InfoScale™ for Kubernetes 9.1.2 - Linux

Last updated: 2026-04-17

Legal Notice

Enterprise Vault, a business unit of Cloud Software Group.

www.enterprisevault.com

Technical Support

Technical Support operates global support centers dedicated to addressing inquiries regarding product features and functionality. Furthermore, the Technical Support group is responsible for the development of content for the online Knowledge Base. Through collaboration with various functional departments, the group ensures that all customer inquiries are addressed with appropriate diligence in a timely manner.

Our support offerings include the following:

- A range of support options that give you the flexibility to select the right amount of service for any size organization
- Telephone and/or Web-based support that provides rapid response and up-to-the-minute information
- Upgrade assurance that delivers software upgrades
- Global support purchased on a regional business hours or 24 hours a day, 7 days a week basis
- Premium service offerings that include Account Management Services

For information about our support offerings, you can visit our website at the following URL:

https://supportenterprisevault.cloud.com/

All support services will be delivered in accordance with your support agreement and the then-current enterprise technical support policy.

Documentation

Make sure that you have the current version of the documentation. Each document displays the date of the last update on page 2. The latest documentation is available here: https://sort.veritas.com/arctera/documents

Services and Operations Readiness Tools (SORT)

Services and Operations Readiness Tools (SORT) is a website that provides information and tools to automate and simplify certain time-consuming administrative tasks. Depending on the product, SORT helps you prepare for installations and upgrades, identify risks in your datacenters, and improve operational efficiency. To see what services and tools SORT provides for your product, see the data sheet:
https://sort.veritas.com/data/support/SORT_Data_Sheet.pdf

Contents

Chapter 1

Overview

This chapter includes the following topics:

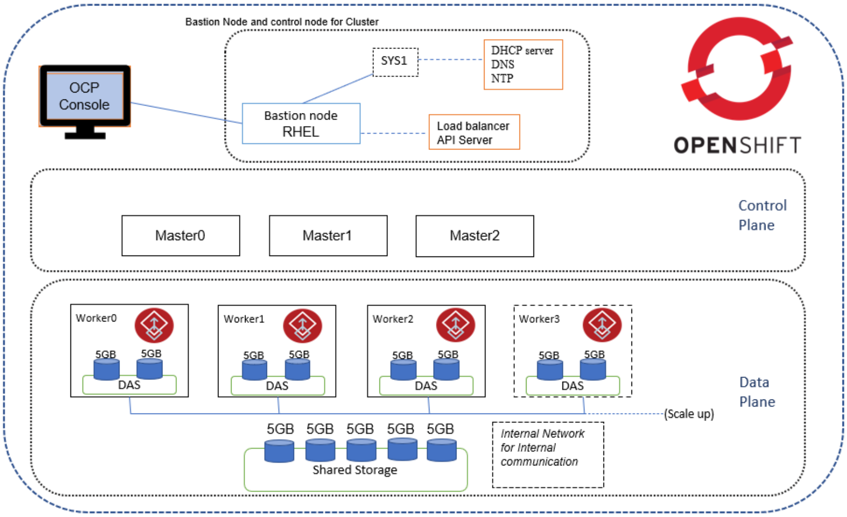
Introduction

With organizations increasingly adopting Container environments, it is necessary that all applications and storage must be available on these environments.

InfoScale provides high-performance shared storage for the OpenShift and Kubernetes clusters by using the fast storage, directly attached to the cluster nodes. InfoScale Storage provides highly available persistent storage that conforms to CSI specifications for enterprise applications by using high-performance parallel storage access on shared storage (SAN) or in Flexible Storage Sharing (FSS) environments. Multiple InfoScale clusters can be configured within a single OpenShift or Kubernetes cluster.

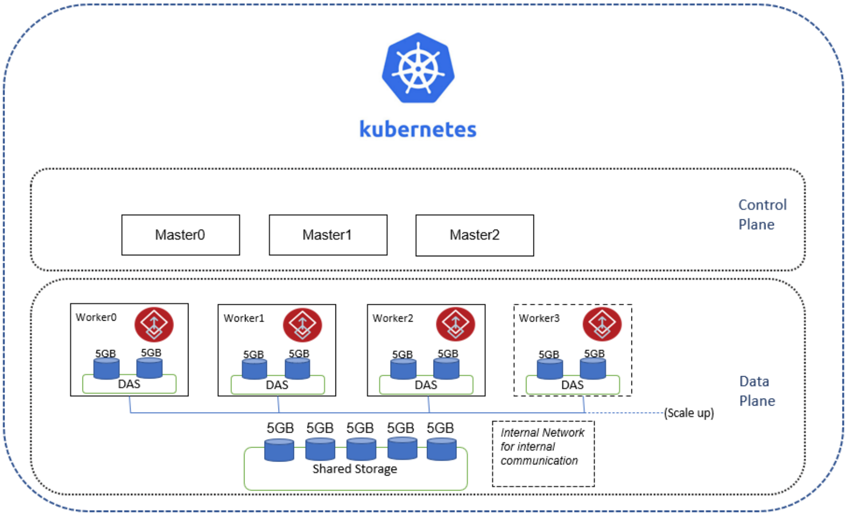
For OpenShift, you can download files from the Red Hat registry and deploy InfoScale. With an active Red Hat account, you can access the InfoScale images. Download from a single source with a single sign-on ensures a high level of security. An example of an OpenShift cluster is as under

Figure 1-1 OpenShift cluster comprising three master nodes, four worker nodes, and one bastion node. Storage can be SAN or DAS.



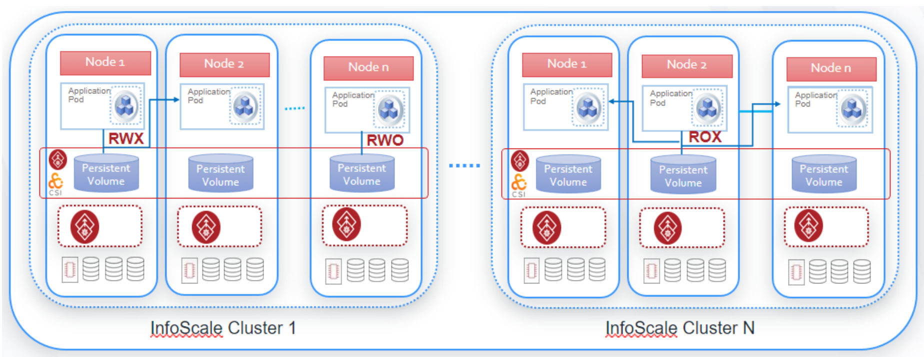
For Kubernetes, you can download files from the Download Center at: https://www.veritas.com/content/support/en_US/downloads, and deploy InfoScale. An example of a Kubernetes cluster is as under

Figure 1-2 Kubernetes cluster comprising three master and four worker nodes. Storage can be SAN or DAS.



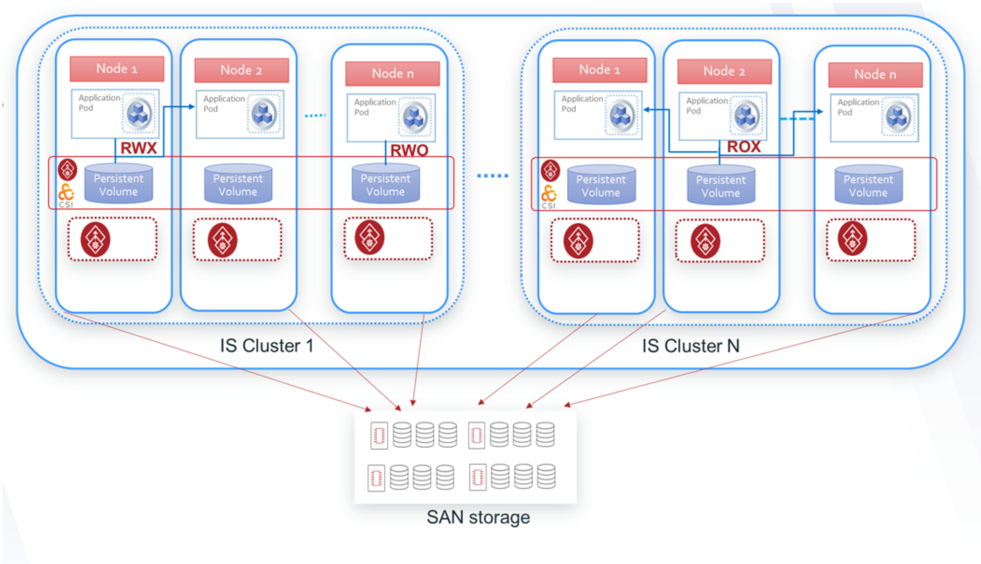
InfoScale InfoScale in Kubernetes environment also offers scalability support which is helpful in case the Container environment has large number of nodes. You can configure any number of InfoScale clusters within a single OpenShift or Kubernetes cluster. Each cluster can have upto 16 nodes. To configure multiple clusters, you must modify the configuration resource file and add information about the nodes you want to be included in that InfoScale cluster.

Figure 1-3 Multiple InfoScale clusters within a single OpenShift or Kubernetes cluster with Direct Attached Storage (DAS)



Multiple InfoScale clusters enable you to run containerized applications on a subset of nodes. Storage attached to all the nodes across the cluster can thus be consumed. However, InfoScale clusters cannot share storage with each other.

Figure 1-4 Multiple InfoScale clusters within a single OpenShift or Kubernetes cluster in a SAN environment



Features of InfoScale in Containerized environment

- Support for Direct-Attached Storage(DAS) and Storage Area Network(SAN) in a multi-vendor environment.
- High-performance parallel storage that provides better performance and reliability than NFS.
- Optimized resource utilization with the ability to use either existing SAN storage or the InfoScale advanced FSS option that provides better performance than SAN at a reduced cost.
- Support for Dynamic Multipathing (DMP).
- Support for encryption at disk group and volume level.

Besides the traditional features listed above, new features are -

- Integrated I/O fencing and arbitration to protect against data corruption and to provide fast recovery in the event of a failure.

- Snapshot copies of the production data for analytics and disaster recovery.
- Volume cloning to address storage disk and node failures.
- Support for stateful applications like MySQL, Oracle, PostgreSQL.
- Support for multiple InfoScale clusters within a single OpenShift or Kubernetes cluster.

CSI Introduction

CSI is a standardized mechanism for Container Orchestrators (COs) to expose arbitrary storage systems to their containerized workloads. The InfoScale CSI plugin is used to provide persistent InfoScale Storage to container workloads or applications. The InfoScale CSI plugin supports creation of storage classes for high availability, performance, and capacity. Online expansion of capacity as well as the usage of snapshots and clones is also supported. Raw block volume is also supported, where applications directly write to the provided block device.

I/O fencing

InfoScale uses majority-based and disk-based I/O fencing to guarantee data protection and provide persistent storage in the Container environment. Fencing ensures that data protection gets highest priority and stops running the systems when a split-brain condition is encountered. The systems thus cannot start services and data is protected. InfoScale checks for the connectivity with each peer nodes periodically while OpenShift or Kubernetes check it for the master to worker nodes.

The OpenShift or Kubernetes cluster performs failover or restart of applications for the nodes that have reached a `NotReady` state in the OpenShift or Kubernetes cluster. If an application is configured as a statefulset pod, the Container stops failover of such application pods till the node becomes active again. In such scenarios, InfoScale uses the fencing module to ensure that the application pods running on such unreachable nodes cannot access the persistent storage so that OpenShift or Kubernetes can restart these pods on the active cluster without the risk of any data corruption.

When InfoScale cluster is deployed on an OpenShift or Kubernetes, InfoScale uses a custom fencing controller to provide the fencing infrastructure. The custom controller interacts with the InfoScale fencing driver and enables failover in OpenShift or Kubernetes in case of a network split. An agent running on the controller ensures that InfoScale fences out the persistent storage and performs the pod failover for the fenced-out node. It also ensures that the fencing decisions of InfoScale I/O fencing do not conflict with the fencing decisions of the fencing controller.

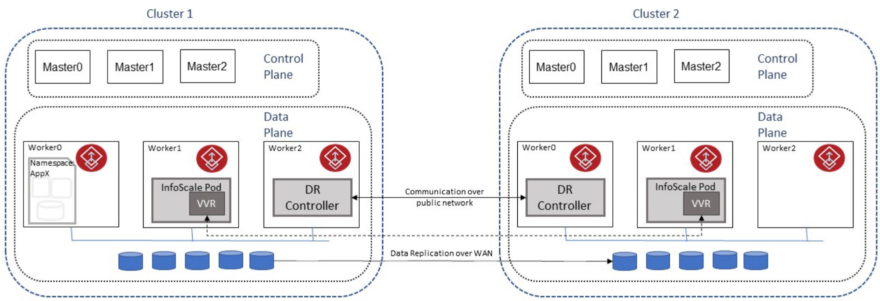
For deployment in containerized environments, when you install InfoScale by using the product installer, the fencing module is automatically installed and configured in majority mode by default. For the default flexible shared storage (fss) disk group configuration, default majority-based fencing can be used. In case of network split, the I/O fencing module takes fencing decisions based on the number of nodes in a sub-cluster. If majority of the nodes are down, all nodes in the cluster are brought down.

