IBM Spectrum Scale

Container Storage Interface Driver Guide Version 2.0.0

SC28-3113-02

Note

Before using this information and the product it supports, read the information in "Notices" on page 49.

This edition applies to version 5 release 0 modification 5 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)
- IBM Spectrum Scale Data Management Edition ordered through AAS (product numbers 5641-DM1, DM3, DM5)
- IBM Spectrum Scale Data Access Edition ordered through AAS (product numbers 5641-DA1, DA3, DA5)
- IBM Spectrum Scale Data Management Edition for IBM® ESS (product number 5765-DME)
- IBM Spectrum Scale Data Access Edition for IBM ESS (product number 5765-DAE)

Significant changes or additions to the text and illustrations are indicated by a vertical line (|) to the left of the change.

IBM welcomes your comments; see the topic "How to send your comments" on page xxii. When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Tables

About this information

This edition applies to IBM Spectrum Scale version 5.0.5 for AIX®, Linux®, and Windows.

IBM Spectrum Scale is a file management infrastructure, based on IBM General Parallel File System (GPFS) technology, which provides unmatched performance and reliability with scalable access to critical file data.

To find out which version of IBM Spectrum Scale is running on a particular AIX node, enter:

```
lslpp -l gpfs\*
```

To find out which version of IBM Spectrum Scale is running on a particular Linux node, enter:

```
rpm -qa | grep gpfs     (for SLES and Red Hat Enterprise Linux)
```

```
dpkg -l | grep gpfs     (for Ubuntu Linux)
```

To find out which version of IBM Spectrum Scale is running on a particular Windows node, open **Programs and Features** in the control panel. The IBM Spectrum Scale installed program name includes the version number.

Which IBM Spectrum Scale information unit provides the information you need?

The IBM Spectrum Scale library consists of the information units listed in Table 1 on page viii.

To use these information units effectively, you must be familiar with IBM Spectrum Scale and the AIX, Linux, or Windows operating system, or all of them, depending on which operating systems are in use at your installation. Where necessary, these information units provide some background information relating to AIX, Linux, or Windows. However, more commonly they refer to the appropriate operating system documentation.

Note: Throughout this documentation, the term "Linux" refers to all supported distributions of Linux, unless otherwise specified.

Table 1. IBM Spectrum Scale library information units

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	This guide provides the following information: **Product overview** • Overview of IBM Spectrum Scale • GPFS architecture • Protocols support overview: Integration of protocol access methods with GPFS • Active File Management • AFM-based Asynchronous Disaster Recovery (AFM DR) • Data protection and disaster recovery in IBM Spectrum Scale • Introduction to IBM Spectrum Scale GUI • IBM Spectrum Scale management API • Introduction to Cloud services • Introduction to file audit logging • Introduction to watch folder API • Introduction to clustered watch folder • IBM Spectrum Scale in an OpenStack cloud deployment • IBM Spectrum Scale product editions • IBM Spectrum Scale license designation • Capacity based licensing • IBM Spectrum Storage™ Suite • Understanding call home **Planning** • Planning for GPFS • Planning for protocols • Planning for Cloud services • Planning for AFM • Planning for AFM DR • Firewall recommendations • Considerations for GPFS applications • Security-Enhanced Linux support • Space requirements for call home data upload	System administrators, analysts, installers, planners, and programmers of IBM Spectrum Scale clusters who are very experienced with the operating systems on which each IBM Spectrum Scale cluster is based

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	**Installing** • Steps for establishing and starting your IBM Spectrum Scale cluster • Installing IBM Spectrum Scale on Linux nodes and deploying protocols • Installing IBM Spectrum Scale on AIX nodes • Installing IBM Spectrum Scale on Windows nodes • Installing Cloud services on IBM Spectrum Scale nodes • Installing and configuring IBM Spectrum Scale management API • Installation of Active File Management (AFM) • Installing and upgrading AFM-based Disaster Recovery • Installing call home • Installing file audit logging • Installing watch folder API • Installing clustered watch folder • Steps to permanently uninstall GPFS **Upgrading** • IBM Spectrum Scale supported upgrade paths • Upgrading to IBM Spectrum Scale 5.0.x from IBM Spectrum Scale 4.2.y or later • Upgrading to IBM Spectrum Scale 4.2.y from IBM Spectrum Scale 4.1.x • Online upgrade support for protocols and performance monitoring	System administrators, analysts, installers, planners, and programmers of IBM Spectrum Scale clusters who are very experienced with the operating systems on which each IBM Spectrum Scale cluster is based

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	• Upgrading IBM Spectrum® Scale non-protocol Linux nodes • Upgrading IBM Spectrum Scale protocol nodes • Upgrading AFM and AFM DR • Upgrading object packages • Upgrading SMB packages • Upgrading NFS packages • Upgrading call home • Manually upgrading the performance monitoring tool • Manually upgrading pmswift • Manually upgrading the IBM Spectrum Scale management GUI • Upgrading Cloud services • Upgrading to IBM Cloud Object Storage software level 3.7.2 and above • Upgrade paths and commands for file audit logging, watch folder API, and clustered watch folder • Upgrading with clustered watch folder enabled • Upgrading IBM Spectrum Scale components with the installation toolkit • Changing IBM Spectrum Scale product edition • Completing the upgrade to a new level of IBM Spectrum Scale • Reverting to the previous level of IBM Spectrum Scale	System administrators, analysts, installers, planners, and programmers of IBM Spectrum Scale clusters who are very experienced with the operating systems on which each IBM Spectrum Scale cluster is based
IBM Spectrum Scale: Concepts, Planning, and Installation Guide	• Coexistence considerations • Compatibility considerations • Considerations for IBM Spectrum Protect for Space Management • Applying maintenance to your GPFS system • Guidance for upgrading the operating system on IBM Spectrum Scale nodes • Servicing IBM Spectrum Scale protocol nodes • Offline upgrade with complete cluster shutdown	

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
IBM Spectrum Scale: Administration Guide	This guide provides the following information: **Configuring** • Configuring the GPFS cluster • Configuring the CES and protocol configuration • Configuring and tuning your system for GPFS • Parameters for performance tuning and optimization • Ensuring high availability of the GUI service • Configuring and tuning your system for Cloud services • Configuring IBM Power Systems for IBM Spectrum Scale • Configuring the message queue • Configuring file audit logging • Configuring clustered watch folder • Configuring Active File Management • Configuring AFM-based DR • Tuning for Kernel NFS backend on AFM and AFM DR • Configuring call home **Administering** • Performing GPFS administration tasks • Verifying network operation with the mmnetverify command • Managing file systems • File system format changes between versions of IBM Spectrum Scale • Managing disks • Managing protocol services • Managing protocol user authentication • Managing protocol data exports • Managing object storage • Managing GPFS quotas • Managing GUI users • Managing GPFS access control lists	System administrators or programmers of IBM Spectrum Scale systems

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
IBM Spectrum Scale: Administration Guide	• Native NFS and GPFS • Considerations for GPFS applications • Accessing a remote GPFS file system • Information lifecycle management for IBM Spectrum Scale • Creating and maintaining snapshots of file systems • Creating and managing file clones • Scale Out Backup and Restore (SOBAR) • Data Mirroring and Replication • Implementing a clustered NFS environment on Linux • Implementing Cluster Export Services • Identity management on Windows / RFC 2307 Attributes • Protocols cluster disaster recovery • File Placement Optimizer • Encryption • Managing certificates to secure communications between GUI web server and web browsers • Securing protocol data • Cloud services: Transparent cloud tiering and Cloud data sharing • Managing file audit logging • Performing a watch with watch folder API • RDMA tuning • Configuring Mellanox Memory Translation Table (MTT) for GPFS RDMA VERBS Operation • Administering AFM • Administering AFM DR • Highly-available write cache (HAWC) • Local read-only cache • Miscellaneous advanced administration • GUI limitations	System administrators or programmers of IBM Spectrum Scale systems

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Problem Determination Guide	This guide provides the following information: **Monitoring** • Performance monitoring • Monitoring system health through the IBM Spectrum Scale GUI • Monitoring system health by using the mmhealth command • Monitoring events through callbacks • Monitoring capacity through GUI • Monitoring AFM and AFM DR • GPFS SNMP support • Monitoring the IBM Spectrum Scale system by using call home • Monitoring remote cluster through GUI • Monitoring the message queue • Monitoring file audit logging • Monitoring clustered watch **Troubleshooting** • Best practices for troubleshooting • Understanding the system limitations • Collecting details of the issues • Managing deadlocks • Installation and configuration issues • Upgrade issues • Network issues • File system issues • Disk issues • Security issues • Protocol issues • Disaster recovery issues • Performance issues	System administrators of GPFS systems who are experienced with the subsystems used to manage disks and who are familiar with the concepts presented in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
IBM Spectrum Scale: Problem Determination Guide	• GUI issues • AFM issues • AFM DR issues • Transparent cloud tiering issues • File audit logging issues • Troubleshooting watch folder API • Troubleshooting mmwatch • Message queue issues • Maintenance procedures • Recovery procedures • Support for troubleshooting • References	

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
IBM Spectrum Scale: Command and Programming Reference	This guide provides the following information: **Command reference** • gpfs.snap command • mmaddcallback command • mmadddisk command • mmaddnode command • mmadquery command • mmafmconfig command • mmafmctl command • mmafmlocal command • mmapplypolicy command • mmaudit command • mmauth command • mmbackup command • mmbackupconfig command • mmblock command • mmbuildgpl command • mmcachectl command • mmcallhome command • mmces command • mmcesdr command • mmchattr command • mmchcluster command • mmchconfig command • mmchdisk command • mmcheckquota command • mmchfileset command • mmchfs command • mmchlicense command • mmchmgr command • mmchnode command • mmchnodeclass command • mmchnsd command • mmchpolicy command • mmchpool command • mmchqos command • mmclidecode command	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Command and Programming Reference	• mmclone command • mmcloudgateway command • mmcrcluster command • mmcrfileset command • mmcrfs command • mmcrnodeclass command • mmcrnsd command • mmcrsnapshot command • mmdefedquota command • mmdefquotaoff command • mmdefquotaon command • mmdefragfs command • mmdelacl command • mmdelcallback command • mmdeldisk command • mmdelfileset command • mmdelfs command • mmdelnode command • mmdelnodeclass command • mmdelnsd command • mmdelsnapshot command • mmdf command • mmdiag command • mmdsh command • mmeditacl command • mmedquota command • mmexportfs command • mmfsck command • mmfsctl command • mmgetacl command • mmgetstate command • mmhadoopctl command • mmhdfs command • mmhealth command • mmimgbackup command • mmimgrestore command • mmimportfs command • mmkeyserv command	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

Table 1. IBM Spectrum Scale library information units (continued)		
Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Command and Programming Reference	• mmlinkfileset command • mmlsattr command • mmlscallback command • mmlscluster command • mmlsconfig command • mmlsdisk command • mmlsfileset command • mmlsfs command • mmlslicense command • mmlsmgr command • mmlsmount command • mmlsnodeclass command • mmlsnsd command • mmlspolicy command • mmlspool command • mmlsqos command • mmlsquota command • mmlssnapshot command • mmmigratefs command • mmmount command • mmmsgqueue command • mmnetverify command • mmnfs command • mmnsddiscover command • mmobj command • mmperfmon command • mmpmon command • mmprotocoltrace command • mmpsnap command • mmputacl command • mmquotaoff command • mmquotaon command • mmreclaimspace command • mmremotecluster command • mmremotefs command • mmrepquota command • mmrestoreconfig command • mmrestorefs command • mmrestripefile command	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	**Type of information**	**Intended users**
IBM Spectrum Scale: Command and Programming Reference	• mmrestripefs command • mmrpldisk command • mmsdrrestore command • mmsetquota command • mmshutdown command • mmsmb command • mmsnapdir command • mmstartup command • mmtracectl command • mmumount command • mmunlinkfileset command • mmuserauth command • mmwatch command • mmwinservctl command • spectrumscale command **Programming reference** • IBM Spectrum Scale Data Management API for GPFS information • GPFS programming interfaces • GPFS user exits • IBM Spectrum Scale management API commands • Watch folder API	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
IBM Spectrum Scale: Big Data and Analytics Guide	This guide provides the following information: Summary of changes Hadoop Scale Storage Architecture • Elastic Storage Server (ESS) • Erasure Code Edition • Share Storage (SAN-based storage) • File Placement Optimizer (FPO) • Deployment model • Additional supported features about storage IBM Spectrum Scale support for Hadoop • HDFS transparency • Supported IBM Spectrum Scale storage modes • Hadoop cluster planning • CES HDFS • Installation and configuration of HDFS transparency • Application interaction with HDFS transparency • Upgrading the HDFS Transparency cluster • Rolling upgrade for HDFS Transparency • Security • Advanced features • Hadoop distribution support • Limitations and differences from native HDFS • Problem determination IBM Spectrum Scale Hadoop performance tuning guide • Overview • Performance overview • Hadoop Performance Planning over IBM Spectrum Scale • Performance guide	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

Table 1. IBM Spectrum Scale library information units (continued)

Information unit	Type of information	Intended users
IBM Spectrum Scale: Big Data and Analytics Guide	Hortonworks Data Platform 3.X • Planning • Installation • Upgrading and uninstallation • Configuration • Administration • Limitations • Problem determination Open Source Apache Hadoop • Open Source Apache Hadoop without CES HDFS • Open Source Apache Hadoop with CES HDFS BigInsights® 4.2.5 and Hortonworks Data Platform 2.6 • Planning • Installation • Upgrading software stack • Configuration • Administration • Troubleshooting • Limitations • FAQ	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

Information unit	Type of information	Intended users
IBM Spectrum Scale Erasure Code Edition Guide	IBM Spectrum Scale Erasure Code Edition • Summary of changes • Introduction to IBM Spectrum Scale Erasure Code Edition • Planning for IBM Spectrum Scale Erasure Code Edition • Installing IBM Spectrum Scale Erasure Code Edition • Uninstalling IBM Spectrum Scale Erasure Code Edition • Incorporating IBM Spectrum Scale Erasure Code Edition in an Elastic Storage Server (ESS) cluster • Creating an IBM Spectrum Scale Erasure Code Edition storage environment • Upgrading IBM Spectrum Scale Erasure Code Edition • Administering IBM Spectrum Scale Erasure Code Edition • Troubleshooting • IBM Spectrum Scale RAID Administration	• System administrators of IBM Spectrum Scale systems • Application programmers who are experienced with IBM Spectrum Scale systems and familiar with the terminology and concepts in the XDSM standard

Table 1. IBM Spectrum Scale library information units (continued)

Prerequisite and related information

For updates to this information, see IBM Spectrum Scale in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/ibmspectrumscale_welcome.html).

For the latest support information, see the IBM Spectrum Scale FAQ in IBM Knowledge Center (www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html).

Conventions used in this information

Table 2 on page xxii describes the typographic conventions used in this information. UNIX file name conventions are used throughout this information.

Note: Users of IBM Spectrum Scale for Windows must be aware that on Windows, UNIX-style file names need to be converted appropriately. For example, the GPFS cluster configuration data is stored in the `/var/mmfs/gen/mmsdrfs` file. On Windows, the UNIX namespace starts under the `%SystemDrive%\cygwin64` directory, so the GPFS cluster configuration data is stored in the `C:\cygwin64\var\mmfs\gen\mmsdrfs` file.

