileges**	
OCP role	GUI role	View	Download snap*	Manage events**	Test connection for call home
Cluster admin	Maintenance	Yes	Yes	Yes	Yes
Kubeadmin	Maintenance	Yes	Yes	Yes	Yes
View	Monitor	Yes	Yes	No	Yes

*Ability to download master and non-master snaps.

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

Accessing the IBM Spectrum Scale GUI

Users created on the Openshift Container Platform (OCP) can log in to the IBM Spectrum Scale container native GUI through single-sign-on (SSO) using the OAuth implementation.

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

1. In a browser, navigate to **`https://ibm-spectrum-scale-gui-ibm-spectrum-scale.apps.<ocp domain>/`**, where **`<ocp domain>`** is the domain of your OpenShift cluster. You should see the IBM Spectrum Scale GUI login page.

   If the 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`**, which redirects to the **`Red Hat Openshift Container Platform`** login page.

3. Authenticate using your OCP user credentials.
   On success, you are redirected back to the IBM Spectrum Scale GUI homepage.

Maintenance for a deployed cluster

The maintenance of a deployed IBM Spectrum® Scale container native cluster includes certain procedures.

- Shutting down a cluster
- Upgrading Red Hat OpenShift Container Platform
- Starting the cluster after shutdown
- Adding a new node to an existing cluster

Shutting down a cluster

Before you begin the maintenance procedure, the IBM Spectrum Scale container native cluster must be shut down to avoid any issues.

> **Note:** For more information, see On the nodes running CSI sidecars in IBM Spectrum Scale CSI documentation.

Complete the following steps to shut down a cluster:

1. Enter the following command to scale the number of IBM Spectrum Scale container native operators to 0.

   ```
   oc edit deploy -n ibm-spectrum-scale-operator
   ```

   Set number of replicas to 0:

   ```
   ...
   spec:
     progressDeadlineSeconds: 600
     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
   ```

Upgrading Red Hat OpenShift Container Platform

Preparation

In preparation of Red Hat® OpenShift® Container Platform upgrade, you must stop all applications that use the IBM Spectrum® Scale file system to ensure that there is no workload running. Once the workload is stopped, proceed with Red Hat OpenShift Container Platform upgrade.

For more information, see Updating a cluster within a minor version from the web console in Red Hat OpenShift documentation.

During the upgrade process, the Machine Config Operator (MCO) applies the new configuration to your cluster machines. It cordons the number of nodes that is specified by the maxUnavailable field on the machine configuration pool and marks them as unavailable. By default, this value is set to 1. It then applies the new configuration and reboots the machine.

As the worker nodes are updated by Red Hat OpenShift Container Platform, the IBM Spectrum Scale pods are restarted and returned to a schedulable state.

> **Note:** If you encounter any problems during OpenShift upgrade, open a support case by gathering cluster information. The details help in providing debugging information to Red Hat Support on your cluster. For more information, see Gathering data about your cluster in Red Hat OpenShift documentation.

Upgrading Red Hat OpenShift from 4.5 to 4.7, 4.8, or 4.9

When upgrading to OpenShift Container Platform 4.7, 4.8, or 4.9 some additional steps must be performed to properly configure the worker nodes through Machine Config Operator.

Complete the following steps:

1. Ensure that you have properly shut down the IBM Spectrum Scale cluster. For more information, see Shutting down a cluster.
2. Configure the worker nodes. For more information, see Red Hat OpenShift Container Platform configuration.

   Note: If Machine Configurations are applied to your Red Hat OpenShift Container Platform cluster in a previous IBM Spectrum Scale container native deployment, do not reapply the same Machine Configuration again. Reapplying an existing Machine Configuration could cause reboots across the worker nodes.

3. Start the IBM Spectrum Scale cluster by following the steps for starting the cluster. For more information, see Starting the cluster after shutdown.