If you are using fully-shared storage, where all disks are connected to all the nodes; InfoScale recommends using disk-based fencing. With disk-based fencing, cluster is up even with a single functional node.

Disaster Recovery

Disaster Recovery(DR) is provided to applications hosted in container ecosystems. Native container HA capabilities provide high availability to application components within a cluster. However, DR functionality provides disaster recovery in the event of a cluster failure and application components can migrate to another peer cluster in membership. You can form a logical notion called 'Global Cluster' comprising clusters that can be used to migrate DR-enabled objects. DR-enabled objects migrate to peer cluster in case of a disaster like entire cluster going down, loss of connectivity with a particular cluster, user-initiated planned migration across cluster(s). Peer-to-peer communication between DR controllers is encrypted.

Figure 1-5 DR Configuration comprising two clusters



You can configure a Disaster Recovery Plan(DR Plan) for a given namespace. For a more granular control, you can specify labels along with the namespace. In DR plan, you also specify the primary cluster and a DR cluster. Workload is shifted to the DR cluster if the primary cluster fails. For maintenance activities, you can also initiate a graceful migration of DR plan across peer cluster. Application instances are migrated along with associated persistent data(in case of stateful application). For replicating persistent data across peer cluster, it uses InfoScale Volume Replicator(VVR).

Disaster Recovery can be configured across on-premise and cloud combinations. Primary site can be on-premise or on cloud. Similarly, the secondary site can be on-premise or on cloud. Disaster Recovery can be set up in any combination of the primary and secondary site. Load balancer can be configured when the site is on cloud. Load balancer configuration ensures traffic during data replication is directed to the appropriate worker nodes. Currently the supported cloud services are - Azure and Amazon Web Services (AWS).

Note: InfoScale Licensing Custom Resource (CR) must be created before creating the Disaster Recovery (DR) custom resources. To enable DR, you must have Enterprise type of license.

Licensing

You must create a license to enable use of InfoScale. InfoScale in Containers Environment is offered in only Enterprise edition.

Encryption

With encryption, you can secure data at rest and during transmission. Encryption is a technology which converts data into a code by using a key. Data when encrypted is not readable. It is readable only when decrypted by authorized users using the same key that was used for encryption.

Encryption is performed by using the FIPS-compliant Advanced Encryption Standard (AES) 256-bit method. By default, encryption is not enabled. You can enable encryption at disk group level while configuring `cr.yaml`.

When Disaster Recovery (DR) is configured, encrypted data is transferred across sites. For DR configuration, encryption is supported at volume level only.

With encryption, you can -

- Protect data from unauthorized access.
- Retire disks without the overhead expense of clearing data.

Chapter 2

System requirements

This chapter includes the following topics:

Introduction

The System Requirements to run InfoScale in Containerized environment are listed here. The supported container platforms are also listed.

This document is intended for the use of a Storage Administrator who knows how to install, configure, and administer Applications in a Linux environment. Additionally, it is assumed that the user knows Container-related concepts like nodes, pods, and types of nodes and commands to access nodes. To know more about these concepts and commands, OpenShift and Kubernetes documentation can be referred.

Supported platforms

The following table lists the supported container platforms.

Table 2-1 Supported container platforms

Platform / Configuration	Version
OpenShift Container Platform (OCP)	4.16.28 , 4.16.55 4.17.10, 4.17.36, 4.17.44, 4.17.47 4.18.16, 4.18.23, 4.18.26, 4.18.27, 4.18.30, 4.18.32, 4.18.33 4.19.11, 4.19.16, 4.19.22, 4.19.23, 4.19.24, 4.19.25 4.20.3, 4.20.8, 4.20.10, 4.20.14, 4.20.15 4.21.0, 4.21.2, 4.21.3
Kubernetes	1.34.3 on RHEL 9.4
Red Hat Cert Manager	Cert Manager v1.17.0 or later for OCP 4.17.x, 4.18.x, 4.19.x, OCP 4.20.x

Note the following:

- Containerd is the supported runtime environment for Kubernetes. Crio is the supported runtime environment for OpenShift.
- UPI, IPI, or any other customized solution is supported for installation on OCP. Deployment environment can be Bare Metal or virtual (VMware ESXi).
- The configuration comprises a network of master nodes, worker nodes, and a bastion node. CLI access from the bastion node to the master nodes and the worker nodes must be enabled.
- You can perform a rolling upgrade of OCP or Kubernetes platforms. Before upgrading, refer to the supported platform and kernel versions table. Ensure that you upgrade to an InfoScale-supported version. Refer to OpenShift or Kubernetes documentation for steps to upgrade. For redundant storage and applications, there is no downtime during an upgrade.

Disk space requirements

The following table lists the minimum disk space requirements for each product when the `/opt`, `/root`, `/var`, and `/bin` directories are created on the same disk.

Table 2-2 Disk space requirements

Product name	Disk space (MB)
InfoScale InfoScale Storage	3305

Note: OpenShift Container Platform runs special components which consume memory. See the OpenShift documentation - https://docs.openshift.com/container-platform/4.9/ nodes/clusters/nodes-cluster-resource-configure.html to understand memory requirements. As an OpenShift administrator, set resource limit for the Prometheus pod. See https://access.redhat.com/solutions/3867881 to know the Red Hat recommendations. Modify `resources.limits` and `resources.requests` before installing InfoScale.

Hardware requirements

Note: These requirements are for a basic cluster configuration. Depending on your workloads/applications, higher values might be required.

Table 2-3 Hardware requirements

Requirement	Description
Memory (Operating System)	Minimum 24 GB
CPU (on Kubernetes)	On Physical servers - a minimum of 2 processors with 6/8 cores each. On Virtual machines (VMware-like environment) - a minimum of 4 vCPUs.
CPU (on OpenShift)	On Physical servers - a minimum of 2 processors with 6/8 cores each. On Virtual machines (VMware-like environment) - a minimum of 12 vCPUs for master node and 8 vCPUs for worker nodes.
Node	All nodes in a Cluster must have the same operating system version.
Storage	Storage can be one or more shared disks, or a disk array connected either directly to the nodes of the cluster or through a Fibre Channel Switch. Nodes can also have non-shared or local devices on a local I/O channel. In a Flexible Storage Sharing (FSS) environment, shared storage may not be required.

Table 2-3 Hardware requirements *(continued)*

Requirement	Description
Cluster platforms	There are several hardware platforms that can function as nodes in a InfoScale cluster. For the InfoScale cluster to work correctly, all nodes must have the same time. If you are not running the Network Time Protocol (NTP) daemon, ensure the time on all the systems comprising your cluster is synchronized.
SAS or FCoE	Each node in the cluster must have an SAS or FCoE I/O channel to access shared storage devices. The primary components of the SAS or Fibre Channel over Ethernet (FCoE) fabric are the switches and HBAs.

For additional information, see the hardware compatibility list (HCL):

https://www.veritas.com/content/support/en_US/doc/infoscale_hcl_8x_unix

Number of nodes supported

InfoScale for Containers supports cluster configurations comprising up to 16 worker nodes. Any number of InfoScale clusters can be configured.

DR support

DR supports two cluster configuration - one primary cluster and one secondary/DR cluster.

Chapter 3

Preparing to install InfoScale on Containers

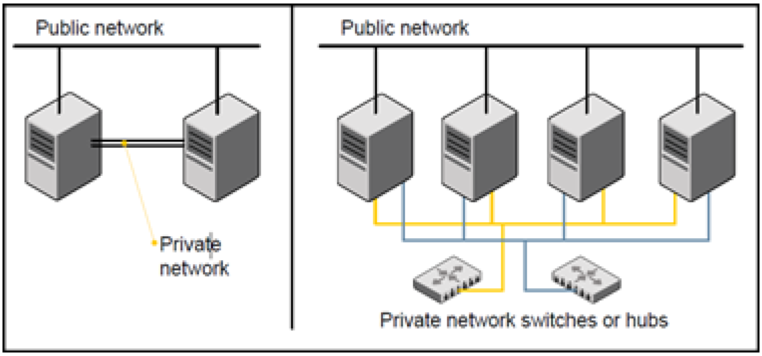
This chapter includes the following topics:

Setting up the private network

If you do not specify IP addresses in `cr.yaml` (the configuration file used for specifying details like Node names, IP addresses), InfoScale for Containers uses the Container links. InfoScale recommends setting up a private network between the systems for optimal performance. You can use either NICs or aggregated interfaces to set up private network. You can use network switches instead of hubs. Refer to the Cluster Server Administrator's Guide to review performance considerations.

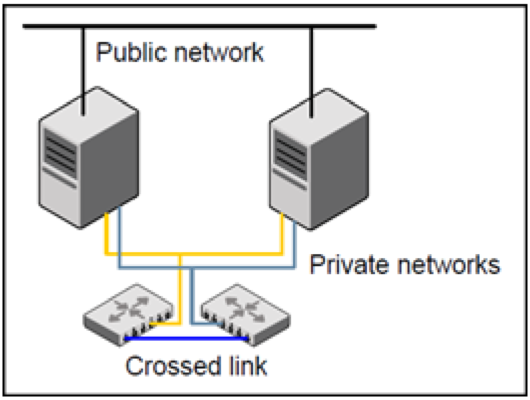
The following figure shows two private networks for use with InfoScale.

Figure 3-1 Private network setups: two-node and four-node clusters



You need to configure at least two independent networks between the cluster nodes with a network switch for each network. You can also interconnect multiple layer 2 (L2) switches for advanced failure protection. Such connections for LLT are called cross-links. The following figure shows a private network configuration with crossed links between the network switches.

Figure 3-2 Private network setup with crossed links



If an L2 network (an L2 network is where hosts are connected directly by a switch without a router in between) is not available, you can alternatively also configure an L3 network. In an L3 network, the hosts can connect to each other using layer 3 (L3) routers. As InfoScale for Containers uses UDP as the transport layer protocol, InfoScale pods can communicate with each other across routers.

InfoScale recommends the following configuration

- Use at least two private interconnect links. The private interconnect link is used to share cluster status across all the systems, which is important for membership arbitration and storage access.

To set up the private network

1. Install the required network interface cards (NICs). Create aggregated interfaces if you want to use the NICs to set up private network
2. Use crossover Ethernet cables, switches, or independent hubs for each InfoScale communication network. Note that the crossover Ethernet cables are supported only on two systems.

   Ensure that you meet the following requirements:

   - The power to the switches or hubs must come from separate sources.
   - On each system, you must use two independent network cards to provide redundancy.
   - If a network interface is part of an aggregated interface, you must not configure the network interface under LLT. However, you can configure the aggregated interface under LLT.
   - When you configure Ethernet switches for LLT private interconnect, disable the spanning tree algorithm on the ports used for the interconnect.
3. During the process of setting up heartbeat connections, consider a case where a failure removes all communications between the systems.

   Note: Note that a chance for data corruption exists if the systems are still running and can access the shared storage.

   ---

4. Test the network connections. Assign network addresses and use telnet or ping to verify communications.

Guidelines for setting the media speed for LLT interconnects

Review the following guidelines for setting the media speed for LLT interconnects:

- InfoScale recommends that you set the same media speed setting on each Ethernet card on each node.
- If you have hubs or switches for LLT interconnects, then set the hub or switch port to the same setting as used on the cards on each node. Details for setting the media speeds for specific devices are outside of the scope of this manual. Consult the device’s documentation or the operating system manual for more information.

Guidelines for setting the maximum transmission unit (MTU) for LLT

LLT can operate on default 1500 MTU but InfoScale recommends increasing MTU to achieve maximum performance. Review the following guidelines for setting the MTU for LLT interconnects:

- Set the maximum transmission unit (MTU) to the highest value (typically 9000) supported by the NICs when LLT links are configured over UDP. Ensure that the switch is also set to 9000 MTU.
- For virtual NICs, all components (the virtual NIC, the corresponding physical NIC, and the virtual switch) must be set to 9000 MTU.

Note: InfoScale for Containers uses only LLT over UDP as the transport layer. Other transport protocols such as LLT over ethernet, LLT over RDMA, and LLT over TCP are not supported in a container environment.

Synchronizing time settings on cluster nodes

Ensure that the time settings on all OpenShift and Kubernetes cluster nodes are synchronized. If the nodes are not synchronized, timestamps for change (ctime) and modification (mtime) may not be consistent with the sequence in which operations actually happened. For instructions, see the operating system documentation.

Securing your InfoScale deployment

Consider the following measures on your OpenShift and Kubernetes clusters. After adopting these measures, InfoScale deployment on these clusters is more secure.

See OpenShift and Kubernetes documentation to know more about these measures.

1. On an air gapped system on OpenShift or a Kubernetes cluster, configure a secure image registry. This registry is used to download and host InfoScale images.

   Enable the following to reduce security risks.

   - Set up secure, encrypted channels to connect to the registry.
   - Authenticate users and control access to registry.
   - Scan images for vulnerabilities found in the Common Vulnerabilities and Exploits (CVE) database and sign these as known and trusted.