Table 2. Conventions

Convention	Usage
bold	Bold words or characters represent system elements that you must use literally, such as commands, flags, values, and selected menu options. Depending on the context, **bold** typeface sometimes represents path names, directories, or file names.
bold underlined	`bold underlined` keywords are defaults. These take effect if you do not specify a different keyword.
`constant width`	Examples and information that the system displays appear in `constant-width` typeface. Depending on the context, `constant-width` typeface sometimes represents path names, directories, or file names.
italic	*Italic* words or characters represent variable values that you must supply. *Italics* are also used for information unit titles, for the first use of a glossary term, and for general emphasis in text.
<*key*>	Angle brackets (less-than and greater-than) enclose the name of a key on the keyboard. For example, <`Enter`> refers to the key on your terminal or workstation that is labeled with the word *Enter*.

\	In command examples, a backslash indicates that the command or coding example continues on the next line. For example:
	`mkcondition -r IBM.FileSystem -e "PercentTotUsed > 90" \`
	`-E "PercentTotUsed < 85" -m p "FileSystem space used"`

{*item*}	Braces enclose a list from which you must choose an item in format and syntax descriptions.
[*item*]	Brackets enclose optional items in format and syntax descriptions.
<`Ctrl-`*x*>	The notation <`Ctrl-`*x*> indicates a control character sequence. For example, <`Ctrl-c`> means that you hold down the control key while pressing <`c`>.
item...	Ellipses indicate that you can repeat the preceding item one or more times.
|	In *synopsis* statements, vertical lines separate a list of choices. In other words, a vertical line means *Or*. In the left margin of the document, vertical lines indicate technical changes to the information.

Note: CLI options that accept a list of option values delimit with a comma and no space between values. As an example, to display the state on three nodes use `mmgetstate -N` *NodeA*,*NodeB*,*NodeC*. Exceptions to this syntax are listed specifically within the command.

How to send your comments

Your feedback is important in helping us to produce accurate, high-quality information. If you have any comments about this information or any other IBM Spectrum Scale documentation, send your comments to the following e-mail address:

`mhvrcfs@us.ibm.com`

Include the publication title and order number, and, if applicable, the specific location of the information about which you have comments (for example, a page number or a table number).

To contact the IBM Spectrum Scale development organization, send your comments to the following e-mail address:

```
scale@us.ibm.com
```

Chapter 1. Summary of changes

Summary of changes for IBM Spectrum Scale Container Storage Interface driver.

The following enhancements have been made in this release:

- Support for Red Hat® OpenShift® 4.4, 4.5, and Kubernetes 1.18
- Support for Red Hat Enterprise Linux (RHEL) 7.8.
- Support for imagePullSecrets
- Upgraded the operator-sdk version from 0.11.0 to 0.17.0
- Upgraded the ansible-operator version from v0.14.0 to v0.17.0
- Made the clusterId parameter optional in storageClass (default: Primary Cluster ID)
- Made the primaryFset parameter optional for creating the primary fileset (default: spectrum-scale-csi-volume-store)
- Improved custom resource parameter validation and error messages
- Improvements in the debug data collection tool
- Removed the remoteFs parameter from the csiscaleoperators.csi.ibm.com_cr.yaml file.

Chapter 2. Introduction to IBM Spectrum Scale Container Storage Interface driver

This section provides a brief introduction to IBM Spectrum Scale Container Storage Interface driver.

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 on IBM Spectrum Scale features, see the *Product overview* section in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.

Container Storage Interface (CSI) is a standard for exposing arbitrary block and file storage systems to containerized workloads on Container Orchestration Systems (COs) like Kubernetes. The IBM Spectrum Scale Container Storage Interface driver specification is defined at https://github.com/container-storage-interface/spec/blob/master/spec.md.

IBM Spectrum Scale Container Storage Interface driver allows IBM Spectrum Scale to be used as a persistent storage for stateful application running in Kubernetes clusters. Through the IBM Spectrum Scale Container Storage Interface driver, Kubernetes persistent volumes (PVs) can be provisioned from IBM Spectrum Scale. Thus, the containers can be used with stateful microservices like MongoDB, PostgreSQL, and so on.

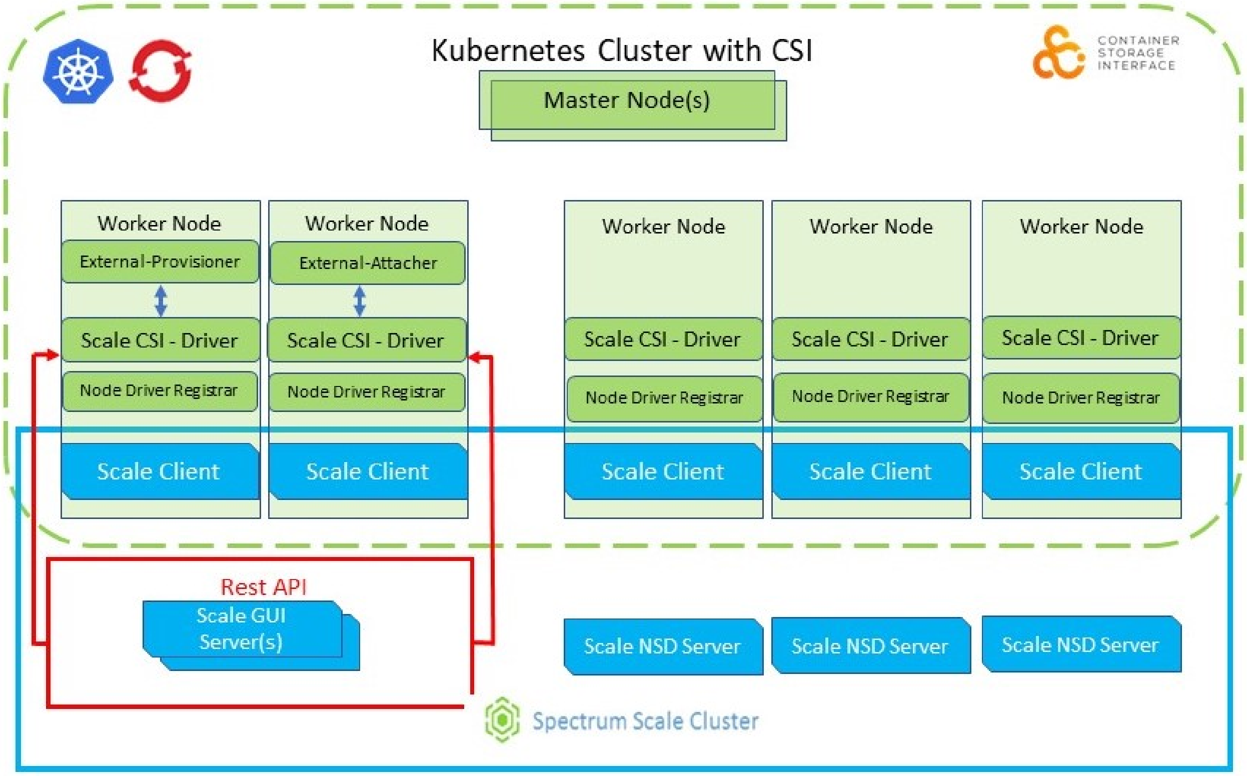

Figure 1. IBM Spectrum Scale Container Storage Interface Driver

IBM implements the CSI specification of storage plug-in in the following manner:

The external-provisioner and external-attacher Sidecar Containers are StatefulSets, which should be deployed on separate infrastructure hosts for resiliency. The external-provisioner watches for create or delete API calls while the external-attacher watches for mount or unmount API calls. The node-driver-registrar is a kubelet service, which runs alongside the node plug-in at the time of initialization. The IBM

Spectrum Scale Container Storage Interface driver provides the interconnect for persistent volume mount from the container worker node to the underlying storage system and is deployed as a DaemonSet. The IBM Spectrum Scale Container Storage Interface driver also makes the REST API calls on the IBM Spectrum Scale storage system to perform any provisioning requests on the storage. For more information on StatefulSets and Daemonsets, see StatefulSets.

Features covered

The following features are available with IBM Spectrum Scale Container Storage Interface driver:

- Static provisioning: Ability to use existing directories and filesets as persistent volumes
- Lightweight dynamic provisioning: Ability to create directory-based volumes dynamically
- Fileset-based dynamic provisioning: Ability to create fileset-based volumes dynamically
- Multiple file systems support: Volumes can be created across multiple file systems
- Remote mount support: Volumes can be created on a remotely mounted file system
- Operator support for easy deployment, upgrade, and cleanup
- Supported volume access mode: RWX (Read-write many) and RWO (ReadWriteOnce)

Chapter 3. Planning for IBM Spectrum Scale Container Storage Interface driver

This topic describes the planning information for using IBM Spectrum Scale Container Storage Interface driver.

Note: If you are using a Red Hat OpenShift cluster, ensure to replace "kubectl" with "oc" in all commands.

Hardware and Software requirements

The following hardware and software requirements must be met for using IBM Spectrum Scale Container Storage Interface driver at your site:

- RHEL version 7.5, 7.6, 7.7, and 7.8 (x86_64 architecture)
- IBM Spectrum Scale version 5.0.4.1 or later
- Kubernetes version 1.16, 1.17, and 1.18
- Red Hat OpenShift 4.3, 4.4, and 4.5 (x86_64 architecture) through RHEL worker nodes

Deployment considerations

Ensure that the following steps are completed before deploying IBM Spectrum Scale Container Storage Interface driver in your cluster.

- **Worker nodes selection**: By default, Kubernetes/Red Hat OpenShift schedules the IBM Spectrum Scale Container Storage Interface driver pods on all worker nodes. It is essential to have IBM Spectrum Scale client installed on all these nodes. If you wish to schedule the IBM Spectrum Scale Container Storage Interface driver pods only on selected worker nodes, you must label the selected nodes and use this label in node selector. For more information, see “Using the node selector” on page 25.
- **Node selection for StatefulSets**: CSI external attacher and CSI external provisioner are sidecar containers that run as two separate StatefulSets. These pods can be scheduled on any of the worker nodes by Kubernetes. As a best practice, it is recommended to run these on two separate stable nodes. The StatefulSets by design of Kubernetes do not automatically fail over to another node, hence it is recommended to schedule them to run on reliable nodes. On Red Hat OpenShift, if the infrastructure nodes are worker nodes, it is recommended to schedule the sidecar containers to run on infrastructure nodes. Scheduling them to run on specific nodes can be achieved by using nodes labels and nodeSelectors. For more information, see “Using the node selector” on page 25. IBM Spectrum Scale Container Storage Interface driver pod must also be scheduled on the nodes that runStatefulSets.
- **Remote cluster setup**: If you plan to use remotely mounted file system for PVC provisioning, ensure the following:
  - IBM Spectrum Scale GUI is initialized and running on both clusters (owning cluster and accessing cluster)
  - Remote cluster details are added to the Operator configuration. For more information, see “Remote cluster support” on page 21.
- **SELinux considerations**: Different Kubernetes distributions handle the SELinux enforcing mode differently. There might be differences in terms of SELinux context being set on files, relabeling of volumes and the process context of containers. As a prerequisite, appropriate SELinux rules must be set up to allow IBM Spectrum Scale Container Storage Interface driver containers to access the required resources on host. For example “container_t” context needs to have access to `csi.sock` and the IBM Spectrum Scale file system, or the files needing access from containers need to have the “`container_file_t`” context set. Refer to audit logs for any SELinux failures and set up appropriate rules as required.

- **Node names**: At times, it is possible that IBM Spectrum Scale cluster and Kubernetes/Red Hat OpenShift cluster are configured with different node names for the same host. Use the **`mmlscluster`** and **`kubectl get nodes`** commands and check the node names of IBM Spectrum Scale cluster and Kubernetes cluster. If the names are different, then configure node mapping in the Operator configuration. For more information, see "Kubernetes to IBM Spectrum Scale node mapping" on page 26.

  Note: Node names should match with the output of the **`mmlsmount <fsname> -L`** command, or the curl command **`curl --insecure -u 'username:password' -X GET https://guiHost:443/scalemgmt/v2/filesystems/<filesystemname>?fields=mount`**.

- **Internet connectivity**: If your worker nodes have internet connectivity and access to the quay.io registry, the IBM Spectrum Scale Container Storage Interface driver downloads and uploads the required images automatically during deployment. Otherwise, you need to manually download the following images:

  - `quay.io/ibm-spectrum-scale/ibm-spectrum-scale-csi-driver:v2.0.0`
  - `quay.io/ibm-spectrum-scale/ibm-spectrum-scale-csi-operator:v2.0.0`
  - `quay.io/k8scsi/csi-node-driver-registrar:v1.2.0`
  - `quay.io/k8scsi/csi-attacher:v2.1.1`
  - `quay.io/k8scsi/csi-provisioner:v1.5.0`

  Note: Upload these images on all worker nodes using **`docker/podman`** command. In case you use a private registry for your Red Hat OpenShift/Kubernetes environment, upload the images to your private registry. Use the new image URL and registry credentials in the `imagePullSecrets` section of the `csiscaleoperators.csi.ibm.com_cr.yaml` file. For more information, see the following topics:

  - To create secrets using private registry credentials, see Pull an Image from a Private Registry
  - For information on the `imagePullSecrets` section, see the `csiscaleoperators.csi.ibm.com_cr.yaml` file on the "Operator" on page 19 page.

- **IBM Spectrum Scale services**: As a best practice, it is recommended that GUI nodes, protocol nodes, and NSD nodes are not part of the Kubernetes cluster.

Chapter 4. Installation of IBM Spectrum Scale Container Storage Interface driver

This section explains how to install or clean up the IBM Spectrum Scale Container Storage Interface driver. CSI Operators are used for performing these activities.