Upgrading Red Hat OpenShift from 4.6 to 4.7, 4.8, or 4.9

Complete the following steps:

1. Ensure that the workload is stopped. For more information, see Preparation.
2. Proceed with Red Hat OpenShift Container Platform 4.7, 4.8, or 4.9 upgrade procedures. For more information, see Upgrading Red Hat OpenShift from 4.5 to 4.7, 4.8, or 4.9.

   Note: Cluster can be upgraded from all releases of Red Hat OpenShift 4.6 or 4.7 to 4.8 or 4.9. If initial installation of IBM Spectrum Scale container native is on any version of Red Hat OpenShift 4.6, and you are upgrading to 4.7 or later, the required Machine Config files need to be applied to the Red Hat OpenShift cluster prior to deployment. It allows the Machine Config Operator (MCO) to install the required kernel support packages on the nodes selected to run core pods.

Starting the cluster after shutdown

If the IBM Spectrum® Scale cluster was shut down, start the cluster using the following steps:

Note: Ensure that the worker nodes are in Ready state before restarting the IBM Spectrum Scale cluster by entering the **`oc get nodes`** command. If any of the worker nodes are in a state other than Ready, the IBM Spectrum Scale cluster fails to restore.

1. Scale the number of operator pods back to 1.

   ```
   oc edit deploy -n ibm-spectrum-scale-operator
   ```

   Set number of replicas to 1:

   ```
   ...
   spec:
     progressDeadlineSeconds: 600
     replicas: 1
   ...
   ```

After the operator pod comes back up, the core pods are rescheduled and the default CSI label is re-applied.

Adding a new node to an existing cluster

To add a new node, you need to add it to an existing cluster and configure CSI on it.

When a new node with labels that match the existing cluster's node selector is added, a pod is created on the new node. The new pod is up and running within a few minutes. For more information, see Labels and Annotations.

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 <scale-pod> -n ibm-spectrum-scale -- 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 is ready to be enabled on this node.

Configuring CSI on new nodes

Note: CSI must only be configured on new nodes after they are finished arbitrating and in Active state. Applying the CSI node label before nodes are in an active state can cause unexpected behavior.

For CSI to recognize the newly added node, apply the label to the node:

```
oc label node <node-name> scale=true
```

The newly added node can now be used for running applications.

Cleaning up the container native cluster

To safely remove the pods or perform other maintenance actions, IBM Spectrum® Scale container native cluster needs to be manually shut down prior to performing these operations. The following procedures outline the steps to complete these actions and validate that it is safe to shut down the cluster.

- Deleting a cluster
- Removing applications
- Custom Resource
  - Filesystems
  - RemoteCluster

- Cleaning up IBM Spectrum Scale operator
- Cleaning up the worker nodes
- Cleaning up on the storage cluster

Deleting a cluster

When deleting the entire cluster, all applications and IBM Spectrum® Scale Container Storage Interface driver must be unloaded prior to the unmount and shutdown steps.

Removing applications

Complete the following steps:

Note: Ensure that you are in the project for IBM Spectrum® Scale Container Storage Interface (CSI) driver.

1. Enter the following command to query the PVC to identify the applications that are active.

   ```
   oc describe <csi pvc>
   ```

2. Enter the following command to remove all the applications. This requires the node to be drained of all data.

   ```
   oc delete <application deployment or daemonSet from csi pvc describe output>
   ```

Custom Resource

There can be situations when you need to change the custom resource definitions but not clean up the whole container native cluster. The following sections describe how to clean up the IBM Spectrum Scale artifacts when only deleting custom resource definitions.

- Filesystems
- RemoteCluster

Filesystems

Deleting a **`Filesystem`** custom resource **does not** result in the operator un-mounting or deleting the remote mount file system configuration on the IBM Spectrum Scale container native cluster.

Before removing the configuration of the remote mounted file system, ensure that there are no applications actively writing to the file system.

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 OpenShift.

   ```
   oc delete filesystem remote-sample -n ibm-spectrum-scale
   ```

2. Log in to a core pod, using the following command, to remove the file system from IBM Spectrum 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. If the remote storage cluster is only configured to mount and serve the single **remote-sample** file system, you can delete the remote cluster definition. Otherwise, the other file system(s) needs to be deleted by using the same process mentioned in the above step.