2. Enable encryption at rest and assign RBAC for sensitive data stored in OpenShift and Kubernetes Secrets. By default, data is stored unencrypted in the API server's underlying data store (`etcd`). Anyone with API access or access to `etcd`, can retrieve or modify a Secret. Additionally, anyone who is authorized to create a pod in a namespace can use that access to read any Secret in that namespace; this includes indirect access such as the ability to create a deployment. When encryption at rest is enabled with appropriate RBAC to secrets, the sensitive data remains protected.
3. Configure the OpenShift or Kubernetes API server with TLS 1.2 or higher, and TLS ciphers to exclude vulnerable ciphers such as ciphers using block ciphers in CBC mode and ciphers using low-length encryption keys like DES block ciphers (56-bit encryption key).

   After this TLS configuration, use of SSL, unauthorized versions of TLS protocols, and vulnerable TLS ciphers is blocked and confidentiality of sensitive data during electronic transmission is maintained.
4. Two strong cipher suites are enabled by default. You can edit the operator yaml files and modify `TLS_CIPHER_SUITES` and change the values of the cipher suites. You can indicate two cipher suites separated by a comma.

   If you enter an invalid cipher input and change the input to a valid cipher input, the InfoScale pods might take time to restart. In such a case, wait for the pods to restart automatically or you can delete the pods manually and restart.
5. On a Kubernetes cluster, create a secret by name infoscale-imgpuller to use the secured custom registry. The secrets must be created in the namespace where you plan to install InfoScale operators and all the namespaces where you want to configure InfoScale clusters.

   Before deploying DR Operator through secured custom registry, we need to create a 'infoscale-dr-imgpuller' secret as follows:

   ```
   kubectl create secret docker-registry infoscale-imgpuller
   --docker-server=<FQDN of the private docker registry>
   --docker-username=<Docker username> --docker-password=<Docker
   password> --docker-email=<Docker email ID> --namespace <Namespace>
   ```

   To know more about image pull secret, refer to Kubernetes documentation.

Configuring kdump

InfoScale recommends configuring kdump on each of the cluster nodes before installing InfoScale. kdump creates crash dumps in the event of a kernel crash which help InfoScale support in troubleshooting issues.

For OpenShift, see https://docs.openshift.com/container-platform/4.10/support/troubleshooting/troubleshooting-operating-system-issues.html.

For Kubernetes, you can refer to the Operating System documentation for the generic steps.

Chapter 4

Installing InfoScale on OpenShift

This chapter includes the following topics:

Introduction

This chapter informs you how to install InfoScale on an OpenShift cluster. For air gapped systems on OpenShift, installer files and container images must be downloaded from the InfoScale Download Center. The container images are different for each platform. OpenShift systems with internet connectivity need to download installer files (yamls) only. You can install InfoScale from a VM/Server termed as the bastion node on an OpenShift cluster.

Note: You can form multiple InfoScale clusters. Each cluster can have upto 16 nodes. InfoScale is deployed on all the nodes you specify in the Custom Resource yaml file. With HyperConverged Architecture supported, a stateful application consuming storage from an InfoScale cluster must run on any of the nodes that are part of that InfoScale cluster.

Note: If you previously installed an older version of InfoScale and are reinstalling while reusing the same Cluster ID and DiskGroup, we recommend that you reboot all nodes with reboot -f before proceeding. This clears stale configurations and helps prevent conflicts during reinstallation.

Prerequisites

1. Be ready with the following information -

   - Names of all the nodes.

     Note: Run `oc get nodes -o wide` on the bastion node to obtain Names and IP addresses of the nodes.

     Use `NAME` and `INTERNAL-IP` from the output similar to the following -

     ```
     NAME                STATUS ROLES  AGE VERSION         INTERNAL-IP
     ocp-cp-1.lab.ocp.lan Ready master 54d v1.22.1+d8c4430 192.168.22.201
     77.rhaos4.9.gitd745cab.el8
     ocp-cp-2.lab.ocp.lan Ready master 54d v1.22.1+d8c4430 192.168.22.202
     77.rhaos4.9.gitd745cab.el8
     ocp-cp-3.lab.ocp.lan Ready master 54d v1.22.1+d8c4430 192.168.22.203
     77.rhaos4.9.gitd745cab.el8
     ocp-w-1.lab.ocp.lan Ready worker  54d v1.22.1+d8c4430 192.168.22.211
     77.rhaos4.9.gitd745cab.el8
     ocp-w-2.lab.ocp.lan Ready worker  54d v1.22.1+d8c4430 192.168.22.212
     77.rhaos4.9.gitd745cab.el8
     ocp-w-3.lab.ocp.lan Ready worker  54d v1.22.1+d8c4430 192.168.22.213
     77.rhaos4.9.gitd745cab.el8
     ocp-w-4.lab.ocp.lan Ready worker  54d v1.22.1+d8c4430 192.168.22.214
     77.rhaos4.9.gitd745cab.el8
     ```

   - Operating system hardware path of the disks which are being managed by other storage vendors that need to be excluded from InfoScale disk group.

- Optionally if you want to exclude boot disks, device path to the boot disks.

  Note: InfoScale recommends excluding boot disks.

- If you have internet connectivity and download is allowed, you must be logged in to Red Hat registry.
- For air gapped systems, Custom Registry address to set up registry where InfoScale images are pushed.
- Fencing controller and fencing enabler pods require access to an external repository. Allow access to the following registry:

  `Fencing registry - "ghcr.io/kvaps"`

2. Ensure that all nodes are synchronized with the NTP Server.
3. Reserve network ports for exclusive use of InfoScale as under -

Component	Port
LLT over UDP	Serially onwards 50000 (as many as configured LLT links)
VVR (Needed only if you want to configure DR)	4145 (UDP), 8199 (TCP), 8989 (TCP)

4. Add local or shared storage to all the worker nodes before you proceed with the deployment.
5. Ensure that stale InfoScale kernel modules (`vxio`,`vxdmp`,`veki`,`vxspec`,`vxfs`,`odm`,`glm`,`gms`) from previous installation do not exist on any of the worker nodes.

   Note: You can reboot a worker node to unload all stale InfoScale kernel modules.

6. Hostnames of the InfoScale nodes must exactly match with the fully qualified domain names(FQDN) of the OpenShift nodes.
7. OpenShift cluster must be configured by using IPv4 only.
8. Cert Manager v1.17.0 or latest based on the OCP version should be installed.

Enabling kubelet inhibitor with systemd

To enable kubelet's graceful shutdown inhibitor on worker nodes, users must apply two configurations in the following order:

1. systemd change with machine config

   Extends InhibitDelayMaxSec so systemd honors kubelet's shutdown inhibitor for the full grace period.

2. Kubelet Configuration shutdown grace periods

   Defines how long kubelet will wait for regular and critical pods during shutdown.

Both configurations trigger MCP updates and node reboots, so they should be applied deliberately.

Note: This must be performed only once in the lifetime of the cluster.

Perform these steps

1. Ensure the worker MCP contains the required label so MachineConfigs target the correct nodes:

   ```
   oc label mcp worker machineconfiguration.openshift.io/role=worker
   ```

2. Verify labels

   ```
   oc get mcp worker --show-labels
   ```

3. Apply the systemd override MachineConfig

   This config sets InhibitDelayMaxSec=900 (15 minutes), allowing kubelet to delay shutdown properly.

   Copy this butane file, change the version to OpenShift major version that is present on your cluster.

   ```
   :: worker.butane

   variant: openshift
   version: 4.19.0
   metadata:
     name: 99-worker-sysd-conf-override
     labels:
       machineconfiguration.openshift.io/role: worker
   storage:
     files:
       - path: /etc/systemd/logind.conf.d/50-max-delay.conf # << systemd
   ```

```
drop in mode: 0644
      overwrite: true
      contents:
        inline: |
          # The Machine Config Operator manages this file.
          [Login]
          InhibitDelayMaxSec=900 # << 15m
```

4. Convert the butane file to MachineConfig:

   ```
   butane worker.bu -o 99-worker-sysd-conf-override.yaml
   ```

   OR

   Can be directly applied

   ```
   https://raw.githubusercontent.com/Arctera/infoscale\n
   -kubernetes-enterprise/main/config/sysd/99-worker-sysd.yaml
   ```

   Output:

   ```
   machineconfig.machineconfiguration.openshift.io/99-worker-sysd-conf-overr
   ```

5. Apply MachineConfig to the cluster:

   ```
   oc apply -f 99-worker-sysd-conf-override.yaml
   ```

6. Check MCP rollout with below commands:

   ```
   oc get mcp
   oc get nodes
   ```

   Nodes will reboot as the new systemd config is applied.

7. Check if status of MachineConfigPool shows as:

   ```
   Updated=True

   Degraded=False
   ```

8. Apply the kubelet configuration for shutdown grace periods.

   This config sets:

   ```
   shutdownGracePeriod: 15m

   shutdownGracePeriodCriticalPods: 5m
   ```

9. Copy below YAML and apply:

```
:: mc-kubelet-config.yaml
---
 apiVersion: machineconfiguration.openshift.io/v1
 kind: KubeletConfig
 metadata:
   name: custom-kubelet-config
 spec:
   machineConfigPoolSelector:
     matchLabels:
       machineconfiguration.openshift.io/role: worker
   kubeletConfig:
     shutdownGracePeriod: "15m"
     shutdownGracePeriodCriticalPods: "5m"
```

```
oc apply -f mc-kubelet-config.yaml
```

OR

Can be directly applied

```
oc apply -f https://raw.githubusercontent.com/Arctera/infoscale-kubernete
main/config/kubelet/kubelet-config.yaml
```

Output:

```
kubeletconfig.machineconfiguration.openshift.io/custom-kubelet-config cre
```

10. Verify custom kubeletconfig.

    ```
    # oc describe kubeletconfigs.machineconfiguration.openshift.io
    custom-kubelet-config
    ```

11. Wait for MCP rollout to complete and nodes to update.

    ```
    oc get mcp
    ```

    ```
    oc get nodes
    ```

12. Check if status of MachineConfigPool shows as:

    ```
    Updated=True
    ```

    ```
    Degraded=False
    ```

13. After nodes return to Ready, login to worker nodes and run:

    ```
    systemd-inhibit --list
    ```

    Expected output:

```
A kubelet entry with WHAT=shutdown and MODE=delay.
```

InfoScale for OpenShift: Cluster configuration guidelines

Node and network infrastructure

You may specify up to 16 worker nodes in the cr.yaml.

While a single Network Interface Card (NIC) is supported, a minimum of two physical links is recommended to ensure High Availability (HA) and optimal performance.

Ensure the number of network links is identical across all nodes. If specific node-level IPs are not provided in the configuration, InfoScale will automatically inherit the existing OpenShift node IP addresses.

```
clusterInfo:
    - nodeName: "<OCP_Worker_Node_Name>"
      ip:
        - "<Primary_NIC_IP>"
        - "<Secondary_NIC_IP>"
```

Storage architecture (FSS versus shared)

InfoScale supports the following storage configurations:

- Shared storage using FSS
- Shared nothing

The configuration defaults to Flexible Storage Sharing (FSS). If your environment uses a traditional SAN, set isSharedStorage: true to create a fully shared, non-FSS disk group.

InfoScale automatically performs mirroring across enclosures to provide robust hardware redundancy.

In case of shared storage environments:

- It is recommended to use disk-based fencing as described in Optimized fencing strategies.
- Make sure that isSharedStorage and enableScsi3pr are both set to `true`.

Optimized fencing strategies

For shared storage environments, it is recommended to switch from the default majority-based fencing to disk-based fencing.

Unlike majority-based fencing—which can block storage access if the majority of nodes fail—disk-based fencing keeps storage available to applications as long as a single node remains active.

To enable disk based fencing this, specify the hardware paths for an odd number of disks (minimum of three) to serve as coordination points.

```
fencingDevice:
        - "/dev/disk/by-id/fencing-disk-1"
        - "/dev/disk/by-id/fencing-disk-2"
        - "/dev/disk/by-id/fencing-disk-3"
```

Security and encryption

Encryption is disabled by default but can be activated at either the Disk Group or individual Volume level by setting encrypted: true.

Use sameEnckey: true to apply a single encryption key across all volumes, or false for unique per-volume keys.

```
encrypted: <true|false>
sameEncKey: <true|false>
```

Selective storage management (IncludeDevices)

Note: In case if the device has multipaths, then specify only one path of the device in the IncludeDevices list.

The includeDevices attribute allows administrators to specify exact hardware paths for InfoScale to manage. This ensures that only designated disks are consumed, leaving other node-level storage available for alternative uses.

- Multipath Environments: If a device has multiple paths, specify only one path in the includeDevices list.
- FSS/DAS Configurations: You may define the list for a subset of nodes. If a node is not listed, InfoScale will automatically consume all locally attached storage on that node.
- Shared Storage (SAN): The device list only needs to be defined under a single node; InfoScale will automatically apply the configuration cluster-wide.

Scaling storage post-deployment

To add new devices to an existing cluster, follow this sequence:

1. Add the required disks to the host and verify OS discovery.
2. Add new device paths under includeDevices in the YAML and reapply:

   ```
   oc apply -f cr.yaml
   ```

Note: Removal of devices from the IncludeDevices list is not supported. Only the addition of new devices is supported. The IncludeDevices functionality can be only used for fresh VIKE deployments and not for already deployed clusters.