Performing pre-installation tasks

Complete the following tasks before you start installing the IBM Spectrum Scale Container Storage Interface driver:

- Install IBM Spectrum Scale along with the IBM Spectrum Scale management API (GUI).
  - For the supported versions, see IBM Spectrum Scale FAQ.
  - For information on installing IBM Spectrum Scale, see the *Installing IBM Spectrum Scale on Linux nodes with the installation toolkit* topic in *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
- Install and set up either Kubernetes or Red Hat OpenShift.
  - For supported versions, see IBM Spectrum Scale FAQ.
  - For information on installing Kubernetes, see the Install and Set Up kubectl topic in the Kubernetes documentation.
  - For information on installing Red Hat OpenShift, see the Installing and configuring OpenShift Container Platform clusters topic in the Red Hat OpenShift documentation.
- Install the IBM Spectrum Scale client on the required Kubernetes worker nodes, and these nodes are added to the IBM Spectrum Scale cluster.
- Mount the primary file system that is used for IBM Spectrum Scale configuration on all Kubernetes worker nodes and on the IBM Spectrum Scale GUI nodes.
- Create an IBM Spectrum Scale user group "CsiAdmin" if it does not already exist. Use the `/usr/lpp/mmfs/gui/cli/mkusergrp CsiAdmin --role csiadmin` command to create the CsiAdmin user. Create an IBM Spectrum Scale user in the "CsiAdmin" group. This user must be used on IBM Spectrum Scale Container Storage Interface driver configuration. Issue this command on the GUI node to create the user:

  ```
  /usr/lpp/mmfs/gui/cli/mkuser <username> -p <password> -g CsiAdmin
  ```

- Ensure that all the worker nodes where IBM Spectrum Scale Container Storage Interface driver should be scheduled are running Red Hat Enterprise Linux (RHEL) x86_64. For supported versions, see https://www.ibm.com/support/knowledgecenter/en/STXKQY/gpfsclustersfaq.html.
- Initialize the IBM Spectrum Scale GUI either by logging into the GUI console once or by running this command on the GUI node:

  ```
  /usr/lpp/mmfs/gui/cli/initgui
  ```

- Issue the following command from the Kubernetes node to ensure that the GUI server is running and can communicate with the Kubernetes nodes:

  ```
  curl --insecure -u 'username:password' -X GET https://guihost:443/scalemgmt/v2/cluster
  ```

  The command gives an output similar to the following:

  ```
  {
    "cluster" : {
      "clusterSummary" : {
        "clusterId" : 17258972170939727157,
  ```

```
      "clusterName" : "node10.node10",
      "primaryServer" : "node10",
      "rcpPath" : "/usr/bin/scp",
      "rcpSudoWrapper" : false,
      "repositoryType" : "CCR",
      "rshPath" : "/usr/bin/ssh",
      "rshSudoWrapper" : false,
      "uidDomain" : "node10.node10"
    },
    "capacityLicensing" : {
      "liableCapacity" : 96636764160,
      "liableNsdCount" : 2,
      "liableNsds" : [ {
        "nsdName" : "nsd1",
        "liableCapacity" : 53687091200
      }, {
        "nsdName" : "nsd2",
        "liableCapacity" : 42949672960
      } ]
    }
  },
  "status" : {
    "code" : 200,
    "message" : "The request finished successfully."
  }
}
```

- Ensure that 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
```

- Enable quota for all the file systems (if not already done) being used for creating persistent volumes that are required for fileset-based dynamic provisioning.

```
mmchfs gpfs0 -Q yes
```

  Verify that quota is enabled.

```
mmlsfs gpfs0 -Q
```

  This command gives an output similar to the following:

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

- Enable quota for `root` user by issuing this command:

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

- For Red Hat OpenShift, ensure that the `controlSetxattrImmutableSELinux` parameter is set to "yes" by issuing the following command:

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

- To display the correct volume size in the container, enable filesetdf of the file system by using the following command:

```
mmchfs <filesystem name> --filesetdf
```

- Mount the file system that is used for IBM Spectrum Scale Container Storage Interface driver on the same mount point on worker nodes.

- Issue the following command to label the Kubernetes worker nodes where IBM Spectrum Scale client is installed and where IBM Spectrum Scale Container Storage Interface driver should run:

  ```
  kubectl label node node1 scale=true --overwrite=true
  ```

  For more information, see "Using the node selector" on page 25.

Installing IBM Spectrum Scale Container Storage Interface driver using Operator Lifecycle Manager

This topic describes the procedure for installing IBM Spectrum Scale Container Storage Interface driver using Operator Lifecycle Manager (OLM).

OLM runs by default on Red Hat OpenShift Container Platform 4.2 and later releases. For more information, see Operator Lifecycle Manager workflow and architecture. OLM is not available by default on Kubernetes, and it is recommended to use CLI-based installation of the IBM Spectrum Scale Container Storage Interface driver on Kubernetes. For more information, see "Installing IBM Spectrum Scale Container Storage Interface driver using CLIs" on page 10. However, if OLM is already installed on the Kubernetes cluster, then use the method that is described here.

Note: If you refer to the installation instructions on the Operator Hub (https://operatorhub.io/operator/ibm-spectrum-scale-csi-operator), make note of the following:

- Ignore step #2 (`kubectl create -f https://operatorhub.io/install/ibm-spectrum-scale-csi-operator.yaml`), and do the following steps:

  1. Download the yaml file.

     ```
     curl -O https://operatorhub.io/install/ibm-spectrum-scale-csi-operator.yaml
     ```

  2. Remove "my-" from any names.

     ```
     sed -i 's/my-//g' ./ibm-spectrum-scale-csi-operator.yaml
     ```

  3. Apply the yaml file.

     ```
     kubectl apply -f ibm-spectrum-scale-csi-operator.yaml
     ```

- Ignore step #3 (`kubectl get csv -n my-ibm-spectrum-scale-csi-operator`), and issue the following command:

  ```
  kubectl get csv -n ibm-spectrum-scale-csi-operator
  ```

Before performing IBM Spectrum Scale Container Storage Interface driver installation, ensure that the prerequisites are met. For more information, see "Performing pre-installation tasks" on page 7.

Create the Operator from the Red Hat OpenShift console by performing the following steps:

a. Log in to the Red Hat OpenShift Container Platform.

b. From the left panel, click **Operators** > **OperatorHub**. The OperatorHub page appears.

c. From the **Project** drop-down list, select the project or create a new project by clicking **Create Project**.

d. Under **All Items**, select **Storage** from the list.

e. In the **Filter by keyword** box, type "`IBM Spectrum Scale CSI`".

f. Click **IBM Spectrum Scale CSI Plugin Operator**. The IBM Spectrum Scale CSI Plugin Operator page appears.

g. Click **Install**. The Create Operator Subscription page appears.

h. On this page, select a namespace as that of operator from the available options, select the approval strategy (automatic or manual), and click **Subscribe**. The Installed Operators page appears, where *IBM Spectrum Scale CSI Plugin Operator* is listed as successfully installed.

i. On the Installed Operators page, click **IBM Spectrum Scale CSI Plugin Operator**, and go to the **IBM CSI Spectrum Scale Driver** tab.

j. Click **Create CSIScale Operator**. The Create CSIScale Operator page appears.

k. On this page, an editor appears, where you can update the manifest file according to your environment. For more information, see "Operator" on page 19, "Secrets" on page 19, and "Certificates" on page 19.

Installing IBM Spectrum Scale Container Storage Interface driver using CLIs

This topic describes the procedures for installing IBM Spectrum Scale Container Storage Interface driver by using CLIs.

Before performing IBM Spectrum Scale Container Storage Interface driver installation, ensure that the prerequisites are met. For more information, see "Performing pre-installation tasks" on page 7.

Note: This procedure is applicable for both Kubernetes and Red Hat OpenShift. For Red Hat OpenShift, replace "kubectl" with "oc" in all the commands.

Installing IBM Spectrum Scale Container Storage Interface driver using Operator involves the following phases:

1. Deploy the Operator on your cluster
2. Use the Operator for deploying IBM Spectrum Scale Container Storage Interface driver

Phase 1: Deploying the Operator

To deploy Operator on your cluster, do the following steps:

1. Create a namespace.

   ```
   kubectl create namespace ibm-spectrum-scale-csi-driver
   ```

   Note: For Red Hat OpenShift, use this command:

   ```
   oc new-project ibm-spectrum-scale-csi-driver
   ```

2.
   ```
   kubectl create -f https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/
   generated/\
   installer/ibm-spectrum-scale-csi-operator.yaml
   ```

3. Verify that the Operator is deployed, and the Operator pod is in running state.

   ```
   kubectl get pod,deployment -n ibm-spectrum-scale-csi-driver
   ```

   ```
   
   NAME                                                   READY   STATUS    RESTARTS   AGE
   pod/ibm-spectrum-scale-csi-operator-6ff9cf6979-v5g4c   1/1     Running   0          6d3h

   NAME                                                   READY   UP-TO-DATE  AVAILABLE AGE
   deployment.extensions/ibm-spectrum-scale-csi-operator  1/1     1           1         6d3h
   ```

Phase 2: Deploying IBM Spectrum Scale Container Storage Interface driver

Now that the Operator is up and running, you must access the Operator's API and request a deployment. This is done through the use of the *CSIScaleOperator* custom resource.

Do the following steps:

1. Create a secret with IBM Spectrum Scale GUI server's credentials in the `ibm-spectrum-scale-csi-driver` namespace. For more information, see "Secrets" on page 19.

   Note: If you are using a remote cluster setup, then create a secret object for each cluster's GUI server.

2. Download the sample custom resource file on your cluster:

```
curl -O https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/operator/deploy/
crds/\
csiscaleoperators.csi.ibm.com_cr.yaml
```

3. Modify the parameters in the file to suit your environment. For more information, see "Operator" on page 19.
4. Apply the custom resource file to deploy IBM Spectrum Scale Container Storage Interface driver.

```
kubectl apply -f csiscaleoperators.csi.ibm.com_cr.yaml
```

5. Verify that IBM Spectrum Scale Container Storage Interface driver is installed, Operator and driver resources are ready, and pods are in running state:

```
# kubectl get pod,daemonset,statefulset -n ibm-spectrum-scale-csi-driver
NAME                                       READY   STATUS             RESTARTS   AGE
pod/ibm-spectrum-scale-csi-attacher-0      1/1     Running            0          4d2h
pod/ibm-spectrum-scale-csi-dxslh           2/2     Running            0          4d2h
pod/ibm-spectrum-scale-csi-provisioner-0   1/1     Running            0.         4d2h
pod/ibm-spectrum-scale-csi-operator-6ff9cf6979-v5g4c   1/1     Running   0          1h

NAME                                    DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE
NODE SELECTOR   AGE
daemonset.apps/ibm-spectrum-scale-csi   1         1         1       1            1
<none>          4d2h

NAME                                                  READY   AGE
statefulset.apps/ibm-spectrum-scale-csi-attacher      1/1     4d2h
statefulset.apps/ibm-spectrum-scale-csi-provisioner   1/1     4d2h
```

For more information, see https://github.com/IBM/ibm-spectrum-scale-csi.

Cleaning up IBM Spectrum Scale Container Storage Interface driver and Operators by using OLM

This topic describes the procedure for uninstalling IBM Spectrum Scale Container Storage Interface driver and Operator by using OLM.

Cleanup procedure includes two tasks: deleting IBM Spectrum Scale Container Storage Interface driver and deleting the Operator.

Cleaning up IBM Spectrum Scale Container Storage Interface driver

Do the following steps:

1. Log in to the Red Hat OpenShift Container Platform.
2. Click **Operators** > **Installed Operators**. The Installed Operators page appears, where *IBM Spectrum Scale CSI Plugin Operator* is listed.
3. From the **Project : default** drop-down list, select the project in which the Operator is installed.
4. On the Installed Operators page, click **IBM Spectrum Scale CSI Plugin**. The Operator Details page appears.
5. Go to the **IBM CSI Spectrum Scale Driver** tab.
6. Click **ibm-spectrum-scale-csi** and select **CSIScale Operator Delete** from the **Actions** drop-down list.
7. Click **Delete** to confirm deletion.

Cleaning up the Operator

Do the following steps:

1. Log in to the Red Hat OpenShift Container Platform.
2. Click **Operators** > **Installed Operators**. The Installed Operators page appears, where *IBM Spectrum Scale CSI Plugin Operator* is listed.

3. From the **Project : default** drop-down list, select the project in which the Operator is installed.
4. Click **IBM Spectrum Scale CSI Plugin Operator** and select **Uninstall Operator** from the **Actions** drop-down list.
5. Click **Remove** for confirmation.

Cleaning up IBM Spectrum Scale Container Storage Interface driver and Operator using CLIs

This topic describes the procedure to clean up or uninstall the IBM Spectrum Scale Container Storage Interface driver and the Operator.

To manage IBM Spectrum Scale Container Storage Interface driver, it is essential that the Operator is always running on your cluster. If the Operator is deleted for some reason, ensure to re-deploy it by using the `kubectl apply -f ibm-spectrum-scale-csi-operator.yaml` command before proceeding with the following steps:

1. To stop and uninstall IBM Spectrum Scale Container Storage Interface driver, issue the following command:

   ```
   kubectl delete -f csiscaleoperators.csi.ibm.com_cr.yaml
   ```

2. To uninstall Operator and clean up all resources, issue the following commands:

   ```
   kubectl delete -f ibm-spectrum-scale-csi-operator.yaml
   kubectl delete namespace ibm-spectrum-scale-csi-driver
   ```

   Note: Delete the secrets for GUI credentials and configmap for CA certificates (if any) under the `ibm-spectrum-scale-csi-driver` namespace.

3. To completely remove IBM Spectrum Scale Container Storage Interface driver and the Operator images, do the following steps:

   a. Find the images by issuing the following command:

   ```
   # docker images | grep spectrum-scale-csi
   quay.io/ibm-spectrum-scale/ibm-spectrum-scale-csi-driver     v2.0.0   8b6cb1a1743e   4
   hours ago     122 MB
   quay.io/ibm-spectrum-scale/ibm-spectrum-scale-csi-operator   v2.0.0   ced0267b0a45   4
   hours ago     505 MB
   ```

   b. Remove the images by issuing the following command:

   ```
   # docker rmi 8b6cb1a1743e
   ```

   Note: If you use a different container engine than Docker, replace the Docker commands with the commands of the container engine that you use.

4. To delete PVC data, unlink and delete the primary fileset defined in the `csiscaleoperators.csi.ibm.com_cr.yaml` file from your IBM Spectrum Scale cluster, by issuing the following commands:

   ```
   /usr/lpp/mmfs/bin/mmunlinkfileset gpfs0 spectrum-scale-csi-volume-store
   /usr/lpp/mmfs/bin/mmdelfileset gpfs0 spectrum-scale-csi-volume-store
   ```

   Note: This will completely delete the PVC data, and any PVCs that were created before will no longer be useful even if the IBM Spectrum Scale Container Storage Interface driver is reinstalled.

Chapter 5. Upgrading IBM Spectrum Scale Container Storage Interface driver

This topic describes the procedure for upgrading IBM Spectrum Scale Container Storage Interface driver.

From version 1.1.0 to version 2.0.0

Perform the following steps to upgrade IBM Spectrum Scale Container Storage Interface driver from version 1.1.0 to version 2.0.0:

1. Download the operator manifest file on your cluster by issuing the following command:

   ```
   curl -O https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/generated\
   /installer/ibm-spectrum-scale-csi-operator.yaml
   ```

2. Apply the manifest file by issuing the following command:

   ```
   kubectl apply -f ibm-spectrum-scale-csi-operator.yaml
   ```

   This upgrades the Operator and the IBM Spectrum Scale Container Storage Interface driver. The Operator and the pods are restarted with the upgraded image.

3. Verify that the pods are back in the running state by issuing the following command:

   ```
   # kubectl get pod -n ibm-spectrum-scale-csi-driver
   ```

   The system displays an output similar to this:

   ```
   NAME                                              READY   STATUS    RESTARTS   AGE
   ibm-spectrum-scale-csi-662zv                      2/2     Running   0          29m
   ibm-spectrum-scale-csi-87rw9                      2/2     Running   0          29m
   ibm-spectrum-scale-csi-attacher-0                 1/1     Running   0          29m
   ibm-spectrum-scale-csi-operator-ddfcc6cb5-z9g29   1/1     Running   0          31m
   ibm-spectrum-scale-csi-provisioner-0              1/1     Running   0          29m
   ibm-spectrum-scale-csi-wkg42                      2/2     Running   0          29m
   ```

4. Verify that the Operator and the pods are using the upgraded images by issuing the following command:

   ```
   # kubectl describe pod ibm-spectrum-scale-csi-operator-ddfcc6cb5-z9g29 -n ibm-spectrum-scale-
   csi-driver \|
    grep "Image:" | grep ibm-spectrum-scale