   - Find the remote clusters.

     ```
     mmremotecluster show all
     ```

   - Delete the remote cluster serving 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 Spectrum 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 additional file systems are using this credential.

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.

Cleaning up IBM Spectrum Scale operator

Complete the following steps:

1. Enter the following command to delete the IBM Spectrum Scale Custom Resources.

```
oc delete -f scale_v1beta1_cluster_cr.yaml -n ibm-spectrum-scale
```

2. Enter the following command to uninstall the Operator, related objects, and namespaces.

```
oc delete -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-
container-native/v5.1.2.1/generated/installer/ibm-spectrum-scale-operator.yaml
```

3. Enter the following command to clean up the performance monitoring and IBM Spectrum® 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 Spectrum® 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
```

   Note: If the current namespace was deleted in these steps (**ibm-spectrum-scale**, **ibm-spectrum-scale-operator**, **ibm-spectrum-scale-csi**) then the working namespace should be changed to an existing one.

```
oc project default
```

Cleaning up the worker nodes

IBM Spectrum® Scale requires host path volume mounts and creates directories on each worker node.

Note: At this point, the project is deleted. Ensure that you are in the default namespace by entering **oc project default** command.

Complete the following steps:

1. Enter the following command to list the nodes that have the **node-role.kubernetes.io/worker=** label.

```
oc get nodes -l 'node-role.kubernetes.io/worker=' -o jsonpath="{range
.items[*]}{.metadata.name}{'\n'}"
```

2. For each of the listed worker nodes, enter the following command to create a debug pod that removes the host path volume mounted directories used by IBM Spectrum Scale:

```
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 ...
```

3. Ensure that none of the files are left by entering the following command:

```
oc debug node/<openshift_worker_node> -T -- chroot /host sh -c "ls /var/mmfs;
ls /var/adm/ras"
```

Example:

```
oc debug node/worker0.example.com -T -- chroot /host sh -c "ls /var/mmfs; ls
/var/adm/ras"
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
Removing debug pod ...
error: non-zero exit code from debug container
```

4. Remove node labels created by IBM Spectrum Scale container native operator:

```
oc label node --all scale.spectrum.ibm.com/role-
oc label node --all scale.spectrum.ibm.com/designation-
oc label node --all scale-
```

Cleaning up on the storage cluster

Delete the access permission that is granted to the IBM Spectrum® Scale client cluster for mounting the remote file system.

Perform the following steps on the IBM Spectrum Scale storage cluster:

1. Enter the following command to query the name of the containerized client cluster:

```
$ mmauth show all | grep ibm-spectrum-scale
Cluster name: ibm-spectrum-scale.clustername.example.com
```

2. Enter the following command to remove the client cluster authorization:

```
$ mmauth delete ibm-spectrum-scale.clustername.example.com
mmauth: Propagating the cluster configuration data to all affected nodes.
mmauth: Command successfully completed
```

Monitoring

The IBM Spectrum® 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 Spectrum 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.

Note: 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 Spectrum Scale bridge for Grafana

IBM Spectrum 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 Spectrum Scale performance monitoring bridge for Grafana deployed on your IBM Spectrum Scale container native cluster. For more information, see IBM Spectrum Scale bridge for grafana repository in GitHub.

The IBM Spectrum Scale bridge for Grafana is an open source tool, available for free usage on IBM Spectrum Scale devices. It translates the IBM Spectrum Scale metadata and performance data collected by the IBM Spectrum Scale performance monitoring tool to query requests acceptable by the Grafana-integrated openTSDB plugin.

Starting with the IBM Spectrum Scale container native 5.1.2.1, the IBM Spectrum Scale performance monitoring bridge for Grafana could be deployed automatically through the operator. For more information, see Configuring the IBM Spectrum Scale container native cluster custom resources.

For more information about setting up a Grafana instance for monitoring the IBM Spectrum Scale container native cluster, see Setup Grafana for monitoring a IBM Spectrum Scale container native cluster in a k8s OCP environment in GitHub documentation.

Troubleshooting

Use the following sections to help troubleshoot and debug specific issues with IBM Spectrum® Scale container native deployment.

- Debugging the IBM Spectrum Scale operator
- Debugging IBM Spectrum Scale deployment
- Debugging IBM Spectrum Scale Container Storage Interface (CSI) deployment
- Debugging OCP upgrade
- Common issues
- Known issues
- Collecting data for support

Debugging the IBM Spectrum Scale operator

Problem: The operator pod is not successfully deployed

No operator pod appears when running **`oc get pods -n ibm-spectrum-scale-operator`**.

- Verify that all worker nodes in the Red Hat® OpenShift® Container Platform cluster are in a **`Ready`** state. If not, the operator pod may not have an eligible node to be deployed to.