Monitoring and supportability

Run `oc describe infoscalecluster -n infoscale-vtas` to view the configured IncludeDevices list.

Sample output:

```
Name:         infoscalecluster-example
Namespace:    infoscale-vtas
...
Status:
  Cluster Name:  infoscalecluster-example
  Cluster Nodes:
    Node Name:   worker-node-01.example.com
    Role:        Joined, Master

    Node Name:   worker-node-02.example.com
    Role:        Joined, Slave
    Include Devices:
      /dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:1:0
      /dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:2:0

    Node Name:   worker-node-03.example.com
    Role:        Joined, Slave
```

If a listed device cannot be found or added to the disk group, the operator emits a warning event that is visible in oc describe:

```
Events:
Type      Reason                        Age                      From
----      ------                        ----                     ----
Warning   Included Devices Not Found    52s (x1744 over 21h)     InfoScale
```

```
Message
Disks missing on node worker-node-01.example.com:
[/dev/disk/by-path/ip-10.210.20
```

Note: For a DR configuration, only volume-level encryption is supported. Disk group-level encryption is not supported.

InfoScale cluster complete reference YAML:

```
apiVersion: infoscale.veritas.com/v1
kind: InfoScaleCluster
metadata:
  name: infoscalecluster-ocpv       # The unique name for your
InfoScale cluster
  namespace: infoscale-vtas         # The namespace where the InfoScale
Operator is deployed
spec:
  version: 9.1.2                    # The specific InfoScale
release version

  # clusterID: 101                  # Optional: Unique ID
(1-65535) for the
InfoScale cluster

  # --- Node Configuration Section ---
  # Define up to 16 worker nodes below
  clusterInfo:
    - nodeName: "worker-0.ocp.example.com"
      # Redundant networking: InfoScale recommends two physical links for
HA/Performance
      ip:
        - "10.10.1.10"
        - "10.10.2.10"

      # Device filtering: Include specific disks or exclude OS/local disks
      # includeDevices:
      #   - "/dev/disk/by-id/scsi-0x600..."
      excludeDevice:
        - "/dev/sda"                # Example: Exclude the OS boot disk

      # I/O Fencing: Provide hardware paths for coordination points
      # Recommended: At least 3 disks (odd number) for disk-based fencing
      fencingDevice:
```

```
        - "/dev/disk/by-id/wwn-0x6000001..."
        - "/dev/disk/by-id/wwn-0x6000002..."
        - "/dev/disk/by-id/wwn-0x6000003..."

    - nodeName: "worker-1.ocp.example.com"
      ip:
        - "10.10.1.11"
        - "10.10.2.11"
      excludeDevice:
        - "/dev/sda"

  # --- Global Storage & Cluster Parameters ---

  # Use for Air-gapped or private registry environments
  # customImageRegistry: "private-reg.example.com/infoscale"

  # Set to 'true' if using SAN (Shared Storage).
  # Set to 'false' (default) to use local DAS pooled via
Flexible Storage Sharing (FSS).
  isSharedStorage: false

  # Required for SAN-based fencing and data integrity
  enableScsi3pr: true

  # --- Security Configuration ---
  # Enable encryption for the Disk Group or specific Volumes
  encrypted: false

  # true: Use same key for all volumes in the DG
  # false: Unique key for each volume (Recommended for granular
VM security)
  sameEncKey: false
```

Considerations for configuring cluster or adding nodes to an existing cluster

You can specify up to 16 worker nodes in `cr.yaml`. Although cluster configuration is allowed even with one Network Interface Card, InfoScale recommends a minimum of two physical links for performance and High Availability (HA). Number of links for each network link must be same on all nodes. Optionally, you can enter node level IP addresses. If IP addresses are not provided, IP addresses of OpenShift cluster nodes are used.

By default, Flexible Storage Sharing (FSS) disk group is created. If shared storage is the only storage available across nodes you can set `isSharedStorage` to true and fully shared non-FSS disk group is created while configuring clusters. Mirroring is performed across enclosures thus ensuring redundancies. While using shared storage, InfoScale recommends changing the default majority-based fencing to disk-based fencing. With shared storage and majority-based fencing, storage becomes inaccessible when majority of the nodes go down; although all disks are connected to all the nodes. With disk-based fencing, storage is available to applications even when one node is up. To configure disk-based fencing, specify hardware path of the fencing disks for at least one node. InfoScale recommends using at least three disks for fencing purpose. Ensure that the number of disks is odd in number.

You can enable encryption at the disk group level or for specific Volumes within the disk group. Encryption is not enabled by default. You can set `encrypted` to true to enable encryption. If you want the same encryption key for all Volumes, you can set `sameEnckey` to true. For different encryption keys per Volume, set `sameEnckey` to false.

Note: For Disaster Recovery (DR) configuration, only Volume level encryption is supported. Disk group level encryption is not supported.

Creating multiple InfoScale clusters

After successfully installing InfoScale operator, you can create a cluster.

You can configure multiple clusters in different namespaces. After a cluster is configured and pods are created successfully, you can update `cr.yaml` with the name, namespace, Cluster ID, and node-related information of the next cluster. You can choose to rename `cr.yaml` or you can overwrite the same `cr.yaml`. You can then create the cluster and after a successful creation of pods, update `cr.yaml` for the subsequent cluster. Likewise, you can configure multiple clusters.

Each InfoScale cluster must have a unique name even when clusters are created across namespaces.

Installing InfoScale on a system with Internet connectivity

If your system is connected to the Internet, you can download operators and install. With a Red Hat account, you can connect to the Red Hat portal to download operators by using Command Line Interface (CLI) or the web console.

- Installing from OperatorHub by using web console
- Installing from OperatorHub by using Command Line Interface (CLI)

Installing from OperatorHub by using web console

Complete the following steps to install InfoScale operator.

Note: Below images are for reference only. Ensure that you select the appropriate versions wherever required.

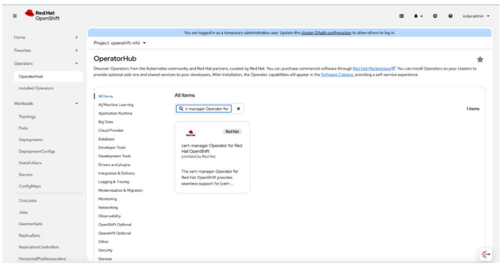
---

1 Connect to the OpenShift console and access the Catalog menu.

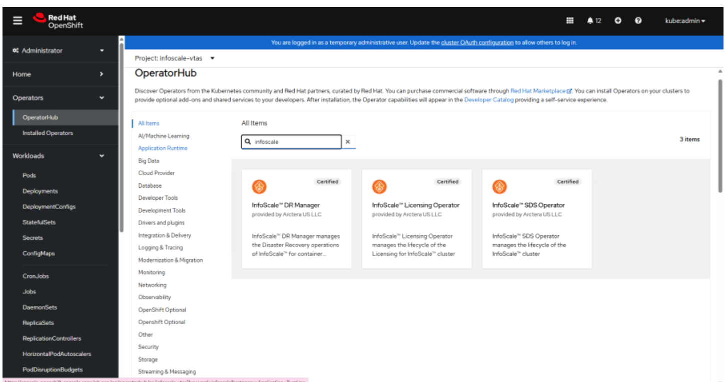
2 In the left frame, click Operators > OperatorHub. You can select and install the operator here.

3 As a prerequisite, you must download and install Cert-Manager.

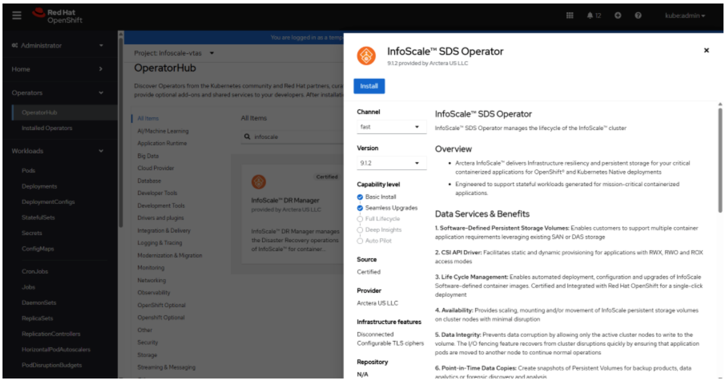
4 Enter Cert-Manager in All Items. Install the cert-manager that is listed as under.



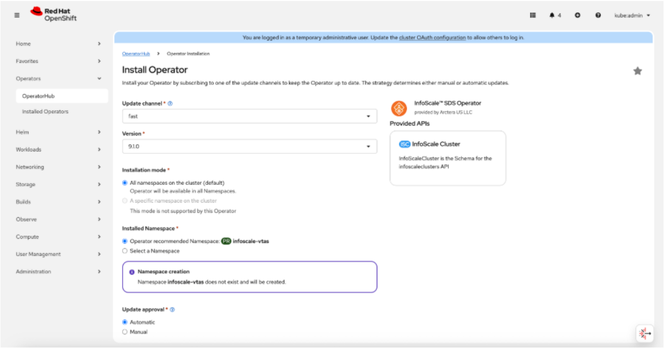
5 Enter InfoScale in All Items. The InfoScale SDS Operator, InfoScale DR Manager, and the InfoScale Licensing Operator are displayed.



6 Select the InfoScale SDS Operator and click Install in the following screen.



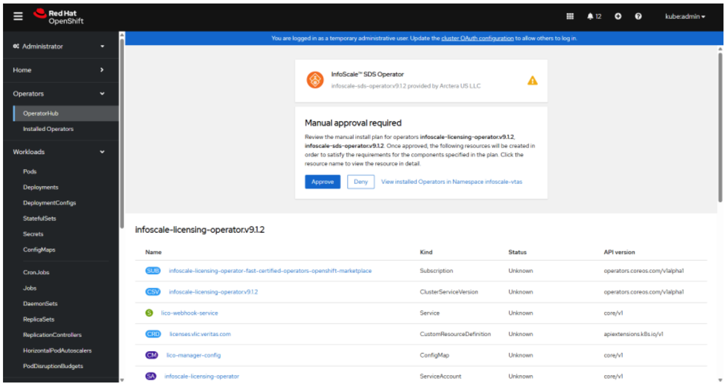
7 During download and installation of operators, when prompted ensure that fast is selected as the Update channel and All namespaces on the cluster is selected as the Installation mode as under.



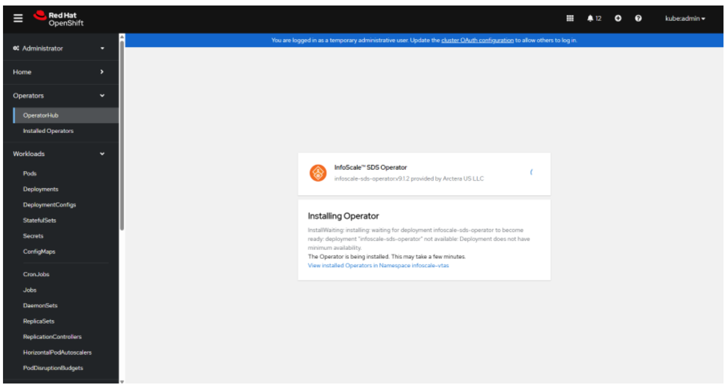
8 In the following screen, select the namespace where you want to install in Installed Namespace and Manual in Update approval. Click Install.

Note: InfoScale recommends these configurations. Selecting Manual as the Install plan avoids automatic updates of the operator.

Click Approve for Manual approval required.

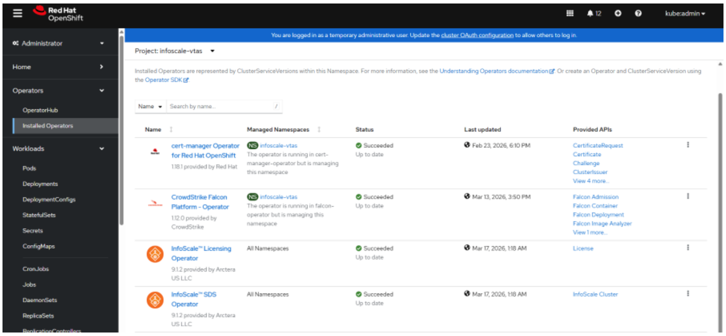


9 InfoScale installation begins as under.



10 Wait till installation is complete. InfoScale Licensing Operator along with InfoScale SDS operator are installed in the namespace you provide. Cert-manager, InfoScale Licensing Operator are the dependencies for InfoScale installation. If any of these dependencies are already installed in the namespace you provide or `openshift-operators`, installation is skipped. License must be applied before installing InfoScale.

11 In the left frame, select Operators > Installed Operators. Check if Status of all Operators is Succeeded as under.



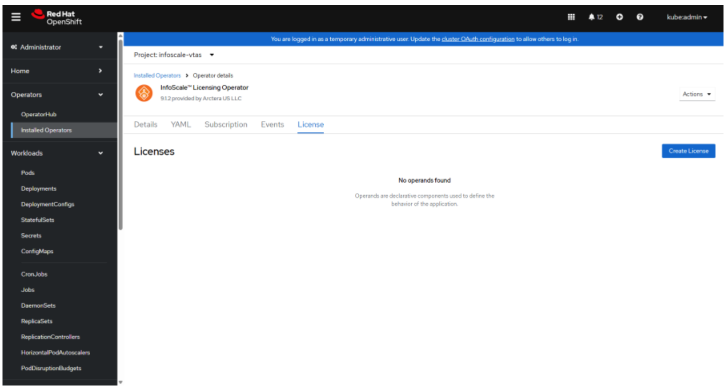
12 You can now apply the license. Click Operators > Installed Operators in the left frame.