       Image:         quay.io/ibm-spectrum-scale/ibm-spectrum-scale-csi-operator:v2.0.0
   ```

   ```
   # kubectl describe pod ibm-spectrum-scale-csi-662zv -n ibm-spectrum-scale-csi-driver |\
   grep "Image:" | grep ibm-spectrum-scale

       Image:         quay.io/ibm-spectrum-scale/ibm-spectrum-scale-csi-driver:v2.0.0
   ```

Chapter 6. Migrating from IBM Storage Enabler for Containers to IBM Spectrum Scale Container Storage Interface driver

This topic describes the procedure for migrating from IBM Storage Enabler for Containers to IBM Spectrum Scale Container Storage Interface driver.

There can be three scenarios in IBM Storage Enabler for Containers, as follows:

- **Kubernetes is used**: In this case, Kubernetes version 1.12 or earlier used in IBM Storage Enabler for Containers should be upgraded to 1.16 or later.
- **IBM Cloud Private (ICP) is used**. In this case, ICP should be upgraded to Red Hat OpenShift 4.x by:
  - Uninstalling ICP
  - Freshly installing Red Hat OpenShift 4.x
- **Red Hat OpenShift is used**. In this case, Red Hat OpenShift version 3.x should be upgraded to version 4.x by:
  - Uninstalling Red Hat OpenShift 3.x
  - Freshly installing Red Hat OpenShift 4.x

  Note: During the Red Hat OpenShift upgrade, IBM Spectrum Scale client nodes are removed and are added back after the upgrade.

After the upgrade

- Dynamic provisioning, especially deletion, will not work on migrated volumes.
- All dynamically provisioned volumes from IBM Storage Enabler for Containers will be used as static provisioned volumes in IBM Spectrum Scale Container Storage Interface driver.

High-level steps

The following procedure provides a high-level overview on migration of IBM Storage Enabler for Containers to IBM Spectrum Scale Container Storage Interface driver.

This is a manual process, which involves downtime till migration completes. For detailed steps, see "Detailed steps " on page 16.

1. Find all the PersistentVolumeClaim (PVCs) created in IBM Storage Enabler for Containers.
2. Create `pv.yaml` and `pvc.yaml` for each PV and PVC.
3. Change the reclaim policy of all PVs to *retain*.
4. Stop the workloads that are there in the IBM Storage Enabler for Containers.
5. Delete all PVs and PVCs created in IBM Storage Enabler for Containers except *ibm-ubiquity-db* or equivalent.
6. Uninstall IBM Storage Enabler for Containers.
7. Upgrade Kubernetes/Red Hat OpenShift/IBM Cloud® Private. For more information, see the respective documentation.
8. Deploy the IBM Spectrum Scale Container Storage Interface driver.
9. Create PVs and PVCs using the `yaml` files (pv.yaml and pvc.yaml).
10. Start the workload using the newly defined PVCs.

Detailed steps

This topic describes the detailed steps for migrating from IBM Storage Enabler for Containers to IBM Spectrum Scale Container Storage Interface driver.

1. Get all StorageClasses that are created in IBM Storage Enabler for Containers by issuing the following command:

   ```
   kubectl get sc | grep ubiquity/flex
   ```

2. Get all namespaces in IBM Storage Enabler for Containers by issuing the following command:

   ```
   kubectl get ns
   ```

3. For each namespace, get all PVCs for the Storage Classes by issuing the following command:

   ```
   kubectl get pvc -n <namespace> | grep <storageClass>
   ```

4. For each **PVC**:

   a. Collect the following details:

Table 3. PVC parameter details

Parameter Name	Description	Example syntax
pvname	Name of the PV to which the PVC is bounded	`kubectl get pvc <pvc name> -n <namespace> -o jsonpath={.spec.volumeName}`
pvmountpoint	File system mount point from PV definition	`kubectl get pv <pv-name> -o jsonpath={.spec.flexVolume.options.mountpoint}`
pvfsname	File system mount point from PV definition	`kubectl get pv <pv-name> -o jsonpath={.spec.flexVolume.options.filesystem}`
pvcapacity	Size of the PV from PV definition	`kubectl get pv <pv-name> -o jsonpath={.spec.capacity.storage}`
pvccapacity	Size of the PVC from the PVC definition	`kubectl get pvc <pvcname> -n <namespace> -o jsonpath={.spec.resources.requests.storage}`
pvcaccessmode	Get the PVC access mode from the PVC definition	`kubectl get pvc <pvcname> -n <namespace> -o jsonpath={.spec.accessModes[0]}`
pvaccessmode	Get the PVC access mode from the PV definition	`kubectl get pv <pvname> -n <namespace> -o jsonpath={.spec.accessModes[0]}`

   b. Generate `pvc.yaml` files as shown in the following example. Adjust the parameters according to your environment.

   ```
   apiVersion: v1
   kind: PersistentVolumeClaim
   metadata:
     name: <pvcname>
     namespace: <pvcnamespace>
   spec:
     accessModes:
     - <pvcaccessmode>
   ```

```
  resources:
    requests:
      storage: <pvccapacity>
  volumeName: <pvname>
```

Note: Update any other parameters such as **`storageClass`** based on your requirement in the `pvc.yaml` files. Ensure that appropriate modification is made to the dependent `pv.yaml` files.

c. Generate `pv.yaml` files using the `generate_pv_yaml.sh` tool. Adjust the parameters according to your environment.

   - If primary file system in IBM Storage Enabler for Containers is same as the volume file system (`$pvfsname`), use this command:

     ```
     ./generate_pv_yaml.sh  --filesystem $pvfsname --linkpath $pvmountpoint --size
     $pvcapacity --pvname $pvname --accessmode $pvaccessmode
     ```

   - If primary file system in IBM Storage Enabler for Containers is different from `$pvfsname`:

     1) Create a softlink in the primary file system area pointing to `$pvmountpoint` by issuing the following command:

        ```
        ln -s $pvmountpoint  <softlink path in primary filesystem>
        ```

     2) Generate the yaml file by issuing the following command:

        ```
        ./generate_pv_yaml.sh  --filesystem $pvfsname --linkpath <softlink path in primary
        filesystem> --size $pvcapacity --pvname $pvname --accessmode $pvaccessmode
        ```

   Note: Update or add any other parameter based on your requirement in the `pv.yaml` file.

d. Change the reclaim policy of PV to *retain*.

   ```
   kubectl patch pv <pv-name> -p '{‘spec"{"persistentVolumeReclaimPolicy":"Retain"}}’
   ```

5. Stop all the I/Os for the PVCs in IBM Storage Enabler for Containers.
6. Delete all PVCs except `ibm-ubiquity-db`.
7. Delete all PVs except `ibm-ubiquity-db`.
8. Uninstall IBM Storage Enabler for Containers.
9. Upgrade `Kubernetes/ICP/Openshift`.
10. Create all PVCs using the `pvc.yaml` file created in step 4 b.
11. Create all PVs using the `pv.yaml` created in step 4 c.
12. Start I/O using the newly created PVCs.

Chapter 7. Configuring IBM Spectrum Scale Container Storage Interface driver

This topic describes the options available for configuring IBM Spectrum Scale Container Storage Interface driver at your site.

IBM Spectrum Scale Container Storage Interface driver configurations

During IBM Spectrum Scale Container Storage Interface driver plug-in deployment, parameters required for communication with IBM Spectrum Scale must be configured in Kubernetes' Secrets.

Secrets

Secret is needed to store credentials to connect to IBM Spectrum Scale REST API server. Secrets are defined in data field with base64 encoded values in json file. The GUI user must have *csiadmin* role.

Perform the following steps:

1. Create secrets.yaml by issuing the following command:

```
apiVersion: v1
kind: Secret
metadata:
  name: [secret_name]
  labels:
    product: ibm-spectrum-scale-csi
data:
  username: [base64_username]
  password: [base64_password]
```

   Note: Credentials must be base64 encoded. For example, `echo -n 'my-password' | base64`.

2. Apply the secrets, as follows:

```
kubectl apply -f secrets.yaml -n ibm-spectrum-scale-csi-driver
```

Certificates

In case of secure SSL mode, a CA certificate must be specified. This is used in SSL communication with the IBM Spectrum Scale GUI server. The certificate should be created as a ConfigMap. There should be as many ConfigMaps as the number of clusters with secure SSL enabled.

For example,

```
kubectl create configmap cert1 --from-file=cert1=/path/to/mycertificate.pem -n ibm-spectrum-
scale-csi-driver
```

The value from "file=" should be the one that should be used as "cacert" value in CSIScaleOperator

Note: Configmap name and `--from-file` value should match and this value should be used as "`cacert`" value in Operator.

Operator

This topic describes the configuration parameters to be defined for creating a CSIScaleOperator custom resource that is used to configure IBM Spectrum Scale Container Storage Interface driver.

A sample configuration yaml file is available here: https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/operator/deploy/crds/csiscaleoperators.csi.ibm.com_cr.yaml.

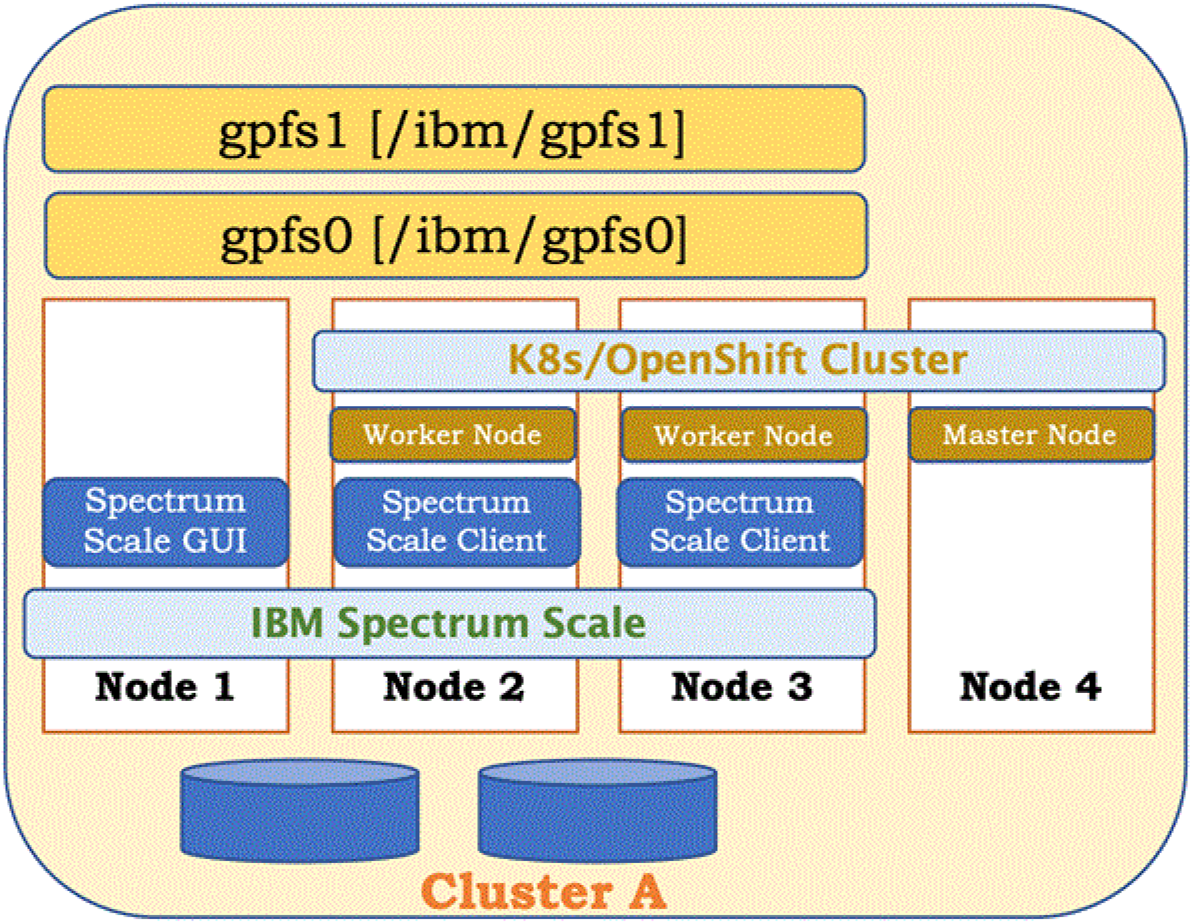

Figure 2. Operator configuration

- Primary cluster: IBM Spectrum Scale cluster where some or all of the client nodes are also worker nodes of Red Hat OpenShift/Kubernetes cluster. The aim of running IBM Spectrum Scale on worker node is to provide persistent storage from IBM Spectrum Scale to the application running on Kubernetes/Red Hat OpenShift.
- Primary file system: One of the existing IBM Spectrum Scale file systems from the primary cluster has to be designated as the primary file system. One fileset from this file system is used by the IBM Spectrum Scale Container Storage Interface driver internally to store the volume references. This fileset is referred to as primary fileset. For proper functioning of IBM Spectrum Scale Container Storage Interface driver, the primary file system must be mounted on all worker nodes all the time.

The CSIScaleOperator custom resource for a sample deployment looks like the following. There are two file systems **gpfs0** and **gpfs1**. For this deployment, we chose **gpfs0** as PrimaryFs.

csiscaleoperators.csi.ibm.com_cr.yam file