```
# 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 may point to any problems.

```
oc get deployment -n ibm-spectrum-scale-operator
oc describe deployment ibm-spectrum-scale-controller-manager -n ibm-spectrum-scale-operator
```

```
oc get replicasets -n ibm-spectrum-scale-operator
oc describe replicaset <replicaset name> -n ibm-spectrum-scale-operator
```

Problem: Operator pod shows container restarts

- Kubernetes keeps the logs of the current container and the previous container. Take a look at the previous container's logs for any clues using the following command:

```
oc logs -p <operator pod> -n ibm-spectrum-scale-operator
```

Debugging IBM Spectrum Scale deployment

Problem: 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
```

Problem: 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
  ```

Problem: Core, GUI, or collector pods show container restarts

- Kubernetes keeps the logs of the current container and the previous container. Take a look at the previous container's logs for any clues using the following command:

  ```
  oc logs -p <scale pod> -n ibm-spectrum-scale
  ```

Problem: Core pods are stuck in Init:1/2

If for some reason, the IBM Spectrum® Scale container native cluster fails to create, the core pods on the worker nodes get stuck in 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. For more information about clean up, see Cleaning up IBM Spectrum Scale operator and Cleaning up the worker nodes. For more information about redeploy, see Installing the IBM Spectrum Scale container native operator and cluster.

Problem: All pods have been deployed but GPFS cluster is stuck in the "arbitrating" state

If the cluster is stuck in 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
```

Problem: 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 remoteclusters -n ibm-spectrum-scale
      oc describe remotecluster <name> -n ibm-spectrum-scale
      ```

    - Filesystems

      ```
      oc get filesystems -n ibm-spectrum-scale
      oc describe filesystem <name> -n ibm-spectrum-scale
      ```

    Check the **Status** and **Events** for any reason of failures.

    If nothing, check the operator logs for any errors:

    ```
    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 Spectrum Scale documentation.

Debugging IBM Spectrum® Scale Container Storage Interface (CSI) deployment

Problem: CSI pods stuck in CrashLoopBackOff (Unauthorized GET request)

```
# oc get pods
NAME                                                READY   STATUS
RESTARTS   AGE
ibm-spectrum-scale-csi-9566l                        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

# 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 recreate 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
```

Problem: CSI CR is never created

If all the core pods are running and the IBM Spectrum Scale container native cluster appears to be in a good state the CSI CR should be created automatically. In some error paths this does not happen and causes the driver pods to not be scheduled:

Note: Only the operator pod is listed and no results are found for csiscaleoperators

```
# 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
```

- Check that the GUI pod(s) 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 non-empty cluster ID.

```
# oc describe daemon -n ibm-spectrum-scale
```

Find the status section and ensure 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 Spectrum Scale container native cluster is experiencing an issue. Check the operator logs for more information.

Debugging OCP upgrade

Problem: GUI mount not getting refreshed as multiple OCP cluster are remote mounted on same FS

To resolve the issue, unmount the FS from another OCP cluster.