13 Click License in Provided APIs.

Note: With a single license of Enterprise edition, you can install multiple InfoScale clusters.

---

14 In the screen that opens, click Create License in the upper-right corner of your screen.

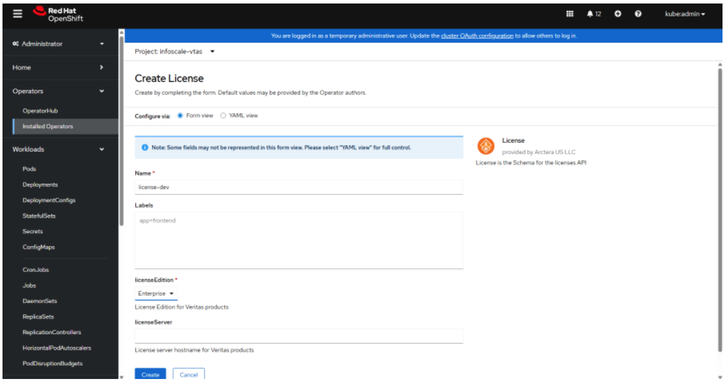


15 Assign a name. Use the cluster name to associate the license with the cluster. Optionally, Enter IP address of the VIOM server on your system in LicenseServer.

16 In License Edition, select the type of license.

See "Licensing" on page 15.

Note: If you want to install and configure Disaster Recovery (DR), you must have Enterprise type of license.



17 Click Create to apply the license.

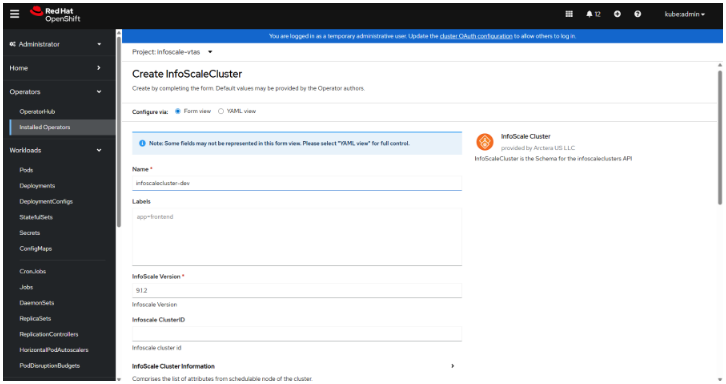
18 After the license is successfully applied, it is listed in Installed Operators > Operator details.

19 You can click the license name for details.

After the license is applied, you can configure InfoScale clusters.

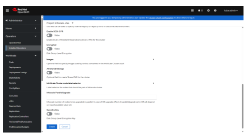
20 Click Operators > Installed Operators in the left frame. You can now create multiple InfoScale clusters. Click InfoScale Cluster in Provided APIs.

21 Click Create InfoScaleCluster in the upper-right corner of your screen. The following screen opens.



22 Choose a project to create InfoScale cluster. Enter Name, Namespace, and InfoScale ClusterID for the cluster. If you are installing in an airgapped environment, enter Image Registry for pulling images. To install on a system with Internet connectivity, do not enter any value for Image Registry.

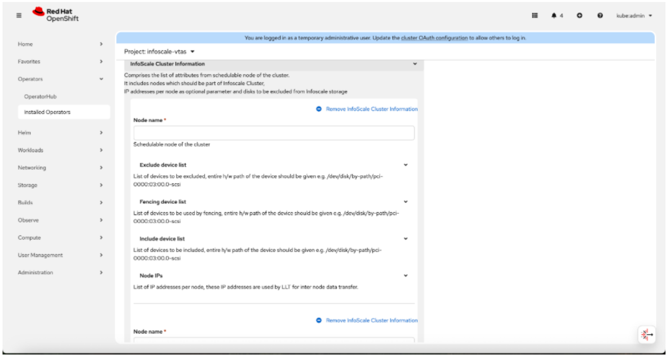
Note: If you do not enter InfoScale ClusterID, a ClusterID is randomly generated and assigned. The ClusterID is suffixed to the disk group.



To change values of parameters, See "Considerations for configuring cluster or adding nodes to an existing cluster" on page 36.

23 Click InfoScale Cluster Information. Enter information about the nodes here. Enter Node name . Optionally, you can enter IP addresses of nodes in Node IPs and the device path of the disk that you want to exclude from the InfoScale disk group in Exclude-device list. You can also add fencing devices in Fencing device list. For each node, you must add two IP addresses.

Note: If you want to add IncludeDevices for storage management instead of exclude device, refer to See "Selective storage management (IncludeDevices) " on page 33.



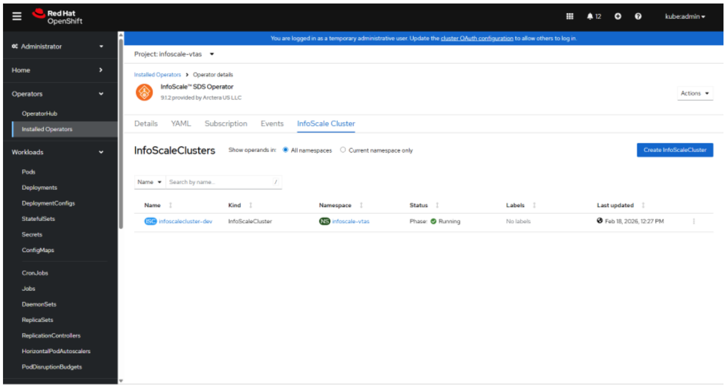
Note: OpenShift cluster must have at least two nodes as minimum two nodes are needed to form a cluster.

Note: A new attribute, includeDevices, has been introduced in the InfoScale CustomResource. This attribute allows users to specify a list of devices that will be consumed by InfoScale for Kubernetes. Only the devices listed under includeDevices will be used to create the DiskGroup. For each device that needs to be part of the DiskGroup, the complete device path (available under /dev/disk/by-path/xx) must be provided. This functionality is applicable only for new cluster deployments. Refer to See "Selective storage management (IncludeDevices) " on page 33.

---

24 To add more nodes, click Add InfoScale Cluster Information. You can add up to 16 nodes.

25 Click Create to create an InfoScale cluster. Cluster formation begins. Watch the status message. It changes to FencingConfigured,



The status then changes to DgCreated and finally Running.

26 If you want to create more clusters, run steps 21 to 25.

After a successful deployment of InfoScale, the diskgroup is automatically created.

Installing from OperatorHub by using Command Line Interface (CLI)

Complete the following steps.

Downloading `infoscale-yamls-v9.1.2.tar.gz`

1. Download `infoscale-yamls-v9.1.2.tar.gz` from the InfoScale Download Center.
2. Unzip and untar the file. A folder `/infoscale-yamls-v9.1.2/openshift/olm/` is created and all the files that are required for installation are available in the folder.

   Note: An OpenShift cluster already has a namespace `openshift-operators`. You can choose to install InfoScale in `openshift-operators`. cert-manager (Red Hat-certified) must be already installed for a successful installation of InfoScale.

   ---

3. If you have installed the cert manager in a namespace other than cert-manager, openshift-cert-manager, or openshift-operators; edit subscription yaml for `lico`, `iso` and `dr` operators and add:

```
name: CERT_MANAGER_NS
value: <namespace where cert manager is installed>
```

Optionally, you can configure a new user - infoscale-admin, that is associated with a Role-based Access Control (RBAC) clusterrole defined in `infoscale-admin-role.yaml`, to deploy InfoScale and its dependent components. infoscale-admin as a user when configured has clusterwide access to only those resources that are needed to deploy InfoScale and its dependent components such as Cert Manager in the desired namespaces.

To provide a secure and an isolated environment for InfoScale deployment and associated resources, the namespace that is associated with these resources must be protected from access of all other users (except super user of the cluster), with appropriate RBAC implemented.

Run the following commands on the bastion node to create a new user - infoscale-admin and a new project and assign role or clusterrole to infoscale-admin. You must be logged in as a super user.

Configuring a new user

1. `oc new-project <New Project name>`

   A new project is created for InfoScale deployment.

2. `oc adm policy add-role-to-user admin infoscale-admin`

   Following output indicates that administrator privileges are assigned to the new user - infoscale-admin within the new project.

   ```
   clusterrole.rbac.authorization.k8s.io/admin added: "infoscale-admin"
   ```

3. `oc apply -f /infoscale-yamls-v9.1.2/openshift/infoscale-admin-role.yaml`

   Following output indicates that a clusterrole is created.

   ```
   clusterrole.rbac.authorization.k8s.io/infoscale-admin-role created
   ```

4. `oc adm policy add-cluster-role-to-user infoscale-admin-role infoscale-admin`

   The following output indicates that a clusterrole is created and is associated with infoscale-admin.

   ```
   clusterrole.rbac.authorization.k8s.io/infoscale-admin-role added:
             "infoscale-admin"
   ```

After creating this user, you can login as `infoscale-admin` to perform all operations that are involved in installing InfoScale, configuring the cluster, and adding nodes.

Installing Operators

1. Run the following command on the bastion node.

   Note: Ignore this step if you want to install in `openshift-operators`.

   ```
   oc create namespace <Namespace>
   ```

   Review output similar to the following to verify whether the namespace is created successfully.

   ```
   namespace/<Namespace> created
   ```

2. Optionally, if you want to change the default kubelet path, edit `/infoscale-yamls-v9.1.2/openshift/olm/infoscale-sub.yaml` as under.

   ```
   env:

      - name: KUBELET_PATH

        value: <enter the new path>
   ```

   The default path is `/var/lib/kubelet`.

   Note: Do not change the kubelet path after clusters are configured.

   ---

3. Run the following command on the bastion node to create subscription.

   Note: If you want to install InfoScale in `openshift-operators`, edit `/infoscale-yamls-v9.1.2/openshift/olm/infoscale-sub.yaml`. Change namespace from <Namespace> to openshift-operators. To install the latest bundle, modify startingCSV to infoscale-sds-operator.v9.1.2.

   ```
   oc create -f
   /infoscale-yamls-v9.1.2/openshift/olm/infoscale-sub.yaml
   ```

   Following output indicates a successful command run.

   ```
   subscription.operators.coreos.com/infoscale-sds-operator created
   ```

4. Run the following command on the bastion node to deploy InfoScale licensing operator subscription.

   ```
   oc create -f
   /infoscale-yamls-v9.1.2/openshift/olm/licensing-sub.yaml
   ```

   Following output indicates a successful command run.

   ```
   subscription.operators.coreos.com
     /infoscale-licensing-operator-sub created
   ```

5. Run the following command on the bastion node to create an operator group.

   Note: Ignore this step if you want to install in `openshift-operators`.

   ```
   oc create -f
   /infoscale-yamls-v9.1.2/openshift/olm/infoscale-og.yaml
   ```

   Following output indicates a successful command run.

   ```
   operatorgroup.operators.coreos.com/infoscale-opgroup created
   ```

6. Run the following command on the bastion node.

   ```
   oc get sub,og -n <Namespace>
   ```

   Following output indicates a successful command run.