```
---
apiVersion: csi.ibm.com/v1
kind: "CSIScaleOperator"
metadata:
  name: "ibm-spectrum-scale-csi"
  namespace: "ibm-spectrum-scale-csi-driver"
  labels:
    app.kubernetes.io/name: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/instance: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/managed-by: ibm-spectrum-scale-csi-operator
    release: ibm-spectrum-scale-csi-operator
status: {}
spec:
  scaleHostpath: "/ibm/gpfs0"
  clusters:
    - id: "<cluster id of IBM Spectrum Scale running on node1,node2,node3>"
      secrets: "guisecretNode1"
      secureSslMode: false
      primary:
        primaryFs: "gpfs0"
      restApi:
        - guiHost: "<FQDN/IP of Node 1>"
  attacherNodeSelector:
    - key: "scale"
      value: "true"
  provisionerNodeSelector:
    - key: "scale"
      value: "true"
  pluginNodeSelector:
    - key: "scale"
      value: "true"
---
```

Table 4. CSIScaleOperator configuration parameter description

Parameter	Usage	Description
id	Mandatory	Cluster ID of the primary IBM Spectrum Scale cluster. For more information, see *mmlscluster command* in the *IBM Spectrum Scale: Concepts, Planning, and Installation Guide*.
primaryFs	Mandatory	Primary file system name.
primaryFset	Optional	Primary fileset name. This will be created if the fileset does not exist. Default value: **spectrum-scale-csi-volume-store**
inodeLimit	Optional	Inode limit for the primary fileset. If not specified, fileset is created with 1 M inodes, which is the IBM Spectrum Scale default.
cacert	Mandatory if *secureSslMode* is true.	Name of the pre-created CA certificate configmap that is used to connect to the GUI server (running on the "guiHost"). For more information, see "Certificates" on page 19.
secrets	Mandatory	Name of the pre-created Secret containing username and password that are used to connect to the GUI server for the cluster specificed against the **id** parameter. For more information, see "Secrets" on page 19.
guiHost	Mandatory	FQDN or IP address of the GUI node of IBM Spectrum Scale cluster that is specified against the **id** parameter.
scaleHostpath	Mandatory	Mount path of the primary file system (primaryFs).
imagePullSecrets	Optional	An array of imagePullSecrets to be used for pulling images from a private registry. This is a pass-through option that distributes the imagePullSecrets array to the containers generated by the Operator. For more information on creating imagePullSecrets, see https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/.

For deployment involving two or more IBM Spectrum Scale clusters, see "Remote cluster support" on page 21.

Remote cluster support

IBM Spectrum Scale provides a feature to mount IBM Spectrum Scale file systems from one IBM Spectrum Scale cluster (owning cluster) to another IBM Spectrum Scale cluster (accessing cluster). You can configure an IBM Spectrum Scale Container Storage Interface driver to work with a remotely mounted IBM Spectrum Scale.

The cluster that owns the file system is responsible for administering the file system and granting access to other clusters on a per-cluster basis. After access to a file system has been granted to nodes in another

IBM Spectrum Scale cluster, the nodes can mount the file system and perform data operations as if the file system are locally owned.

For more information on the remote mount setup, see *Accessing a remote GPFS file system* in the *IBM Spectrum Scale: Command and Programming Reference*.

Note: Remote mount setup must be done before configuring IBM Spectrum Scale Container Storage Interface driver.

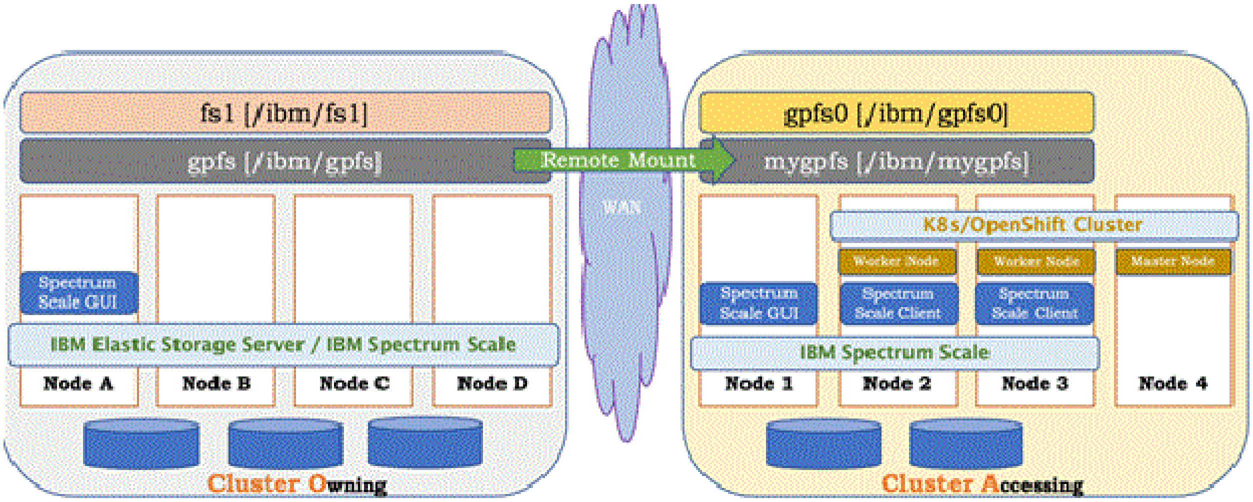

Figure 3. Deployment of two IBM Spectrum Scale clusters with remote-mounted file systems

Primary cluster is IBM Spectrum Scale cluster where Red Hat OpenShift/Kubernetes worker nodes coexist with IBM Spectrum Scale client nodes. In this example deployment, cluster A is designated as the primary cluster.

The cluster O is another IBM Spectrum Scale cluster having two file systems `gpfs` and `fs1`. The file system `gpfs` is mounted on Cluster A as file system `mygpfs` while file system `fs1` is not exposed to Cluster A.

For each IBM Spectrum Scale cluster, cluster entry must be added under the **clusters** section of the custom resource.

```
- id: "<cluster id of IBM Spectrum Scale>”
  restApi:
  - guiHost: "< FQDN or IP GUI node of IBM Spectrum Scale cluster>"
  secrets: "<secret name for GUI of IBM Spectrum Scale cluster>"
  secureSslMode: false
```

One IBM Spectrum Scale cluster has to be the primary cluster for IBM Spectrum Scale Container Storage Interface driver deployment. Primary cluster is marked by adding the **primary** section in the respective cluster entry. In the example described in the figure, deployment Cluster A is the primary cluster. Entry for primary cluster looks as follows:

```
- id: "<cluster id of IBM Spectrum Scale Cluster which is Primary cluster >”
     primary:
     primaryFs: <name of primary filesystem>
     restApi:
     - guiHost: "< FQDN or IP of Primary Spectrum Scale cluster’s GUI node>"
     secrets: "<secret name for GUI of Primary Spectrum Scale cluster >”
     secureSslMode: false
```

In the example deployment, there are two IBM Spectrum Scale clusters, hence two entries of clusters are added, one for the primary cluster (Cluster A) and another one for cluster O (Owning cluster).

The custom resource configuration slightly changes based on whether the primary file system is locally owned (`gpfs0` in the example deployment) or remotely mounted (`mygpfs` in the example deployment). The changes will be in the `primary` section of the primary cluster entry.

The custom resource for the example deployment when primaryFS is a locally owned file system (`gpfs0`) looks as follows:

```
---
apiVersion: csi.ibm.com/v1
kind: "CSIScaleOperator"
metadata:
  name: "ibm-spectrum-scale-csi"
  namespace: "ibm-spectrum-scale-csi-driver"
  labels:
    app.kubernetes.io/name: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/instance: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/managed-by: ibm-spectrum-scale-csi-operator
    release: ibm-spectrum-scale-csi-operator
status: {}
spec:
  scaleHostpath: "/ibm/gpfs0"
  clusters:
    - id: "<cluster id of IBM Spectrum Scale Cluster A>"
      secrets: "guisecretNode1"
      secureSslMode: false
      primary:
        primaryFs: "gpfs0"
      restApi:
        - guiHost: "<FQDN/IP of Node 1>"
    - id: "<cluster id of IBM Spectrum Scale Cluster O>"
      secrets: "guisecretNodeA"
      secureSslMode: false
      restApi:
        - guiHost: "<FQDN/IP of Node A>"
  attacherNodeSelector:
    - key: "scale"
      value: "true"
  provisionerNodeSelector:
    - key: "scale"
      value: "true"
  pluginNodeSelector:
    - key: "scale"
      value: "true"
---
```

The custom resource for example deployment when primaryFs is a remotely mounted file system (mygpfs) looks like this.

```
---
apiVersion: csi.ibm.com/v1
kind: "CSIScaleOperator"
metadata:
  name: "ibm-spectrum-scale-csi"
  namespace: "ibm-spectrum-scale-csi-driver"
  labels:
    app.kubernetes.io/name: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/instance: ibm-spectrum-scale-csi-operator
    app.kubernetes.io/managed-by: ibm-spectrum-scale-csi-operator
    release: ibm-spectrum-scale-csi-operator
status: {}
spec:
  scaleHostpath: "/ibm/mygpfs"
  clusters:
    - id: "<cluster id of IBM Spectrum Scale Cluster A>"
      secrets: "guisecretNode1"
      secureSslMode: false
      primary:
        primaryFs: "mygpfs"
    remoteCluster: "<cluster id of IBM Spectrum Scale Cluster O (Owning cluster)>"
      restApi:
        - guiHost: "<FQDN/IP of Node 1>"
    - id: "<cluster id of IBM Spectrum Scale Cluster O(owning cluster)>"
      secrets: "guisecretNodeA"
      secureSslMode: false
      restApi:
        - guiHost: "<FQDN/IP of Node A>"
  attacherNodeSelector:
    - key: "scale"
```

```
      value: "true"
  provisionerNodeSelector:
    - key: "scale"
      value: "true"
  pluginNodeSelector:
    - key: "scale"
      value: "true"
---
```

Table 5. Parameter description

Parameter name	Status	Parameter Description
id	Mandatory	Cluster ID of IBM Spectrum Scale cluster. For more information, see *mmlscluster* in the *IBM Spectrum Scale: Command and Programming Reference*.
primaryFs	Mandatory if cluster is primary	Name of the primary file system on the primary cluster.
primaryFset	Optional	Primary fileset name. Fileset of the specified name will be created if it does not exist. Default: `spectrum-scale-csi-volume-store`
remoteCluster	Mandatory if the primary file system (PrimaryFS) is a remotely mounted file system	Cluster ID of the remote cluster, who is the owner of the file system that is specified against **primaryFs**.
inodeLimit	Optional	Inode limit for the primary fileset. If not specified, fileset is created with 1 M inodes, which is the IBM Spectrum Scale default value.
cacert	Mandatory if *secureSslMode* is true.	Name of the pre-created CA certificate configmap that is used to connect to the GUI server that is running on the **guiHost**. For more information, see "Certificates" on page 19.
secrets	Mandatory	Name of the pre-created Secret containing username and password to connect to the GUI running on the guiHost for cluster specified against the **id** parameter. For more information, see "Secrets" on page 19.
guiHost	Mandatory	FQDN or IP address of the GUI node of IBM Spectrum Scale cluster specified against the **id** parameter.
scaleHostpath	Mandatory	Mount path of the primary the file system (**primaryFs**) on primary cluster.

Table 5. Parameter description (continued)

Parameter name	Status	Parameter Description
`imagePullSecrets`	Optional	An array of imagePullSecrets to be used for pulling images from a private registry. This is a pass-through option that distributes the imagePullSecrets array to the containers generated by the Operator. For more information on creating imagePullSecrets, see https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/.

Note:

- Owning cluster might have more than one file system and not all file systems need to be remotely mounted on the accessing cluster.
- There can be more than one owning cluster exposing their file systems to the accessing cluster.
- Accessing cluster/primary cluster can be compute-only cluster without any of its own file system.
- Secrets contain the credentials to connect to the GUI for a specified cluster. For each cluster in the custom resource, there should be a pre-created secret before Operator deployment. For more information, see "Secrets" on page 19. Same secret cannot be used for multiple clusters even if the credential are same.
- Custom resource also contains other parameters that are optional, so those parameter should be added as per your requirement.

Using the node selector

By default, the IBM Spectrum Scale Container Storage Interface driver gets deployed on all worker nodes. Node selector controls on which Kubernetes worker nodes IBM Spectrum Scale Container Storage Interface driver should be running. It helps in cases where new worker nodes are added to Kubernetes cluster but do not have IBM Spectrum Scale installed. It helps in ensuring that StatefulSets are running on the desired nodes.

To configure node selector, perform the following steps:

1. Label the Kubernetes worker nodes where StatefulSets should run, as shown in the following example:

   -
     ```
     kubectl label node node1 infranode=1 --overwrite=true
     ```

   -
     ```
     kubectl label node node2 infranode=2 --overwrite=true
     ```

   Note:

   - Use specific labels like the one for attacher and provisioner StatefulSet, only if there is a requirement of running these StatefulSets of very specific nodes. Otherwise, use single label like `scale=true` for running StatefulSets and IBM Spectrum Scale Container Storage Interface driver DaemonSet.
   - Nodes marked for running StatefulSet must be subset of the nodes marked with the `scale=true` label.

2. Label the Kubernetes worker nodes where IBM Spectrum Scale Container Storage Interface driver should run, as follows:

   ```
   kubectl label node node1 scale=true --overwrite=true
   ```

3. Configure the following parameters in the Operator custom resource (`ibm-spectrum-scale-csi-operator-cr.yaml`) under the "spec" section: IBM Spectrum Scale client must be installed and running on the nodes that have the `scale=true` label.

```
attacherNodeSelector:
   - key: "scale"
     value: "true"
#  - key: "infranode"       # Only if there is requirement of running Attacher
#    value: "2"             # on specific Node

provisionerNodeSelector:
   - key: "scale"
     value: "true"
#  - key: "infranode".      # Only if there is requirement of running Provisioner
#    value: "1"             # on specific Node

pluginNodeSelector:
   - key: "scale"
     value: "true"
```

Note: If you choose to run IBM Spectrum Scale Container Storage Interface driver on selective nodes using the nodeSelector, then make sure that the pod using IBM Spectrum Scale Container Storage Interface driver PVC is getting scheduled on the nodes where IBM Spectrum Scale Container Storage Interface driver is running.

Kubernetes to IBM Spectrum Scale node mapping
In some environments, Kubernetes node names might be different from the IBM Spectrum Scale node names. This results in failure during mounting of pods. Kubernetes node to IBM Spectrum Scale node mapping must be configured to address this condition during the Operator configuration.

To configure this, add "`nodeMapping`" section under "spec" in the `csiscaleoperators.csi.ibm.com_cr.yaml`, as shown in the following example:

```
nodeMapping:
   - k8sNode: "kubernetesNode1"
     spectrumscaleNode: "scaleNode1"
   - k8sNode: "kubernetesNode2"
     spectrumscaleNode: "scaleNode2"
```

If Kubernetes node name starts with a number, then add node mapping for such nodes in this format:

```
    - k8sNode: "K8sNodePrefix_<Kubernetes Node Name/ID>"
      spectrumscaleNode: "<Spectrum Scale Node Name/ID>"
```

For example, if Kubernetes node name is 198.51.100.10, then use the following node mapping:

```
    - k8sNode: "K8sNodePrefix_198.51.100.10"
      spectrumscaleNode: "spectrumscalenode11"
```

Note:

- Kubernetes node name is listed by using the `kubectl get nodes` command.
- IBM Spectrum Scale node name is listed by the **`mmlscluster`** command.
- All entries for nodes that differ in name must be added.

Storage class

Storage class is used for creating lightweight volumes as well as fileset-based volumes.

Storage Class for creating lightweight volumes:

This is the configuration:

```
apiVersion: storage.k8s.io/v1
```

```
kind: StorageClass
metadata:
   name: ibm-spectrum-scale-csi-lt
provisioner: spectrumscale.csi.ibm.com
parameters:
    volBackendFs: "gpfs0"
    volDirBasePath: "pvfileset/lwdir"
reclaimPolicy: Delete
```

Description of the fields

Field name	Description
volBackendFs	File system on which directory-based volume should be created.
volDirBasePath	Base path under which all volumes with this storage class will be created. This path must exist. Relative path from the file system mount point.
uid	uid/username that should be assigned to the directory. This is optional. The uid/gid must exist on the IBM Spectrum Scale GUI node.
gid	gid/groupname that should be assigned to the directory. This is optional. The uid/gid must exist on the IBM Spectrum Scale GUI node.

Storage class for creating fileset based volumes (Local file system):

If you want to create PVCs under a file system that is owned by the primary cluster, then use the storageClass details provided in the following example:

In this example, it is assumed that the user wants to create PVCs under the file system, **gpfs0**, as per the sample deployment explained in "Remote cluster support" on page 21. The same storageClass format is applicable for sample deployment explained in "Operator" on page 19.

Independent fileset storage class:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: ibm-spectrum-scale-csi-fileset
provisioner: spectrumscale.csi.ibm.com
parameters:
    volBackendFs: gpfs0
    clusterId: "17797813605352210071"
    uid: "1000"
    gid: "1000"
reclaimPolicy: Delete
```