```
# /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
```

Common issues

Error: daemon and kernel extension do not match

This error occurs when there is an unintentional upgrade of GPFS code.

The issue presents itself as the GPFS state is down and the above error is found in the GPFS logs.

To resolve the issue, follow proper upgrade procedures. The issue occurs because the kernel module cannot be unloaded when a file system is in use. Rebooting the node resolves the problem, or follow procedures to remove application workloads and then enter the following command on the node issue:

```
rmmod tracedev mmfs26 mmfslinux
```

Note: For more information, see Removing applications.

RestError: Failed to get storage cluster information. errmsg: 401 Unauthorized GET

The **oc describe gpfs** command shows the following error:

```
  Warning  RestError                48s (x12 over 2m2s)  RemoteMount <filesystem>:
[storage cluster] Failed to get storage cluster information. errmsg: 401
Unauthorized GET request to https://<storage cluster GUI>:443/scalemgmt/v2/cluster
```

The IBM Spectrum® Scale GUI REST credentials for storage clusters are stored in kubernetes secrets. For more information, see IBM Spectrum Scale storage cluster configuration. The **RestError** indicates that the GUI user in the kubernetes secret does not match the GUI user in the storage cluster.

There are different 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 Operator User and Group.
- 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 -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 Spectrum Scale GUI REST API.

   ```
   oc get secret cnsa-remote-mount-storage-cluster-1 -n ibm-spectrum-scale -
   ojsonpath='{.data.username}' | base64 -d -
   ```

```
 oc get secret cnsa-remote-mount-storage-cluster-1 -n ibm-spectrum-scale -
ojsonpath='{.data.password}' | base64 -d -
```

Note: In some shells, the end of the line has a highlighted `%`. This denotes 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 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 Spectrum 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
```

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
```

The pod create times show that the ssh key secret was created after the deployment. This means that the deployment rightfully could not find the secret to mount, as it did not yet exist.

This message can be misleading as the pods should resolve themselves once the secret is created. If core pods are not in a **Running** state, and the secret is already created, deleting the ibm-spectrum-scale-core pods should resolve the issue. This restarts the pods and allow the mount to complete successfully for the already created SSH key.

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.

Failed to establish RemoteScale connector when cacert ConfigMap doesn't exist

```
# oc describe gpfs
...
Normal   RemoteMountAttempt            0s (x11 over 11s)      RemoteMount  fs1,
Attempting to configure remote storage file system=fs1 from
remoteCluster=storageCluster1 as /mnt/fs1.
Warning  RemoteMountRestError          0s (x11 over 11s)      RemoteMount  fs1,
[storage cluster] Failed to establish RemoteScale connector. 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, see Configuring Certificate Authority (CA) certificates for storage cluster.

To resolve this issue, choose a configuration option from *Configuring certificate authority (CA) certificates for storage cluster* procedure and follow the instructions below for the corresponding option of choice. For more information, see Configuring Certificate Authority (CA) certificates for storage cluster.

- Option 1

  Create the **cacert-storage-cluster-1** ConfigMap. For more information, see Configuring Certificate Authority (CA) certificates for storage cluster.

- Option 2

  Ensure that the storage cluster GUI is using a default trusted CA certificate. If configured correctly, the storage cluster GUI should connect successfully.

- Option 3

  Patch the Custom Resource to use **insecureSkipVerify: true**.

```
  oc patch scaleclusters ibm-spectrum-scale --type='json' -n ibm-spectrum-scale
\
  -p='[{"op": "replace", "path":
"/spec/remoteClusters/0/gui/insecureSkipVerify", "value": true}]'
```

Known issues

pmsensors showing 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 killed and restarted manually on all scale-core pods, to get the performance metrics collection resumed.

To kill 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, there is no process running, and you do not need to enter the following command to kill 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
```

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, execute 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
```

File system fails to mount because it is already mounted on a IBM Spectrum Scale container native cluster

If a file system is failing to mount to the container native cluster ensure that this is not caused by the single cluster limitation:

The same remote file system cannot be mounted on multiple IBM Spectrum Scale container native clusters.