   ```
   NAME
   subscription.operators.coreos.com/infoscale-sds-operator

   PACKAGE                  SOURCE                            CHANNEL
   infoscale-sds-operator   infoscale-sds-operator-catalog    fast

   NAME                                                        AGE
   operatorgroup.operators.coreos.com/infoscale-sds-opgroup   117s
   ```

7 Run the following command on the bastion node.

```
oc get installplan -A
```

Use installation-name from the output similar to the following output.

```
NAME         NAME            CSV

<Namespace> install-k7hjl  cert-manager-operator.v1.18.0
<Namespace> install-9v2q5  infoscale-sds-operator.v9.1.2
<Namespace> install-xhbqx  nfd.4.19.0-202510211212


APPROVAL   APPROVED
Automatic  true
Manual     false
Automatic  true
```

8 Run the following command on the bastion node.

Note: Do not include the angle brackets (< >) in the command.

```
oc patch installplan --namespace <Namespace> --type merge --patch
'{"spec":{"approved":true}}'
```

Following output indicates a successful command run.

```
installplan.operators.coreos.com/<installation-name> patched
```

9 Run the following command on the bastion node.

```
oc get ip -A
```

Review output similar to the following . Check if `APPROVED` is `true`.

```
NAMESPACE     NAME            CSV

<Namespace>   install-k7hjl   cert-manager-operator.v1.18.0

<Namespace>   install-9v2q5   infoscale-sds-operator.v9.1.2

<Namespace>   install-xhbqx   nfd.4.19.0-202510211212


APPROVAL   APPROVED
Automatic  true
Manual     true
Automatic  true
```

10 Run the following command on the bastion node to check the status of csv.

```
oc get csv
```

Components which are getting installed or are pending installation are listed as follows:

```
NAME                                 DISPLAY
cert-manager-operator.v1.18.0        cert-manager Operator for Red Hat
                                     OpenShift
infoscale-licensing-operator.v9.0.1  InfoScale™ Licensing Operator
infoscale-sds-operator.v9.1.2        InfoScale™ SDS Operator


VERSION   REPLACES                          PHASE
1.18.0    cert-manager-operator.v1.17.0     Succeeded
9.0.1                                       Installing
9.1.2     infoscale-sds-operator.v9.1.2   Installing
```

11 Run the following command on the bastion node again till all operators are installed successfully.

```
oc get csv
```

Review output as under.

```
NAME                                 DISPLAY

cert-manager-operator.v1.18.0        cert-manager Operator for Red Hat
                                     OpenShift

infoscale-licensing-operator.v9.0.1  InfoScale™ Licensing Operator

infoscale-sds-operator.v9.1.2  InfoScale™ SDS Operator


VERSION   REPLACES                          PHASE

1.18.0    cert-manager-operator.v1.17.0     Succeeded

9.0.1                                       Succeeded

9.1.2     infoscale-sds-operator.v9.1.2   Succeeded


VERSION               REPLACES       PHASE
9.1.2                              Succeeded
9.1.2                              Succeeded
4.12.0-202307170916                  Succeeded
1.11-1                               Succeeded
```

12 Run the following command to check the status of all operator pods in <Namespace>.

Note: If you have installed in `openshift-operators`, run `oc get pods -n openshift-operators`.

```
oc get pods -n cert-manager
```

```
NAME                                      READY   STATUS    RESTARTS   AGE
cert-manager-759f64656b-6986d             1/1     Running   1          20d
cert-manager-cainjector-8966b78d4-wzqkf   1/1     Running   1          20d
cert-manager-webhook-597b96b99b-hmdfq     1/1     Running   1          20d
```

Applying Licenses

1 Edit `/infoscale-yamls-v9.1.2/openshift/vlic_v1_license.yaml` for the license edition. Optionally, you can change the license name. You must have Enterprise as the license edition.

See "Licensing" on page 15.

```
apiVersion: vlic.veritas.com/v1
kind: License
metadata:
  name: license-dev
spec:
  # valid licenseEdition values are Developer,
             Storage or Enterprise
  licenseEdition: "Enterprise"
  licenseServer: <Optional - IP address of
                  the VIOM server on your system>
```

2 Run `oc create -f /infoscale-yamls-v9.1.2/openshift/vlic_v1_license.yaml` on the bastion node.

3 Run `oc get licenses` on the bastion node to verify whether licenses have been successfully applied.

An output similar to the following indicates that `license_cr.yaml` is successfully applied.

```
NAME          NAMESPACE   LICENSE-EDITION   AGE
license                   ENTERPRISE        27s
```

Deploying InfoScale Cluster

1 Edit /infoscale-yamls-v9.1.2/openshift/cr.yaml.

```
---
apiVersion: infoscale.veritas.com/v1
kind: InfoScaleCluster
metadata:
  # Please change cluster name if required
  name: infoscalecluster-dev
  namespace: infoscale-vtas

spec:

  # This denotes version of the InfoScale release
  version: 9.1.2

  # (optional) This denotes the user-provisioned ID for InfoScale
     cluster
  # The value can range from 1 to 65535

  # clusterID: <Cluster_ID>

  clusterInfo:

  # Please change worker node names according to your cluster
    configuration
  # Mention additional worker node names and corresponding
    node specific
  # parameters, as applicable.

  - nodeName: "<Hostname_of_worker_node>"

    # (optional) Specifies node IP address(es) to be used for
      InfoScale cluster.
    # If omitted, InfoScale chooses available IP address(es) for
      cluster config.
    # Please change worker IP address(es) according to your cluster
      configuration

    # ip:
    # - "<IP_Address_1>"
    # - "<IP_Address_2>"

    # (optional) Specifies a node-specific list of devices for
```

```
InfoScale configuration.
   #
   # Only one of the following fields can be used at a time:
   # - `includeDevices`: List of devices to be explicitly included in
InfoScale configuration.
   # - `excludeDevice`: List of devices to be excluded from
InfoScale configuration.
   #
   # Please update the device names according to your cluster setup.

   # includeDevices:
   # - "<Hardware_Path_to_device_to_be_included>"

   # excludeDevice:
   # - "<Hardware_Path_to_device_to_be_excluded>"

   # (optional) Specifies node specific list of devices to be used
     for fencing purpose
   # It is sufficient to provide fencing device information
     from one node
   # (optional) Specifies node specific list of devices to be used
     for fencing purpose
   # It is sufficient to provide fencing device information
     from one node
   # Please change device names according to your cluster
     configuration

   # fencingDevice:
   # - "<Hardware_Path_to_device_to_be_used_for_fencing>"

 - nodeName: "<Hostname_of_worker_node>"

   # (optional) Specifies node IP address(es) to be used for
     InfoScale cluster.
   # If omitted, InfoScale chooses available IP address(es)
     for cluster config.
   # Please change worker IP address(es) according to your
     cluster configuration

   # ip:
   # - "<IP_Address_1>"
   # - "<IP_Address_2>"
```

```
    # (optional) Specifies node specific list of devices to be
      excluded from
    # InfoScale configuration.
    # Please change device names according to your cluster
      configuration

    # excludeDevice:
    # - "<Hardware_Path_to_device_to_be_excluded>"

  # Please change below customImageRegistry according to your
    environment
  # This is mandatory for Kubernetes and Air gapped system
    deployments
  # This is optional for OCP deployments

  # customImageRegistry: "<Custom_Registry_Address>"

  # (optional) Specifies whether SCSI-3 Persistent Reservation
should be enabled
  # If omitted, SCSI3-PR reserveation will be disabled by default.
  # Valid values are true or false.

  # enableScsi3pr : <true/false>

  # (optional) Specifies whether Disk Group Level Encryption
should be enabled
  # If omitted, Disk Group Level Encryption will be disabled
by default.
  # Valid values are true or false.

  # encrypted: <true|false>

  # (optional) Specifies whether Disk Group Level Encryption Key
should be same for all volumes in the DG
  # If omitted, different key will be created for each volume by
default.
  # Valid values are true or false.

  # sameEncKey: <true|false>

  # (optional) Specifies whether to create Shared NonFSS Disk Group.
  # With this option only the full shared disks will be included
in Disk Group
```

```
# If omitted, FSS Disk Group will be created by default.
# Valid values are true or false.

# isSharedStorage: <true|false>
```

You can add up to 16 nodes. To add IncludeDevices, refer to See "Selective storage management (IncludeDevices) " on page 33.

Note: Do not enclose parameter values in angle brackets (<>). For example, Primarynode is the name of the first node; for nodeName : , enter nodeName : Primarynode. InfoScale on Kubernetes is a keyless deployment.

---

2 You can choose to rename cr.yaml. If you rename the file, ensure that you use that name in the next step.

Note: InfoScale recommends renaming cr.yaml and maintaining a custom resource for each cluster. The renamed cr.yaml is used to add more nodes to that InfoScale cluster.

---

3 Run the following command on the master node.

```
oc create -f /infoscale-yamls-v9.1.2/openshift/cr.yaml
```

4 Run the following command on the master node to know the name and namespace of the cluster.

```
oc get infoscalecluster -A
```

5 Use the namespace from the output similar to the following:

```
NAMESPACE       NAME                   VERSION
infoscale-vtas  infoscalecluster-dev   9.1.2


CLUSTERID  STATE    DISKGROUPS         STATUS   AGE
1230       Running  vrts_kube_dg-1230  Healthy  72m
```

6 Run the following command on the master node to verify whether the pods are created successfully.

```
oc get pods -n infoscale-vtas
```

7 An output similar to the following indicates a successful creation of nodes.

```
NAME                                          READY   STATUS
infoscale-csi-controller-8c5bfcdbd-g9jzp      5/5     Running
infoscale-csi-node-6pt78                      2/2     Running
infoscale-csi-node-bczgc                      2/2     Running
infoscale-csi-node-cbxkf                      2/2     Running
infoscale-csi-node-k7l7w                      2/2     Running
infoscale-csi-node-mk48n                      2/2     Running
infoscale-fencing-controller-566dc674fb-ghqbg 1/1     Running
infoscale-fencing-enabler-6688s               1/1     Running
infoscale-fencing-enabler-hrqgr               1/1     Running
infoscale-fencing-enabler-n728h               1/1     Running
infoscale-fencing-enabler-qsbdf               1/1     Running
infoscale-fencing-enabler-tm2zk               1/1     Running
infoscale-licensing-operator-786df478b-5snwz  1/1     Running
infoscale-sds-1230-f836b69ee261cd15-c4nsq     1/1     Running
infoscale-sds-1230-f836b69ee261cd15-fnj6k     1/1     Running
infoscale-sds-1230-f836b69ee261cd15-j58rj     1/1     Running
infoscale-sds-1230-f836b69ee261cd15-m5597     1/1     Running
infoscale-sds-1230-f836b69ee261cd15-nhhm9     1/1     Running
infoscale-sds-operator-5696f66584-vclc4       1/1     Running
```

```
infoscale-toolset-1230-d596bb7bf-4ldqx          1/1     Running

infoscale-toolset-1230-d596bb7bf-9n4zq          1/1     Running

infoscale-toolset-1230-d596bb7bf-jqnx6          1/1     Running

infoscale-toolset-1230-d596bb7bf-r25j5          1/1     Running

infoscale-toolset-1230-d596bb7bf-zkwjr          1/1     Running


RESTARTS   AGE
0          3m22s
0          3m22s
0          3m22s
0          3m22s
0          3m22s
0          3m22s
0          3m23s
0          3m23s
0          3m23s
0          3m23s
0          3m23s
0          3m23s
0          135m
0          3m23s
0          3m23s
0          3m23s
0          3m23s
0          3m23s
0          135m
0          3m23s
0          3m23s
0          3m23s
0          3m23s
0          3m23s
```

After a successful InfoScale deployment, a disk group is automatically created.

8 You can now create Persistent Volumes/Persistent Volume Claims (PV/PVC) by using the corresponding storage class.

See "Adding nodes to an existing cluster" on page 261.

See "Removing nodes from an existing cluster" on page 267.

Using InfoScale storage with OpenShift virtualization

VIKE is fully qualified for deployment with Red Hat OpenShift virtualization. InfoScale CSI StorageClasses and SnapshotClasses can be used to provision virtual machine disks that provide high performance, enterprise-level reliability, and consistent behavior across container and VM workloads.

Refer to https://www.veritas.com/support/en_US/article.100074725 to know more about configuration guidelines for OpenShift VMs.

The following is an example of a storage profile that references an InfoScale-based StorageClass and SnapshotClass to create virtual machines.

Note: The cloneStrategy can be either snapshot or clone. If snapshot is chosen as the cloneStrategy, the user must create a VolumeSnapshotClass in advance, after deploying InfoScale.

Note: Creating a large number of VMs (500+) in a single batch is not recommended. We suggest creating a smaller set of 4-5 VMs at a time and ensuring that each VM reaches the Running state before initiating the creation of the next batch.

Sample VolumeSnapshotClass yaml

These yamls can be found at:
/infoscale-yamls-v9.1.2/infoscale-csi/dynamic-provisioning

```
---
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-infoscale-snapclass
  annotations:
    snapshot.storage.kubernetes.io/is-default-class: "true"
driver: org.veritas.infoscale
```

```
deletionPolicy: Delete
parameters:
  # (optional) Specifies the type of the snapshot to be created.
  # If omitted, InfoScale by default creates space-optimized snapshot.
  # Supported values: space-optimized, full-instant
   snapType: space-optimized
  # (optional) Specifies the size of the cache volume to be created
   for space-optimized snapshots.
  # If omitted, InfoScale internally chooses the cacheSize as 30% of
   orignial volume size.
   cacheSize: 5g
```

Sample StorageClass yaml:

These yamls can be found at: /infoscale-yamls-v9.1.2/infoscale-csi/storage-class-templates

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-infoscale-sc-block
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: org.veritas.infoscale
reclaimPolicy: Delete
allowVolumeExpansion: true
parameters:
  # (optional) Specifies a volume layout type.
  # Supported layouts: stripe, mirror, stripe-mirror, mirror-stripe, concat,
concat-mirror, mirror-concat  layout: "concat"
  #nstripe: "4"
  # (optional) Specifies the number of disk or host failures a storage object
can tolerate.
  #faultTolerance: "1"
  # (optional) Specifies disks with the specified media type. All disks with
the given mediatype are selected for volume creation.
  # Supported values: hdd, ssd
  mediaType: "hdd"
```

Creating OpenShift virtual machines using InfoScale-backed storage

1. Ensure that the InfoScale cluster is deployed and healthy by running `oc get pods -n infoscale-vtas`.