Dependent fileset storage class example:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: ibm-spectrum-scale-csi-fileset-dependent
provisioner: spectrumscale.csi.ibm.com
parameters:
    volBackendFs: "gpfs0"
    clusterId: "17797813605352210071"
    uid: "1000"
    gid: "1000"
    filesetType: "dependent"
    parentFileset: "independent-fileset-fset1"
reclaimPolicy: Delete
```

Note: "17797813605352210071" is the cluster ID of the primary cluster.

Description of the fields

Field name	Description
volBackendFs	Name of the file system under which the fileset should be created.
clusterId	Cluster ID of the primary cluster. This is optional. Default value is the cluster ID of the primary cluster.
uid	uid/username that will be assigned to the fileset. This is optional. The uid/gid must exist on the IBM Spectrum Scale GUI node. Default value is "root".
gid	gid/group name that will be assigned to the fileset. This is optional. The gid/group name must exist on the IBM Spectrum Scale GUI node. Default value is "root".
filesetType	Default value is independent.
parentFileset	Parent fileset name. Valid with filesetType=dependent. Default is root.
inodeLimit	inode limit for fileset. This is optional. Valid with filesetType=independent. Default is 1 million.

StorageClass for creating fileset-based volumes (Remote file system)

If you want to create PVCs under a file system that is owned by a different cluster other than the primary cluster, then use the storageClass details provided in this example. In this example, it is assumed that the user wants to create PVCs under the file system, **gpfs0**, as per the sample deployment explained in "Remote cluster support" on page 21. The same storageClass format is applicable for sample deployment explained in "Operator" on page 19.

Independent fileset storageClass for the file system

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: ibm-spectrum-scale-csi-fileset
provisioner: spectrumscale.csi.ibm.com
parameters:
    volBackendFs: "mygpfs"
    clusterId: "7118073361626808055"
    uid: "1000"
    gid: "1000"
reclaimPolicy: Delete
```

Dependent fileset storageClass example

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: ibm-spectrum-scale-csi-fileset-dependent
provisioner: spectrumscale.csi.ibm.com
parameters:
    volBackendFs: "mygpfs"
    clusterId: " 7118073361626808055"
    uid: "1000"
    gid: "1000"
    filesetType: "dependent"
     parentFileset: "independent-fileset-fset1"
reclaimPolicy: Delete
```

Note: Here, 7118073361626808055 represents the cluster ID of the owning cluster(Cluster O).

Description of the fields

Field name	Description
volBackendFs	Name of the file system under which the fileset should be created. File system name is the name of the remotely mounted file system on the primary cluster.
clusterId	Cluster ID of the owning cluster.
uid	uid/username that will be assigned to the fileset. This is optional. The uid/gid must exist on the IBM Spectrum Scale GUI node of accessing and owning clusters. Default value is "root".
gid	gid/group name that will be assigned to the fileset. This is optional. The gid/group name must exist on the IBM Spectrum Scale GUI node.of the accessing and owning clusters. Default value is "root".
filesetType	Default is independent. This is optional.
parentFileset	Parent fileset name. Valid with filesetType=dependent. Default is root.
inodeLimit	inode limit for fileset. This is optional. Valid with filesetType=independent. Default is 1 million.

Changing the configuration after deployment

IBM Spectrum Scale Container Storage Interface driver configuration can be changed after the driver is deployed. Any change in the configuration post deployment will re-initialize IBM Spectrum Scale Container Storage Interface driver.

Updating a Secret

The IBM Spectrum Scale Container Storage Interface driver leverages secrets to store API authentication. In the event of an authentication going stale, the user will need to update the secret in Kubernetes.

Note: Due to `ansible-operator` constraints, when updating a secret, `kubectl apply` and `kubectl edit` commands are not usable.

To update the secret and have the operator apply it, do the following steps:

1. Edit the `json` or `yaml` defining your secret to have the updated authentication information.

   ```
   export SECRET_NAME="mysecret"
   export NAMESPACE="ibm-spectrum-scale-csi-driver"
   ```

   Note: If you still have a json or yaml file, you can just edit that:

   ```
   kubectl get secret -n ${NAMESPACE} ${SECRET_NAME} -o yaml > secret.yaml
   ```

2. Ensure the secret has the correct labelling.

   ```
       metadata:
           labels:
               app.kubernetes.io/name: ibm-spectrum-scale-csi-operator
   ```

   Note: If the label is not set, the Operator will not trigger.

3. Delete the old secret and apply the updated secret configuration.

```
kubectl delete secret -n ${NAMESPACE} ${SECRET_NAME}
kubectl apply -f secret.yaml
```

After running the fresh apply, you should see the *spec.trigger* field increment if the secret was successfully created. The process may then be monitored in the Operator logs.

Additionally, if the Operator's custom resource was deployed before the secrets were created the above process may be leveraged to start the operator without deleting the Custom Resource.

Cluster Details

To change cluster details such as *guiHost*, remote cluster information or node mapping, edit the **CSIScaleOperator** using the following command.

```
kubectl edit CSIScaleOperator ibm-spectrum-scale-csi  -n  ibm-spectrum-scale-csi-driver
```

When this command is executed, a `vi` editor opens up, which contains a temporary `yaml` file with the contents for *CSIScaleOperator* object. Here, the user can update the cluster details, save the file, and exit. The Operator will restart the IBM Spectrum Scale Container Storage Interface driver with the new configuration.

Chapter 8. Using IBM Spectrum Scale Container Storage Interface driver

You can create storage volumes such as PVCs and PVs to suit your requirements.

Dynamic provisioning

Administrators use dynamic volume provisioning to create storage volumes on-demand.

Do the following steps:

1. Create a storageClass for the type of the volume to be created. For more information, see "Storage class" on page 26.
2. Apply the configuration:

   ```
   kubectl apply -f storageclass.yaml
   ```

3. Create a persistent volume claim (PVC) using this storageclass, as shown in the following example:

   ```
   # cat pvc.yaml

   apiVersion: v1
   kind: PersistentVolumeClaim
   metadata:
     name: scale-fset-pvc
   spec:
     accessModes:
     - ReadWriteMany
     resources:
       requests:
         storage: 1Gi
     storageClassName: [name_of_your_storageclass]
   ```

   Modify the PVC name, storage, and storageClassName values according to your requirement.

4. Create a PVC by issuing this command:

   ```
   kubectl apply -f pvc.yaml
   ```

Static provisioning

In static provisioning, an administrator creates a number of Persistent Volumes (PVs), which include information about the storage that is available to each user in the cluster.

To create a storage class for static provisioning, do the following steps:

1. Create a persistent volume using the PV manifest file. For more information, see "Creating a persistent volume (PV)" on page 32.
2. Create a persistent volume claim (PVC) using the PVC manifest file. For more information, see "Creating a PersistentVolumeClaim (PVC)" on page 33.

Generating a PV manifest

To generate a PV manifest, use the following script:

```
generate_pv_yaml.sh
```

You can download and run this sample script from the following location:

```
https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/tools/generate_pv_yaml.sh
```

Note: This script should be run on an IBM Spectrum Scale cluster node.

Usage of the script is as follows:

```
Usage: ./generate_pv_yaml.sh
        -f|--filesystem <Name of Volume's Source Filesystem>
        -l|--linkpath <full Path of Volume in Primary Filesystem>
        -s|--size <size in GB>
        [-p|--pvname <name for pv>]
              [-c|--storageclass <StorageClass for pv>]
              [-a|--accessmode <AccessMode for pv>]
        [-h|--help]
```

Example 1: Single file system

```
./generate_pv_yaml.sh --filesystem gpfs0 --linkpath /ibm/gpfs0/fileset1/\
.volumes/staticpv --size 10 --pvname mystaticpv
```

Here, only one file system 'gpfs0' and the directory from the same file system are used as volume.

Example 2: Two or more file systems

```

./generate_pv_yaml.sh --filesystem gpfs1 --linkpath /ibm/gpfs0/fileset1/\
.volumes/staticpv1 --size 10 --pvname mystaticpv1
```

Here, two file systems 'gpfs0' and 'gpfs1' are used, where 'gpfs0' is configured as primary file system. To create a volume from the directory present in the gpfs1 file system (say a directory in the gpfs1 is /ibm/gpfs1/dir1), you need to create a softlink `/ibm/gpfs1/dir1 --> /ibm/gpfs0/fileset1/.volumes/staticpv1` and then issue the following command to generate the `pv.yaml` file:

```
./generate_pv_yaml.sh --filesystem gpfs1 --linkpath /ibm/gpfs0/fileset1/.volumes/staticpv1\
--size 10 --pvname mystaticpv1
```

Note: The script does not validate if softlinks are correctly created. The path that is specified for the `--linkpath` option must be a valid GPFS path from the primary file system.

Creating a persistent volume (PV)

A PersistentVolume (PV) is a storage that is statically provisioned by an administrator or dynamically provisioned using storage classes.

To create a PV, do the following steps:

1. Configure the https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/driver/examples/static/static_pv.yaml file with the required parameters.
2. Configure persistent volume (PV) manifest file with a volumeHandle as described in this example:

```
apiVersion: v1
kind: PersistentVolume
metadata:
        name: static-scale-static-pv
spec:
  capacity:
    storage: 1Gi
  accessModes:
    - ReadWriteMany
  csi:
    driver: spectrumscale.csi.ibm.com
    volumeHandle: 7118073361626808055;09762E69:5D36FE8D;path=/ibm/gpfs0/staticdir
```

Field Name	Description
volumeHandle	This must be in the [clusterID];[Filesystem UID];path=[Path to the directory] format
clusterID	ID of the primary cluster

Field Name	Description
	Note: **`mmlscluster`** command displays the current configuration including the cluster ID.
Filesystem UID	This is the UID of the filesystem consisting of directory to be provisioned **Note:** `mmlsfs <filesystem name> --uid` command displays the file system UID.
path	Complete path of the directory to be provisioned.

Note: This manifest file can be auto-generated using the `generate_pv_yaml.sh` tool.

For more information, see https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/tools/generate_pv_yaml.sh.

3. Issue this command to create a PV:

```
kubectl apply -f pv.yaml
```

Creating a PersistentVolumeClaim (PVC)

A PVC is a request for storage by a user. There are two types of PVCs, static provisioning and dynamic provisioning.

Create a PVC manifest as follows:

1. Create a `pvc.yaml` file:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: scale-static-pvc
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
```

2. Create a PVC by issuing the following command:

```
kubectl apply -f pvc.yaml
```

This PVC is bound to an available PV with storage equal to or greater than what is specified in the `pvc.yaml` file.

Creating pods

To configure a pod, do the following steps:

1. Create a manifest file (pod.yaml) with pod definition referencing the persistent volume claim (PVC). Following is an example of a pod definition for creating a nginx container using a previously created PVC:

```
# cat pod.yaml

apiVersion: v1
kind: Pod
metadata:
  name: csi-scale-staticdemo-pod
  labels:
    app: nginx
spec:
  containers:
   - name: web-server
```

```
     image: nginx
     volumeMounts:
       - name: mypvc
         mountPath: /usr/share/nginx/html/scale
     ports:
     - containerPort: 80
  volumes:
   - name: mypvc
     persistentVolumeClaim:
       claimName: [pvc name]
       readOnly: false
```

Note: `claimName` is the PVC name to be used by pod for persistent storage. The readOnly flag can be set to true in which case the pod will mount the PVC in the read-only mode.

2. Issue the following command to create the pod:

```
kubectl apply -f pod.yaml
```

For more information on pods, see Configure a Pod to Use a PersistentVolume for Storage.

Considerations for mounting read-write many (RWX) volumes

Red Hat OpenShift relabels the volumes while mounting them inside a pod. In case of volumes with RWX access mode, when multiple pods mount them, they are relabeled multiple times. If the SELinux labeling is not managed, it might cause data access issues from the pod.

You can address this issue in either of the following ways:

- Use an SCC (Security Context Constraints) with `seLinuxContext.type` set as "MustRunAs".

  ```
  seLinuxContext:
       type: MustRunAs
  ```

  Ensure that correct SCC is used by the pod:

  ```
  metadata:
       annotations:
         openshift.io/scc: <scc_name>
  ```

- Specify appropriate `seLinuxOptions.level` in the deployment specification of the pod as shown in the following example:

  ```
  securityContext:
      seLinuxOptions:
        level: <SELinux level label>
  ```

  An example of SELinux level label is "s0:c123,c456"

Chapter 9. Managing IBM Spectrum Scale when used with IBM Spectrum Scale Container Storage Interface driver

When IBM Spectrum Scale is used for providing persistent volumes for containers, then the following must be considered:

Adding a new node to the Kubernetes or Red Hat OpenShift cluster

This topic describes the procedure for adding a node to the Kubernetes or Red Hat OpenShift cluster.

Do the following steps:

1. Add the node to the IBM Spectrum Scale cluster. For more information, see *mmaddnode command* in the *IBM Spectrum Scale: Command and Programming Reference*.
2. Mount the required file systems on the newly added node.
3. Add the new node into the Kubernetes cluster. For more information, see the Kubernetes documentation.
4. Add a label to the node. For example, `scale = true`
5. Check whether the IBM Spectrum Scale Container Storage Interface driver pods are running correctly.

Unmounting IBM Spectrum Scale file system

Follow these steps to unmount IBM Spectrum Scale file systems from a node:

1. Move the containers that are using the file system, which is being unmounted, to other nodes.
2. Ensure that the new pods that are using the IBM Spectrum Scale file systems, which is being unmounted, are not scheduled on the node.
3. Unmount the IBM Spectrum Scale file system using the **mmunmount** command.

For more information on the **mmunmount** command, see *mmunmount command* in the *IBM Spectrum Scale: Command and Programming Reference* Guide.

If you face an issue in unmounting the IBM Spectrum Scale file systems, see *File system fails to unmount* topic in the *IBM Spectrum Scale: Problem Determination Guide*.

Shutting down IBM Spectrum Scale

Follow these steps to shut down IBM Spectrum Scale when using IBM Spectrum Scale Container Storage Interface driver:

1. Move the containers that are using the file system, which is being unmounted, to other nodes.
2. Ensure that the new pods that are using the IBM Spectrum Scale file systems that are being unmounted are not scheduled on the node.
3. Stop Kubernetes and Docker.
4. Shutdown the IBM Spectrum Scale file system using the **mmshutdown** command.

Note: Stop all pods manually before running the **mmshutdown** command. Otherwise, a worker node might crash. If a crash occurs, its recovery involves recovery of the node, followed by manually stopping all pods before resuming any prior shutdown.

For more information on the **mmshutdown** command, see *mmshutdown command* in the *IBM Spectrum Scale: Command and Programming Reference* Guide.