Collecting data for support

You need to perform the following procedures to gather data for support:

- Generating GPFS trace reports
- Configuring GPFS trace reports from cluster creation
- Kernel crash dumps
- Gather data about the IBM Spectrum Scale container native cluster
- Gather data about the Red Hat OpenShift Container Platform cluster

Generating GPFS trace reports

Some issues might require low-level system detail accessible only through the IBM Spectrum Scale daemon and the IBM Spectrum 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 Spectrum Scale documentation.

Note: 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>
   ```

   Note: 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 Spectrum Scale installs. For more information, see Generating GPFS trace reports in IBM Spectrum 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
   ```

Configuring GPFS trace reports from cluster creation

In some situations, it may be required to configure GPFS tracing from cluster creation. This can be accomplished using the cluster core profile and settings directed by IBM Support Center.

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.

Gather data about the IBM Spectrum Scale container native cluster

To gather logs and diagnostic data to assist IBM Support in debugging an issue, enter the **`oc adm must-gather`** CLI command with the supporting must-gather image specifically for IBM Spectrum Scale container native.

The **`ibm-spectrum-scale-must-gather`** image collects the Kubernetes objects associated with its namespace and also retrieve a GPFS snap from the IBM Spectrum Scale container native cluster.

Prerequisites

- Running **`oc adm must-gather`** requires the user to be logged in to an account on the Red Hat OpenShift Container Platform cluster that has sufficient privileges to query OpenShift and Kubernetes resources. Collaboration with the administrator may be needed to get necessary credentials for **`oc login -u <username>`** to successfully query OpenShift and Kubernetes resources.
- **`oc adm must-gather --image`** requires the must-gather image that is stored in a repository where it can be anonymously pulled (no credentials required). In order to satisfy this requirement, the must-

gather image must be pulled from the IBM Cloud Container Registry's entitled repository, and then uploaded to an image registry allowing anonymous pull. There are many methods for doing so, two of which are outlined.

1. If a production grade Docker V2 compatible registry is already configured for Red Hat OpenShift Container Platform cluster, and it can be set for anonymous pull of this image, then proceed with using this image registry.

2. If you do not have a pre-configured image registry, one possible temporary solution is to configure the Red Hat OpenShift Container Platform internal registry.

   Note: The Red Hat OpenShift Container Platform internal registry may default to an emptydir storage setup. Any images stored within may be deleted if the image registry restarts. For more information, see Image Registry in Red Hat OpenShift documentation. Once the image registry is configured, proceed with the instructions below.

Retrieve ibm-spectrum-scale-must-gather image from IBM Cloud Container Registry

The following instructions outline how to pull the must-gather image from the IBM Cloud Container Registry's entitled repository.

Note: `podman` 1.6+ is required to perform the following steps.

1. Log in to `cp.icr.io` and follow the prompts, using your IBM Cloud Container Registry ID and entitlement key.

```
podman login cp.icr.io
```

2. Pull the image.

```
podman pull cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-must-
gather@sha256:a80d3e9bab066595fdb9995e028898b19733d782a917b59c90b06dd05f1dad68
```

Upload ibm-spectrum-scale-must-gather to OpenShift Container Platform internal registry

Before performing the following steps, the Red Hat OpenShift Container Platform internal registry must be configured and ready for use.

Note: For more information, see Image Registry in Red Hat OpenShift documentation.