```
NAME                                           READY   STATUS    RESTART
infoscale-csi-controller-84488fd788-knfgf      5/5     Running   0
infoscale-csi-node-m27cc                       2/2     Running   0
infoscale-csi-node-mtc57                       2/2     Running   0
infoscale-csi-node-sk2jd                       2/2     Running   0
infoscale-fencing-controller-777ddd5cc8-qtjkc  1/1     Running   0
infoscale-fencing-enabler-mv5wr                1/1     Running   0
infoscale-fencing-enabler-pmhcb                1/1     Running   0
infoscale-fencing-enabler-tb69x                1/1     Running   0
infoscale-licensing-operator-bd5bd97f5-46wwh   1/1     Running   0
infoscale-sds-35322-f836b69ee261cd15-gv5b2     1/1     Running   0
infoscale-sds-35322-f836b69ee261cd15-lzxfr     1/1     Running   0
infoscale-sds-35322-f836b69ee261cd15-zstw6     1/1     Running   0
infoscale-sds-operator-d4546c849-mttwc         1/1     Running   0
infoscale-toolset-35322-8988985cf-hlmws        1/1     Running   0
infoscale-toolset-35322-8988985cf-kxqbv        1/1     Running   0
infoscale-toolset-35322-8988985cf-njpth        1/1     Running   0
[root@ocp319-ba1]# oc get infoscalecluster -A
NAMESPACE      NAME                       VERSION CLUSTERID STATE
infoscale-vtas infoscalecluster-dev-new 9.1.2   35322     Running
```

```
DISKGROUPS         STATUS  AGE
vrts_kube_dg-35322 Healthy 17h
```

2. Create a StorageClass using the sample template below. Any valid StorageClass configuration from the available options can be used.

   Note: Make the infoscale-concat-sc StorageClass the default by adding the appropriate annotation.

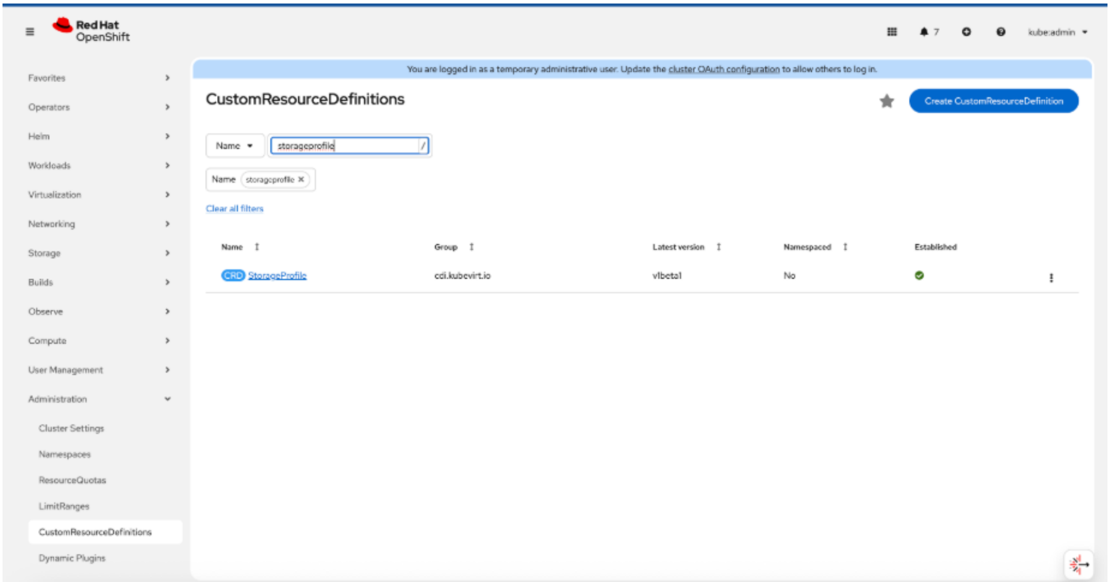
   ```
   ---
   apiVersion: storage.k8s.io/v1
   kind: StorageClass
   metadata:
     name: infoscale-concat-sc
     annotations:
       storageclass.kubernetes.io/is-default-class: "true"
   provisioner: org.veritas.infoscale
   reclaimPolicy: Delete
   allowVolumeExpansion: true
   parameters:
     layout: "concat"
     mediaType: "hdd"
   ```

3. Run `oc create -f infoscale-concat-sc.yaml`.
4. Install OpenShift Virtualization Operator and create HyperConverged from UI.

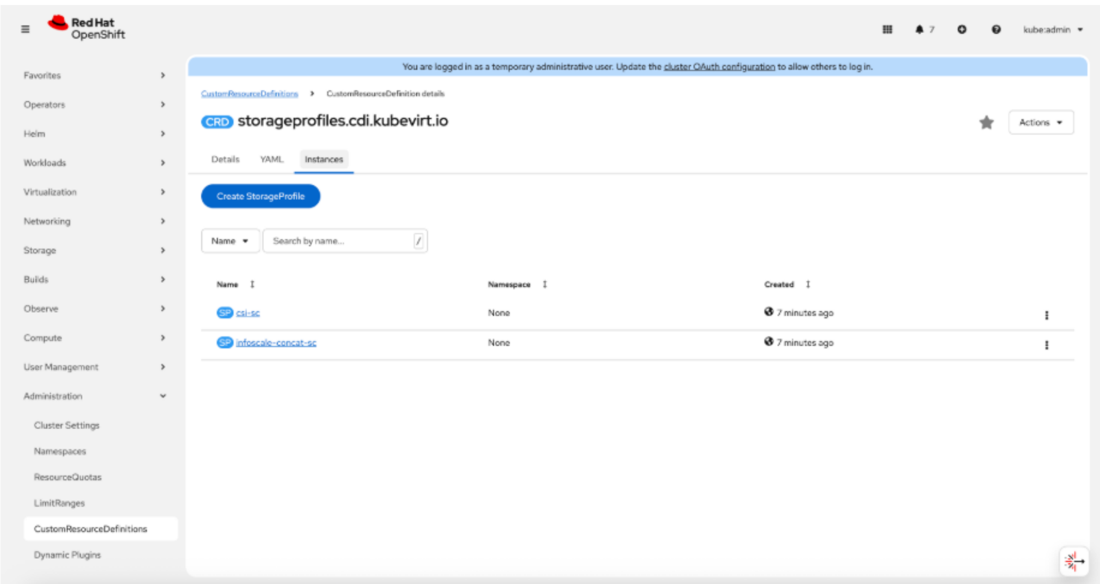
   ```
   Conditions:ReconcileComplete, Available, Upgradeable
   ```

5. Before creating VMs, the StorageClasses within the storage profile for OpenShift Virtualization must be modified. The StorageClasses in the storage profile are copies of those defined in Step 3.

   - Navigate to OpenShift Console > Administration > CustomResourceDefinitions and search for storageprofile.



- Click StorageProfile, then select the Instances tab. As mentioned earlier, when the OpenShift Virtualization operators were installed, copies of the StorageClasses defined in the previous section were created.



- Click the StorageClass that you previously created and annotated as the default, then modify the spec section in the storageprofiles.cdi.kubevirt.io CustomResourceDefinition. Open infoscale-concat-sc, switch to the YAML tab, scroll to the bottom of the file, and delete all lines starting from spec:{} to the end. After removing them, insert the lines shown below if you want to use cloneStrategy as csi-clone.

  ```
  spec:
    claimPropertySets:
    - accessModes:
        - ReadWriteMany
      volumeMode: Block
      cloneStrategy: csi-clone
  ```

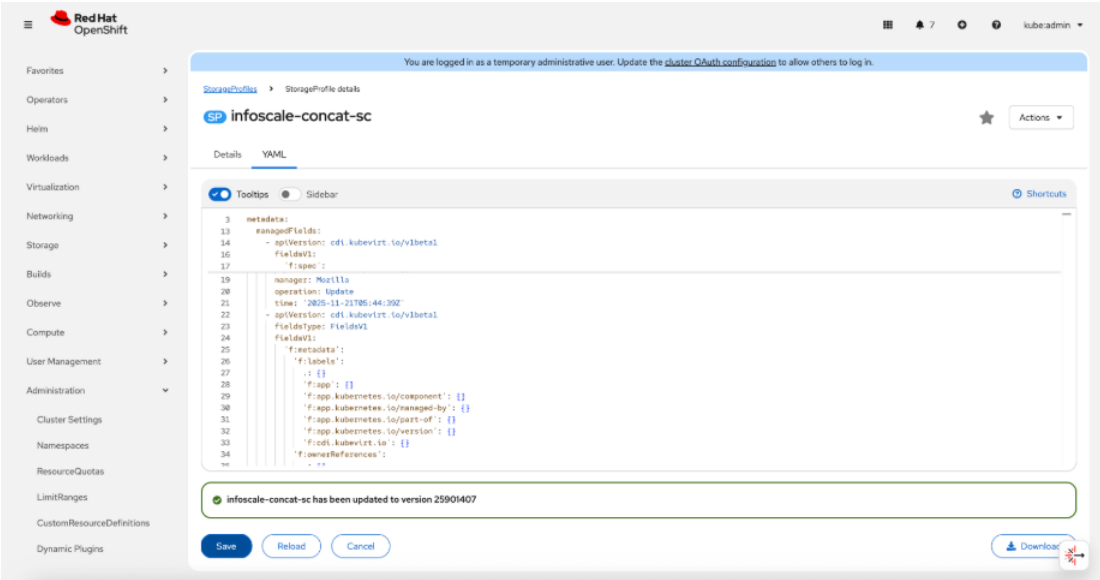
  Insert the lines below if you want to use cloneStrategy as snapshot. Specify the name of the VolumeSnapshotClass you created earlier in the snapshotClass field.