IBM Spectrum Scale monitoring considerations

Consider the following information for IBM Spectrum Scale when using IBM Spectrum Scale Container Storage Interface driver:

- If an IBM Spectrum Scale file system that is being used by Kubernetes gets unmounted, or if there is an issue with the IBM Spectrum Scale file system mounted on a particular node, then the applications in the containers that are using the PVC from IBM Spectrum Scale throw an I/O error.
- It is recommended that users directly monitor IBM Spectrum Scale for any IBM Spectrum Scale specific issues, since such monitoring is not done by Kubernetes or IBM Spectrum Scale Container Storage Interface driver.

Upgrading IBM Spectrum Scale on IBM Spectrum Scale Container Storage Interface driver nodes

IBM Spectrum Scale can be upgraded on the nodes where IBM Spectrum Scale Container Storage Interface driver is already running.

On the worker nodes

As a first of step of upgrade, upgrade IBM Spectrum Scale on the worker nodes. Perform the following steps to upgrade IBM Spectrum Scale on worker nodes:

1. Cordon the worker node so that scheduling is disabled.

   ```
   kubectl cordon <node>
   ```
2. Move any workload off the worker node.
3. Remove the `scale` label from the node.

   ```
   kubectl label <node> scale-
   ```
4. Prepare the worker node to shut down IBM Spectrum Scale.

   a. Check for open files.

   ```
   lsof <filesystem>
   ```

   b. Check for mounted kernel modules.

   ```
   lsmod | grep mm
   ```

   c. If there is any mm* present, then unmount and shut down file systems on the worker node. You can use the following options to view the details of the mounted file systems:

   - Check the GPFS status:

     ```
     mmgetstate
     ```

     Note: The GPFS state should be active.
   - List the mounted file systems across all nodes by using the following command:

     ```
     mmlsmount all
     ```
   - List disk space usage and more importantly to see what file systems are mounted by using the **df** command:

     ```
     df
     ```

   d. Unmount all file systems for the current node.

   ```
   mmunmount all
   ```

e. Shut down GPFS on the current node.

```
mmshutdown
```

f. Check for open files.

```
lsof <file system>
```

g. Check for mounted kernel modules.

```
lsmod | grep mm
```

Note:

- All file systems must be unmounted, and GPFS must be shut down. Continue to step 5 to proceed with IBM Spectrum Scale upgrade.
- If there are any file systems or mounted kernel modules (mm*) present, then you require a reboot of the worker node to clean up the state. Ensure **autoload** is set to `off` for the node before rebooting.

h. Set **autoload** to `off`.

```
mmchconfig autoload=no -N <node>
```

i. Reboot the worker node.

```
reboot
```

5. Upgrade the IBM Spectrum Scale by using the toolkit, set the worker node as an offline node, exclude the other nodes.
6. After the upgrade is completed, do the following steps:

a. Log on to the worker node and ensure that **autoload** is set back to on.

```
mmchconfig autoload=yes -N <node>
```

b. Log on to the worker node and start GPFS.

```
mmstartup
```

c. Uncordon the worker node.

```
kubectl uncordon <node>
```

d. Relabel the node `scale`.

```
kubectl label node <node> scale=true
```

On the nodes running provisioner and attacher pods

As the next step, you must upgrade IBM Spectrum Scale on the nodes where provisioner and attacher pods are running.

1. Move or stop all pods that use volumes that are managed by the IBM Spectrum Scale Container Storage Interface driver.
2. Drain the nodes so that `StatefulSets` move to other nodes.

```
kubectl drain <nodename> --ignore-daemonsets –delete-local-data
```

3. Remove the `PluginSelector` label that is assigned to the infrastructure node.

```
kubectl label node <nodename> scale-
```

4. Prepare the node to shut down IBM Spectrum Scale.

a. Check for open files.

```
lsof <filesystem>
```

b. Check for mounted kernel modules.

```
lsmod | grep mm
```

c. If there is any mm* present, then unmount and shut down file systems on the node.

- Ensure that GPFS is active on the node.

  ```
  mmgetstate
  ```

- To list the mounted file systems across all nodes.

  ```
  mmlsmount all
  ```

- To list disk space usage and more importantly to see what file systems are mounted.

  ```
  df
  ```

d. Unmount all file systems for the current node.

```
mmunmount all
```

e. Shut down GPFS on the current node.

```
mmshutdown
```

f. Check for open files.

```
lsof <filesystem>
```

g. Check for mounted kernel modules.

```
lsmod | grep mm
```

Note:

- If all file systems are unmounted and GPFS is shut down, continue to step 5 to proceed with IBM Spectrum Scale upgrade.
- If there are any file systems or mounted kernel modules (mm*) present, then do a reboot of the worker node to clean up the state. Ensure that **autoload** is set to `off` for the node before rebooting.

h. Disable **autoload** by issuing the following command:

```
mmchconfig autoload=no -N <node>
```

i. Reboot the worker node.

```
reboot
```

5. Upgrade the IBM Spectrum Scale using the IBM Spectrum Scale installation toolkit, set the worker node as an offline node, and exclude other nodes.
6. After the upgrade is completed, do the following steps:

a. Log on to the worker node and issue the following command to enable autoload:

```
mmchconfig autoload=yes -N <node>
```

b. Log on to the worker node and start GPFS.

```
mmstartup
```

c. Uncordon the node.

```
kubectl uncordon <node>
```

d. Relabel the node with IBM Spectrum Scale Container Storage Interface driver.

```
kubectl label node <node> scale=true
```

Chapter 10. Limitations

The following are the known limitations of the IBM Spectrum Scale Container Storage Interface driver:

- IBM Spectrum Scale Container Storage Interface driver does not honor the size specified in `PersistentVolumeClaim` for lightweight volume. The **df** command inside a container does not show the correct volume size in the container for lightweight Volumes. Instead, this command shows the size of the file system or fileset based on the `filesetdf` configuration.
- Maximum number of supported volumes that can be created using independent fileset storage class is 998, excluding the root fileset and primary fileset reserved for IBM Spectrum Scale Container Storage Interface driver. This limitation is based upon the number of fileset that are supported by IBM Spectrum Scale. For more information, see https://www.ibm.com/support/knowledgecenter/STXKQY/gpfsclustersfaq.html.
- IBM Spectrum Scale Container Storage Interface driver relies on the GUI server for performing IBM Spectrum Scale operations during volume provisioning or deprovisioning and attach or detach. If the GUI password or CA certificate expires, then manual intervention is needed by the admin to reset the password on GUI or generate a new certificate and update the configuration in IBM Spectrum Scale Container Storage Interface driver.
- IBM Spectrum Scale Container Storage Interface driver does not support ReadOnlyMany access mode of PVC.
- IBM Spectrum Scale file systems should remain mounted on worker nodes while IBM Spectrum Scale Container Storage Interface driver is running. If unmounted for some reason, it might affect pods on a node, and a reboot of the node is required to mount the file system back.
- Volume provisioning and attachment operations rely on REST API status. Occasionally, there may be some delay in reflecting the file system status from the cluster into the REST API. In such instances, you may experience interim failures in volume provisioning and attach or detach operations, which will go away once the REST API status is updated.
- Although multiple instances of the IBM Spectrum Scale GUI are allowed, the IBM Spectrum Scale Container Storage Interface driver is currently limited to point to a single GUI node.
- We do not support the rolling upgrade of IBM Spectrum Scale while IBM Spectrum Scale Container Storage Interface driver/Operator pods are running on the worker nodes.
- On IBM Spectrum Scale 5.0.4.1, volume (PVC) creation fails for file systems with POSIX ACL.
- IBM Spectrum Scale Container Storage Interface driver is supported on Red Hat OpenShift with RHEL worker nodes only. There is no support for CoreOS worker nodes.

Chapter 11. Troubleshooting

If there is any issue with IBM Spectrum Scale Container Storage Interface driver functionality, you must obtain the logs, which can be done by running the `spectrum-scale-driver-snap.sh` tool. These logs along with the output of the **`gpfs.snap`** command can be used for debugging the issue.

Debug data collection

IBM Spectrum Scale Container Storage Interface driver provides the `spectrum-scale-driver-snap.sh` tool to collect the debug data. This tool gathers the state of required Kubernetes resources like nodes, pods, service accounts, and so on and collects `StatefulSet` and `DaemonSet` logs from all nodes. It collects definition of resources in the given namespace with the label, `product=ibm-spectrum-scale-csi`. The collected logs are stored in the given output directory.

Download the tool from this location:

```
https://raw.githubusercontent.com/IBM/ibm-spectrum-scale-csi/v2.0.0/tools/spectrum-scale-driver-
snap.sh
```

Usage of the tool

```
spectrum-scale-driver-snap.sh [-n namespace] [-o output-dir] [-h]
```

```

-n: Debug data for CSI resources under this namespace will be collected. If not specified,
default namespace is used. The tool returns error if CSI is not running under the given
namespace.
-o: Output directory where debug data will be stored. If not specified, the debug data is
stored in current directory.
-h: Prints the usage
```

The resultant folder contains the following files with debug information:

- `nodes.json`
- `ibm-spectrum-scale-csi-k8snodes`
- `ibm-spectrum-scale-csi-configmap`
- `ibm-spectrum-scale-csi-get-all-by-label`
- `ibm-spectrum-scale-csi-describe-all-by-label`
- `ibm-spectrum-scale-csi-operator-XXX-XXXXX-operator-previous.log`
- `ibm-spectrum-scale-csi-operator-XXX-XXXXX-ansible-previous.log`
- `ibm-spectrum-scale-csi-operator-XXX-XXXXX-operator.log`
- `ibm-spectrum-scale-csi-operator-XXX-XXXXX-ansible.log`
- `ibm-spectrum-scale-csi-describe-CSIScaleOperator`
- `ibm-spectrum-scale-csi-XXXXX-driver-registrar-previous.log`
- `ibm-spectrum-scale-csi-XXXXX-previous.log`
- `ibm-spectrum-scale-csi-XXXXX-driver-registrar.log`
- `ibm-spectrum-scale-csi-XXXXX.log`
- `ibm-spectrum-scale-csi-XXXXX-driver-registrar-previous.log`
- `ibm-spectrum-scale-csi-XXXXX-previous.log`
- `ibm-spectrum-scale-csi-XXXXX-driver-registrar.log`
- `ibm-spectrum-scale-csi-XXXXX.log`

- `ibm-spectrum-scale-csi-provisioner.log`
- `ibm-spectrum-scale-csi-attacher.log`
- `ibm-spectrum-scale-csi-k8snodes`
- `ibm-spectrum-scale-csi-configmap`
- `ibm-spectrum-scale-csi-get-all-by-label`
- `ibm-spectrum-scale-csi-describe-all-by-label`
- `ibm-spectrum-scale-csi-attacher.log`
- `ibm-spectrum-scale-csi-provisioner.log`
- `ibm-spectrum-scale-csi-xxxxx-driver-registrar.log`
- `ibm-spectrum-scale-csi-xxxxx.log`

Debugging initialization issues

This section outlines how to debug IBM Spectrum Scale Container Storage Interface driver initialization issues.

Issue: IBM Spectrum Scale Container Storage Interface driver pod goes in an error state during deployment

```
# kubectl get pod -n ibm-spectrum-scale-csi-driver

NAME                                       READY   STATUS             RESTARTS   AGE
ibm-spectrum-scale-csi-attacher-0          1/1     Running            0          4d2h
ibm-spectrum-scale-csi-dxslh               0/2     Error              20         4d2h
ibm-spectrum-scale-csi-provisioner-0       1/1     Running            0.         4d2h
```

How to troubleshoot this?

Look for the IBM Spectrum Scale Container Storage Interface driver container logs in the IBM Spectrum Scale Container Storage Interface driver pod as shown in the following example, where you can see the root cause of the failure.

```
# kubectl logs ibm-spectrum-scale-csi-dxslh ibm-spectrum-scale-csi

E1017 06:13:07.529701       1 gpfs.go:252] Error getting cluster ID: rpc error: code =
Unauthenticated desc = Unauthorized GET request to https://ip-10-0-73-
79.ec2.internal:443/scalemgmt/v2/cluster: 401 Unauthorized
E1017 06:13:07.529711       1 gpfs.go:196] Error in plugin initialization: rpc error: code =
Unauthenticated desc = Unauthorized GET request to https://ip-10-0-73-
79.ec2.internal:443/scalemgmt/v2/cluster: 401 Unauthorized
F1017 06:13:07.529719       1 main.go:65] Failed to initialize Scale CSI Driver: rpc error:
code
= Unauthenticated desc = Unauthorized GET request to https://ip-10-0-73-
79.ec2.internal:443/scalemgmt/v2/cluster: 401 Unauthorized
```

The logs indicate that the credentials that are used to fetch the cluster ID from IBM Spectrum Scale GUI are incorrect. To fix the problem, ensure that the secrets used during deployment are correct. For more information, see "Changing the configuration after deployment" on page 29.

Debugging PVC creation issues

This section discusses the troubleshooting of PVC creation issues.

Issue: PVC remains in the pending state

```
# kubectl get pvc  scale-fset-pvc
NAME             STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
scale-fset-pvc   Pending                                      ibm-spectrum-scale-csi-fileset
9s
```

How to troubleshoot?

Look for the PVC description. It should highlight any error prohibiting the volume creation, as shown in the following example:

```
# kubectl describe pvc scale-fset-pvc
Name:          scale-fset-pvc
Namespace:     ibm-spectrum-scale-csi-driver
StorageClass:  ibm-spectrum-scale-csi-fileset
Status:        Pending
Volume:
Labels:        <none>
Annotations:   volume.beta.kubernetes.io/storage-provisioner: spectrumscale.csi.ibm.com
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Events:
  Type       Reason                Age
From
     Message
  ----       ------                ----
----
     -------
  Normal     Provisioning          11s                spectrumscale.csi.ibm.com_ibm-spectrum-
scale-csi-provisioner-0_c58323a3-436a-11ea-9c1a-920ed99f44ce  External provisioner is
provisioning volume for claim "ibm-spectrum-scale-csi-driver/scale-fset-pvc"
  Warning    ProvisioningFailed    10s                spectrumscale.csi.ibm.com_ibm-spectrum-
scale-csi-provisioner-0_c58323a3-436a-11ea-9c1a-920ed99f44ce  failed to provision volume with
StorageClass "ibm-spectrum-scale-csi-fileset": rpc error: code = Internal desc = Unable to
create fileset [pvc-4696a3e4-5006-11ea-8b62-000c2932e5ce] in FS [scale0]. Error [[EFSSG0072C
File set myscalefileset does not exist.]]
  Normal     ExternalProvisioning  10s (x2 over 10s)  persistentvolume-
controller                                                                          waiting for
a volume to be created, either by external provisioner "spectrumscale.csi.ibm.com" or manually
created by system administrator
Mounted By:  <none>
```

Debugging pod mounting issues

This section discusses the troubleshooting of issues related to pod mounting.

Issue

Application pod fails to start and does not go in the Running state

How to troubleshoot?