1. Log in as `kubeadmin`.

```
oc login -u kubeadmin
```

2. Add roles of registry-editor and edit to user `kube:admin`.

```
oc policy add-role-to-user registry-editor kube:admin
oc policy add-role-to-user edit kube:admin
```

3. Patch the configuration to expose the route to the OpenShift Container Platform internal registry.

```
oc patch configs.imageregistry.operator.openshift.io/cluster \
--patch '{"spec":{"defaultRoute":true}}' --type=merge
```

4. Set the following variables to facilitate with pushing images into the OpenShift Container Platform internal registry:

   - HOST - The name of the OpenShift Internal Container Registry

- NAMESPACE - The namespace of your IBM Spectrum Scale container native cluster, default is **ibm-spectrum-scale**
- IMAGE - The image name, **ibm-spectrum-scale-must-gather**
- TAG - The tag given for this release, **v5.1.2.1**

```
export HOST=$(oc get route default-route -n openshift-image-registry --
template='{{ .spec.host }}')
export NAMESPACE=ibm-spectrum-scale
export IMAGE=ibm-spectrum-scale-must-gather
export TAG=v5.1.2.1
```

5. Log in to the Red Hat OpenShift Container Platform integrated container registry through **podman**:

```
 oc whoami -t | podman login -u kubeadmin --password-stdin --tls-verify=false
$HOST
```

6. List your images in **podman** and find the Image ID associated with the ibm-spectrum-scale-must-gather image retrieved from the IBM Cloud Container Registry.

```
 # podman images
 REPOSITORY                                                    TAG     IMAGE ID
CREATED     SIZE
 cp.icr.io/cp/spectrum/scale/ibm-spectrum-scale-must-gather  <none>
663727979b51  3 days ago  374 MB
```

7. Assign a new image name to the ibm-spectrum-scale-must-gather image by using its Image ID.

```
 podman tag 663727979b51 $HOST/$NAMESPACE/$IMAGE:$TAG
```

Listing the images should now yield the image with the new name:

```
 # podman images
 REPOSITORY
TAG       IMAGE ID      CREATED     SIZE
 default-route-openshift-image-registry.example.com/ibm-spectrum-scale/ibm-
spectrum-scale-must-gather  v5.1.2.1  663727979b51  3 days ago  374 MB
```

8. Push the image into the Red Hat OpenShift Container Platform Image Registry.

```
 podman push $HOST/$NAMESPACE/$IMAGE:$TAG --tls-verify=false
```

9. Ensure that the ImageStream contains the ibm-spectrum-scale-must-gather image.

```
 oc get is $IMAGE -n ibm-spectrum-scale -oyaml | egrep
"name:|dockerImageRepository|tag"
```

Example:

```
 name: ibm-spectrum-scale-must-gather
 dockerImageRepository: image-registry.openshift-image-registry.svc:5000/ibm-
spectrum-scale/ibm-spectrum-scale-must-gather
 tags:
     tag: v5.1.2.1
```

Execute the ibm-spectrum-scale-must-gather image

Complete the following steps:

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=image-registry.openshift-image-
registry.svc:5000/ibm-spectrum-scale/ibm-spectrum-scale-must-gather:v5.1.2.1
```

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/
   ```

   Note: Replace the directory name used in this command with your respective must-gather directory.

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.

Note: Executing a default **must-gather** for OpenShift Container Platform debug does **not** collect information for IBM Spectrum Scale container native.

References

- IBM Spectrum Scale
- Red Hat OpenShift or Kubernetes

IBM Spectrum® 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 Spectrum Scale Container Storage Interface driver using CLI

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

Notices and Trademarks

The Notices and Trademarks of IBM Spectrum® Scale container native includes the following topics:

- Notices
- Trademarks
- Terms and conditions for product documentation
- IBM online privacy statement

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing Legal and Intellectual Property Law IBM Japan Ltd. 19-21, NihonbashiHakozakicho, Chuo-ku Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing IBM Corporation North Castle Drive, MD-NC119 Armonk, NY 10504-1785 US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. *enter the year or years*.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and trademark information at www.ibm.com/legal/copytrade.shtml.

Intel is a trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Red Hat®, OpenShift®, and Ansible® are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of the Open Group in the United States and other countries.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM online privacy statement

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, See IBM’s Privacy Policy and IBM’s Online Privacy Statement at http:// www.ibm.com/privacy/details the section entitled “Cookies, Web Beacons and Other Technologies” and the IBM Software Products and Software-as-a-Service Privacy Statement.

SC28-3168-14