Look for pod description for the root cause of failure.

```
# kubectl describe pod my-csi-pod -n ibm-spectrum-scale-csi-driver
.
.
Events:
  Type     Reason              Age                     From                     Message
  ----     ------              ----                    ----                     -------
  Normal   Scheduled           8s                      default-scheduler        Successfully
assigned spectrum-scale-csi/csi-scale-fsetdemo-pod to scuttleclaw-compute4
  Warning  FailedAttachVolume  <invalid> (x6 over 8s)  attachdetach-controller
AttachVolume.Attach failed for volume "pvc-f3024f7a-06be-11ea-9384-00505695e231" : rpc error:
code = Internal desc = ControllerPublishVolume : SKIP_MOUNT_UNMOUNT == yes and either fs1 or
fs1 in not mounted on node scuttleclaw-compute4.
```

Root cause

If the above error is seen despite the file system being mounted on a given node, then the root cause is that IBM Spectrum Scale node names and Kubernetes node names are different, and node mapping is not configured. For more information, see “Kubernetes to IBM Spectrum Scale node mapping” on page 26.

Accessibility features for IBM Spectrum Scale

Accessibility features help users who have a disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in IBM Spectrum Scale:

- Keyboard-only operation
- Interfaces that are commonly used by screen readers
- Keys that are discernible by touch but do not activate just by touching them
- Industry-standard devices for ports and connectors
- The attachment of alternative input and output devices

IBM Knowledge Center, and its related publications, are accessibility-enabled. The accessibility features are described in IBM Knowledge Center (www.ibm.com/support/knowledgecenter).

Keyboard navigation

This product uses standard Microsoft Windows navigation keys.

IBM and accessibility

See the IBM Human Ability and Accessibility Center (www.ibm.com/able) for more information about the commitment that IBM has to accessibility.

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, Nihonbashi-Hakozakicho, 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.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

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

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 at http://www.ibm.com/privacy 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” at http://www.ibm.com/software/info/product-privacy.

Glossary

This glossary provides terms and definitions for IBM Spectrum Scale.

The following cross-references are used in this glossary:

- *See* refers you from a nonpreferred term to the preferred term or from an abbreviation to the spelled-out form.
- *See also* refers you to a related or contrasting term.

For other terms and definitions, see the IBM Terminology website (www.ibm.com/software/globalization/terminology) (opens in new window).

B

block utilization
The measurement of the percentage of used subblocks per allocated blocks.

C

cluster
A loosely coupled collection of independent systems (nodes) organized into a network for the purpose of sharing resources and communicating with each other. See also *GPFS cluster*.

cluster configuration data
The configuration data that is stored on the cluster configuration servers.

Cluster Export Services (CES) nodes
A subset of nodes configured within a cluster to provide a solution for exporting GPFS file systems by using the Network File System (NFS), Server Message Block (SMB), and Object protocols.

cluster manager
The node that monitors node status using disk leases, detects failures, drives recovery, and selects file system managers. The cluster manager must be a quorum node. The selection of the cluster manager node favors the quorum-manager node with the lowest node number among the nodes that are operating at that particular time.

Note: The cluster manager role is not moved to another node when a node with a lower node number becomes active.

clustered watch folder
Provides a scalable and fault-tolerant method for file system activity within an IBM Spectrum Scale file system. A clustered watch folder can watch file system activity on a fileset, inode space, or an entire file system. Events are streamed to an external Kafka sink cluster in an easy-to-parse JSON format. For more information, see the *mmwatch command* in the *IBM Spectrum Scale: Command and Programming Reference*.

control data structures
Data structures needed to manage file data and metadata cached in memory. Control data structures include hash tables and link pointers for finding cached data; lock states and tokens to implement distributed locking; and various flags and sequence numbers to keep track of updates to the cached data.

D

Data Management Application Program Interface (DMAPI)
The interface defined by the Open Group's XDSM standard as described in the publication *System Management: Data Storage Management (XDSM) API Common Application Environment (CAE) Specification C429*, The Open Group ISBN 1-85912-190-X.

deadman switch timer
A kernel timer that works on a node that has lost its disk lease and has outstanding I/O requests. This timer ensures that the node cannot complete the outstanding I/O requests (which would risk causing file system corruption), by causing a panic in the kernel.

dependent fileset
A fileset that shares the inode space of an existing independent fileset.

disk descriptor
A definition of the type of data that the disk contains and the failure group to which this disk belongs. See also *failure group*.

disk leasing
A method for controlling access to storage devices from multiple host systems. Any host that wants to access a storage device configured to use disk leasing registers for a lease; in the event of a perceived failure, a host system can deny access, preventing I/O operations with the storage device until the preempted system has reregistered.

disposition
The session to which a data management event is delivered. An individual disposition is set for each type of event from each file system.

domain
A logical grouping of resources in a network for the purpose of common management and administration.

E

ECKD
See *extended count key data (ECKD)*.

ECKD device
See *extended count key data device (ECKD device)*.

encryption key
A mathematical value that allows components to verify that they are in communication with the expected server. Encryption keys are based on a public or private key pair that is created during the installation process. See also *file encryption key*, *master encryption key*.

extended count key data (ECKD)
An extension of the count-key-data (CKD) architecture. It includes additional commands that can be used to improve performance.

extended count key data device (ECKD device)
A disk storage device that has a data transfer rate faster than some processors can utilize and that is connected to the processor through use of a speed matching buffer. A specialized channel program is needed to communicate with such a device. See also *fixed-block architecture disk device*.

F

failback
Cluster recovery from failover following repair. See also *failover*.

failover
(1) The assumption of file system duties by another node when a node fails. (2) The process of transferring all control of the ESS to a single cluster in the ESS when the other clusters in the ESS fails. See also *cluster*. (3) The routing of all transactions to a second controller when the first controller fails. See also *cluster*.

failure group
A collection of disks that share common access paths or adapter connections, and could all become unavailable through a single hardware failure.

FEK
See *file encryption key*.

fileset
A hierarchical grouping of files managed as a unit for balancing workload across a cluster. See also *dependent fileset*, *independent fileset*.

fileset snapshot
A snapshot of an independent fileset plus all dependent filesets.

file audit logging
Provides the ability to monitor user activity of IBM Spectrum Scale file systems and store events related to the user activity in a security-enhanced fileset. Events are stored in an easy-to-parse JSON format. For more information, see the *mmaudit command* in the *IBM Spectrum Scale: Command and Programming Reference*.

file clone
A writable snapshot of an individual file.

file encryption key (FEK)
A key used to encrypt sectors of an individual file. See also *encryption key*.

file-management policy
A set of rules defined in a policy file that GPFS uses to manage file migration and file deletion. See also *policy*.

file-placement policy
A set of rules defined in a policy file that GPFS uses to manage the initial placement of a newly created file. See also *policy*.

file system descriptor
A data structure containing key information about a file system. This information includes the disks assigned to the file system (*stripe group*), the current state of the file system, and pointers to key files such as quota files and log files.

file system descriptor quorum
The number of disks needed in order to write the file system descriptor correctly.

file system manager
The provider of services for all the nodes using a single file system. A file system manager processes changes to the state or description of the file system, controls the regions of disks that are allocated to each node, and controls token management and quota management.

fixed-block architecture disk device (FBA disk device)
A disk device that stores data in blocks of fixed size. These blocks are addressed by block number relative to the beginning of the file. See also *extended count key data device*.

fragment
The space allocated for an amount of data too small to require a full block. A fragment consists of one or more subblocks.

G

global snapshot
A snapshot of an entire GPFS file system.

GPFS cluster
A cluster of nodes defined as being available for use by GPFS file systems.

GPFS portability layer
The interface module that each installation must build for its specific hardware platform and Linux distribution.

GPFS recovery log
A file that contains a record of metadata activity and exists for each node of a cluster. In the event of a node failure, the recovery log for the failed node is replayed, restoring the file system to a consistent state and allowing other nodes to continue working.

I

ill-placed file
A file assigned to one storage pool but having some or all of its data in a different storage pool.

ill-replicated file
A file with contents that are not correctly replicated according to the desired setting for that file. This situation occurs in the interval between a change in the file's replication settings or suspending one of its disks, and the restripe of the file.

independent fileset
A fileset that has its own inode space.

indirect block
A block containing pointers to other blocks.

inode
The internal structure that describes the individual files in the file system. There is one inode for each file.

inode space
A collection of inode number ranges reserved for an independent fileset, which enables more efficient per-fileset functions.

ISKLM
IBM Security Key Lifecycle Manager. For GPFS encryption, the ISKLM is used as an RKM server to store MEKs.

J

journaled file system (JFS)
A technology designed for high-throughput server environments, which are important for running intranet and other high-performance e-business file servers.

junction
A special directory entry that connects a name in a directory of one fileset to the root directory of another fileset.

K

kernel
The part of an operating system that contains programs for such tasks as input/output, management and control of hardware, and the scheduling of user tasks.

M

master encryption key (MEK)
A key used to encrypt other keys. See also *encryption key*.

MEK
See *master encryption key*.

metadata
Data structures that contain information that is needed to access file data. Metadata includes inodes, indirect blocks, and directories. Metadata is not accessible to user applications.

metanode
The one node per open file that is responsible for maintaining file metadata integrity. In most cases, the node that has had the file open for the longest period of continuous time is the metanode.

mirroring
The process of writing the same data to multiple disks at the same time. The mirroring of data protects it against data loss within the database or within the recovery log.

Microsoft Management Console (MMC)

A Windows tool that can be used to do basic configuration tasks on an SMB server. These tasks include administrative tasks such as listing or closing the connected users and open files, and creating and manipulating SMB shares.

multi-tailed

A disk connected to multiple nodes.

N

namespace

Space reserved by a file system to contain the names of its objects.

Network File System (NFS)

A protocol, developed by Sun Microsystems, Incorporated, that allows any host in a network to gain access to another host or netgroup and their file directories.

Network Shared Disk (NSD)

A component for cluster-wide disk naming and access.

NSD volume ID

A unique 16-digit hex number that is used to identify and access all NSDs.

node

An individual operating-system image within a cluster. Depending on the way in which the computer system is partitioned, it may contain one or more nodes.

node descriptor

A definition that indicates how GPFS uses a node. Possible functions include: manager node, client node, quorum node, and nonquorum node.

node number

A number that is generated and maintained by GPFS as the cluster is created, and as nodes are added to or deleted from the cluster.

node quorum

The minimum number of nodes that must be running in order for the daemon to start.

node quorum with tiebreaker disks

A form of quorum that allows GPFS to run with as little as one quorum node available, as long as there is access to a majority of the quorum disks.

non-quorum node

A node in a cluster that is not counted for the purposes of quorum determination.

Non-Volatile Memory Express (NVMe)

An interface specification that allows host software to communicate with non-volatile memory storage media.

P

policy

A list of file-placement, service-class, and encryption rules that define characteristics and placement of files. Several policies can be defined within the configuration, but only one policy set is active at one time.

policy rule

A programming statement within a policy that defines a specific action to be performed.

pool

A group of resources with similar characteristics and attributes.

portability

The ability of a programming language to compile successfully on different operating systems without requiring changes to the source code.

primary GPFS cluster configuration server

In a GPFS cluster, the node chosen to maintain the GPFS cluster configuration data.

private IP address
A IP address used to communicate on a private network.

public IP address
A IP address used to communicate on a public network.

Q

quorum node
A node in the cluster that is counted to determine whether a quorum exists.

quota
The amount of disk space and number of inodes assigned as upper limits for a specified user, group of users, or fileset.

quota management
The allocation of disk blocks to the other nodes writing to the file system, and comparison of the allocated space to quota limits at regular intervals.

R

Redundant Array of Independent Disks (RAID)
A collection of two or more disk physical drives that present to the host an image of one or more logical disk drives. In the event of a single physical device failure, the data can be read or regenerated from the other disk drives in the array due to data redundancy.

recovery
The process of restoring access to file system data when a failure has occurred. Recovery can involve reconstructing data or providing alternative routing through a different server.

remote key management server (RKM server)
A server that is used to store master encryption keys.

replication
The process of maintaining a defined set of data in more than one location. Replication consists of copying designated changes for one location (a source) to another (a target) and synchronizing the data in both locations.

RKM server
See *remote key management server*.

rule
A list of conditions and actions that are triggered when certain conditions are met. Conditions include attributes about an object (file name, type or extension, dates, owner, and groups), the requesting client, and the container name associated with the object.

S

SAN-attached
Disks that are physically attached to all nodes in the cluster using Serial Storage Architecture (SSA) connections or using Fibre Channel switches.

Scale Out Backup and Restore (SOBAR)
A specialized mechanism for data protection against disaster only for GPFS file systems that are managed by IBM Spectrum Protect Hierarchical Storage Management (HSM).

secondary GPFS cluster configuration server
In a GPFS cluster, the node chosen to maintain the GPFS cluster configuration data in the event that the primary GPFS cluster configuration server fails or becomes unavailable.

Secure Hash Algorithm digest (SHA digest)
A character string used to identify a GPFS security key.

session failure
The loss of all resources of a data management session due to the failure of the daemon on the session node.

session node
The node on which a data management session was created.

Small Computer System Interface (SCSI)
An ANSI-standard electronic interface that allows personal computers to communicate with peripheral hardware, such as disk drives, tape drives, CD-ROM drives, printers, and scanners faster and more flexibly than previous interfaces.

snapshot
An exact copy of changed data in the active files and directories of a file system or fileset at a single point in time. See also *fileset snapshot*, *global snapshot*.

source node
The node on which a data management event is generated.

stand-alone client
The node in a one-node cluster.

storage area network (SAN)
A dedicated storage network tailored to a specific environment, combining servers, storage products, networking products, software, and services.

storage pool
A grouping of storage space consisting of volumes, logical unit numbers (LUNs), or addresses that share a common set of administrative characteristics.

stripe group
The set of disks comprising the storage assigned to a file system.

striping
A storage process in which information is split into blocks (a fixed amount of data) and the blocks are written to (or read from) a series of disks in parallel.

subblock
The smallest unit of data accessible in an I/O operation, equal to one thirty-second of a data block.

system storage pool
A storage pool containing file system control structures, reserved files, directories, symbolic links, special devices, as well as the metadata associated with regular files, including indirect blocks and extended attributes The `system storage pool` can also contain user data.

T

token management
A system for controlling file access in which each application performing a read or write operation is granted some form of access to a specific block of file data. Token management provides data consistency and controls conflicts. Token management has two components: the token management server, and the token management function.

token management function
A component of token management that requests tokens from the token management server. The token management function is located on each cluster node.

token management server
A component of token management that controls tokens relating to the operation of the file system. The token management server is located at the file system manager node.

transparent cloud tiering (TCT)
A separately installable add-on feature of IBM Spectrum Scale that provides a native cloud storage tier. It allows data center administrators to free up on-premise storage capacity, by moving out cooler data to the cloud storage, thereby reducing capital and operational expenditures.

twin-tailed
A disk connected to two nodes.

U

user storage pool
A storage pool containing the blocks of data that make up user files.

V

VFS
See *virtual file system*.

virtual file system (VFS)
A remote file system that has been mounted so that it is accessible to the local user.

virtual node (vnode)
The structure that contains information about a file system object in a virtual file system (VFS).

W

watch folder API
Provides a programming interface where a custom C program can be written that incorporates the ability to monitor inode spaces, filesets, or directories for specific user activity-related events within IBM Spectrum Scale file systems. For more information, a sample program is provided in the following directory on IBM Spectrum Scale nodes: `/usr/lpp/mmfs/samples/util` called tswf that can be modified according to the user's needs.

Index

SC28-3113-02