  ```
  spec:
    claimPropertySets:
    - accessModes:
       - ReadWriteMany
      volumeMode: Block
      cloneStrategy: snapshot
      snapshotClass: csi-infoscale-snapclass
  ```

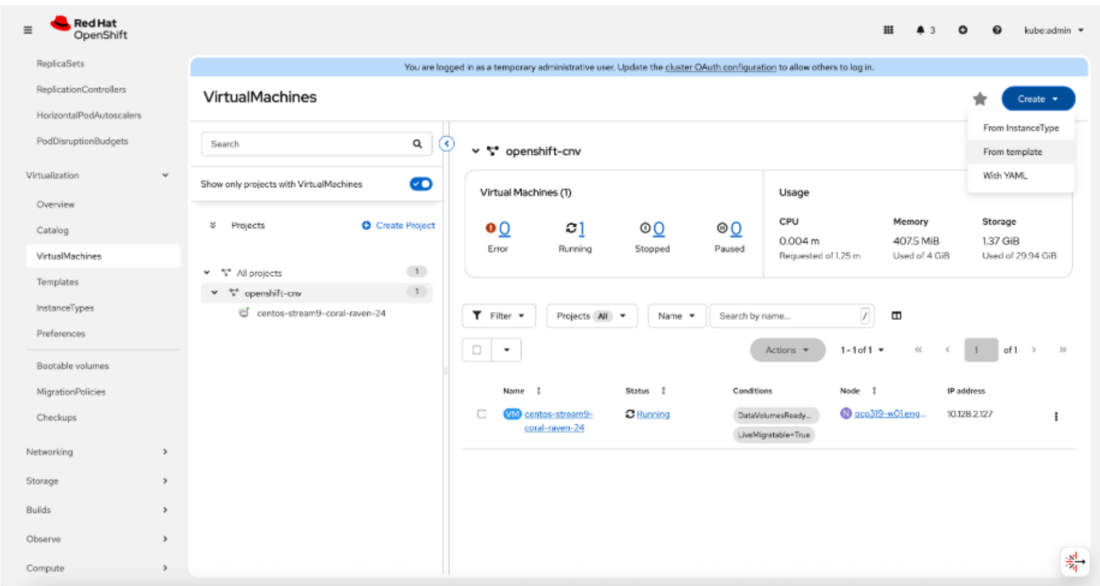
  To create a virtual machine using InfoScale-provisioned storage, the user must select the appropriate StorageClass for both the root disk and the data disk.

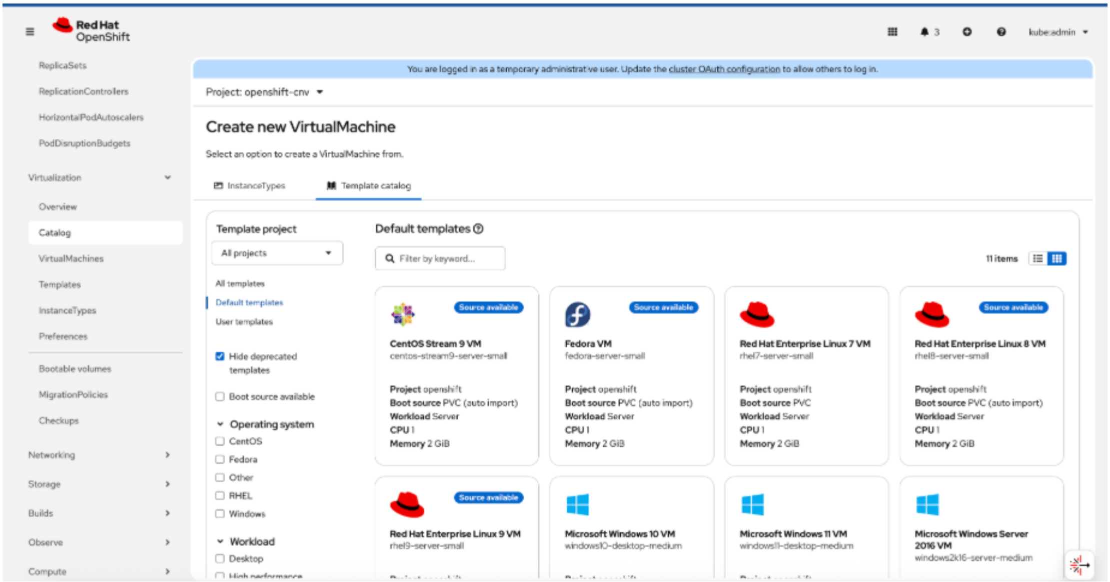
- Click Save, and then Reload.

  Note: After reloading, the line storageClass: infoscale-concat-sc is added again.

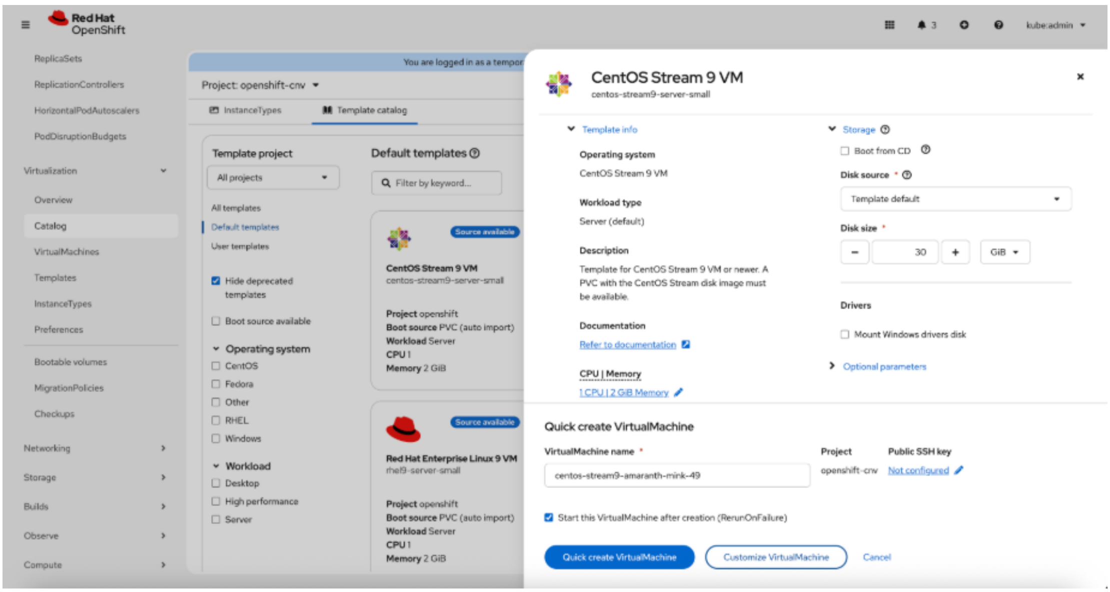


6. Navigate to Virtualization to create a new virtual machine.

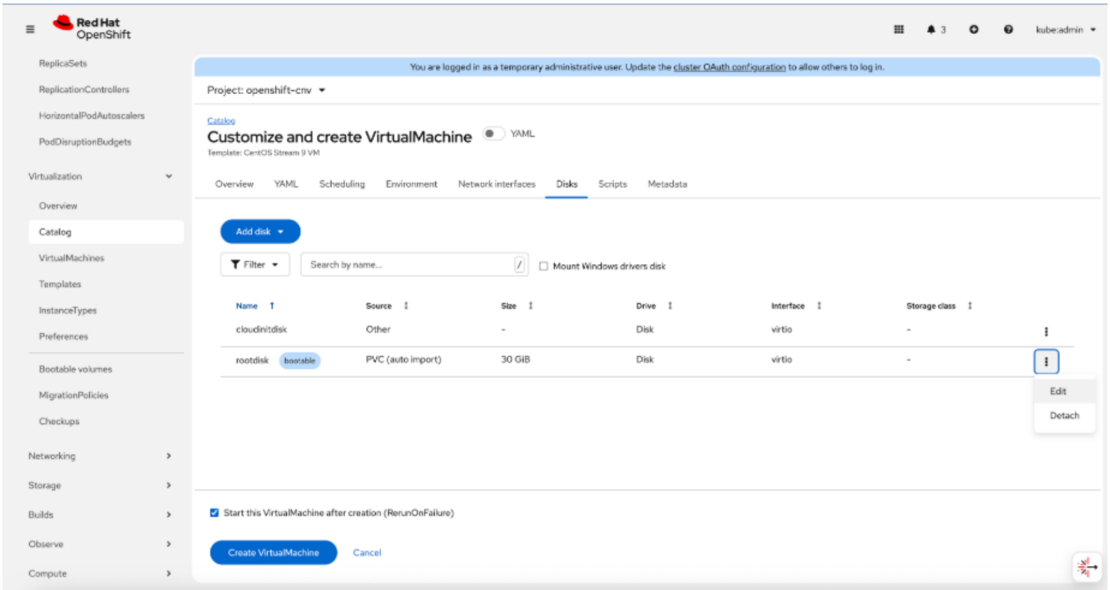




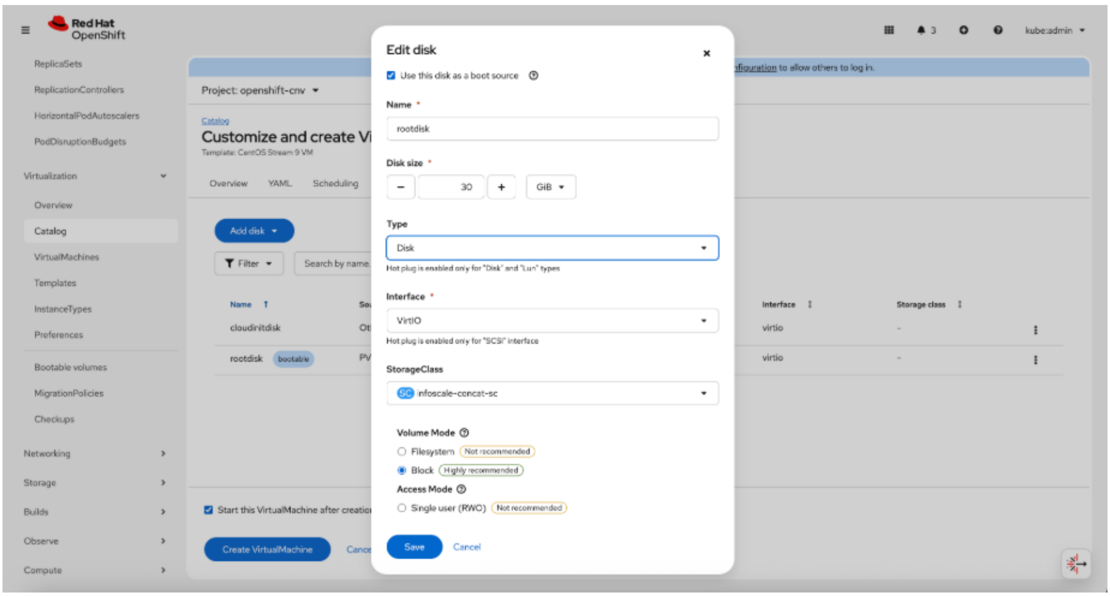
7. Select the boot source and click Customize VirtualMachine.



8. Go to Disks and click Edit.

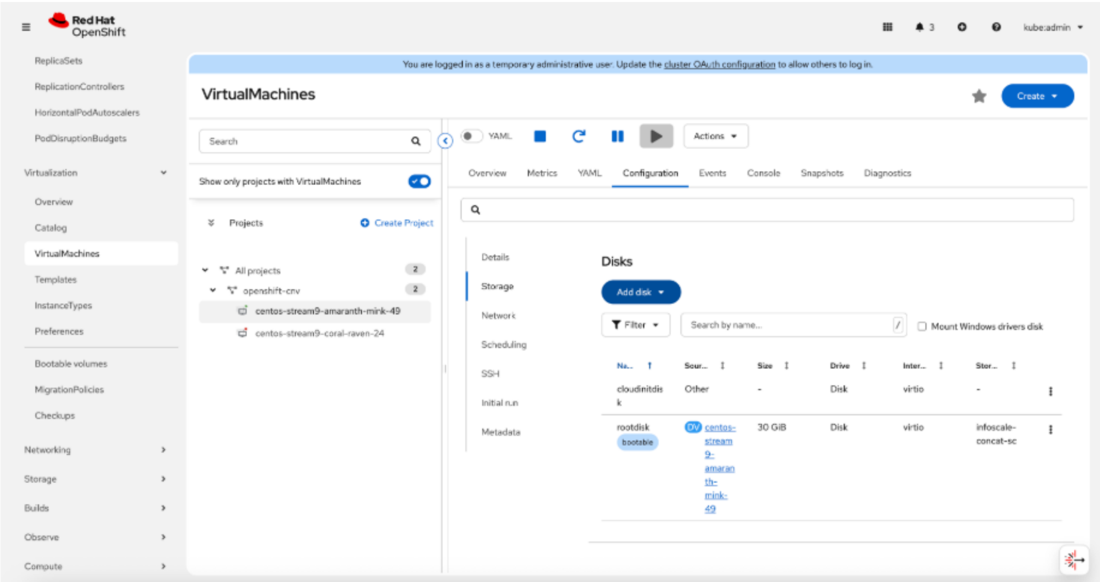


9. Specify the InfoScale StorageClass for both the boot disk and the data disk, then click Save. After that, select Create VirtualMachine.

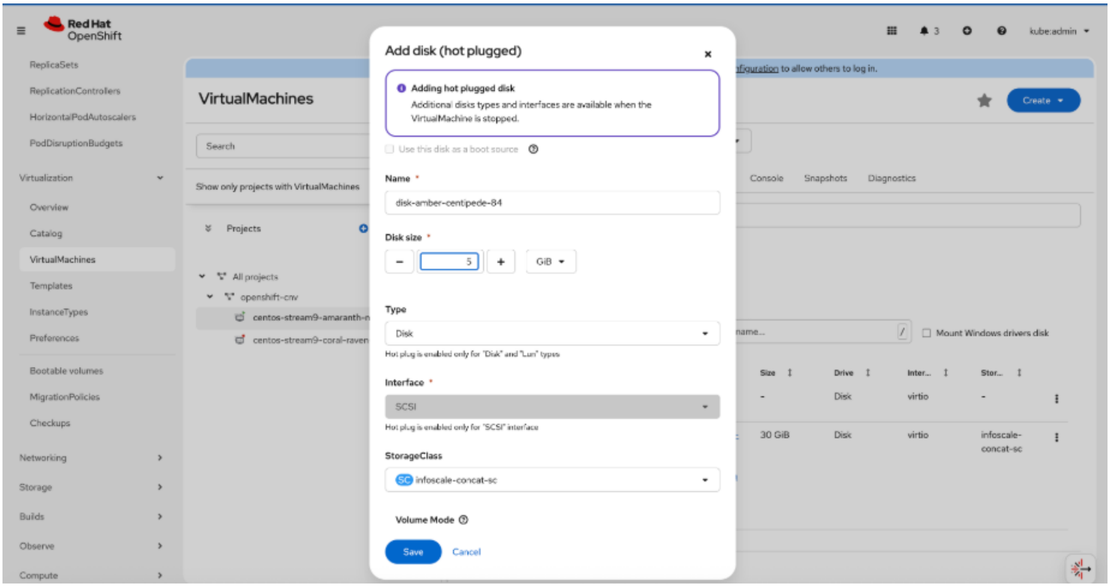


This creates a virtual machine which has InfoScale backed PVC provisioned rootdisk.

10. Select VirtualMmachines > Configuration > Storage > Add disk.



11. Mention the InfoScale storage class while adding the data disk.



InfoScale for Kubernetes support for Two-Node Arbiter (TNA) clusters

Two-Node Arbiter (TNA) capability was introduced by Red Hat in OCP 4.19. InfoScale fully supports running VIKE on TNA-based OpenShift clusters. InfoScale components work correctly within Red Hat's TNA model. This includes deployment, storage provisioning, and fencing workflows.

Support has been confirmed through extensive testing using external SAN disks for both data and fencing, validating interoperability and reliability across OCP 4.19 and 4.20 environments.

To know more, refer to this Red Hat blog.

Prerequisites and compatibility requirements

TNA clusters require a specific installation workflow that is fully supported and validated by Red Hat.

Configuration

- No additional configuration is required for VIKE to operate on TNA clusters.

- The arbiter node remains empty (no VIKE pods, no drivers, no CSI components).
- VIKE automatically detects node roles and schedules workloads only on the two full nodes.
- Built-in arbiter node taints ensure no storage or application workloads are placed on it.

Installation

- The arbiter node must not be included in the InfoScale cluster membership.
- Only the two worker nodes should participate as InfoScale cluster nodes.
- Standard IKE installation steps must be followed with no custom overrides.

Chapter 5

Installing InfoScale on Kubernetes

This chapter includes the following topics:

Introduction

This chapter informs you how to install InfoScale on a Kubernetes cluster.

On Kubernetes systems, installer files and container images must be downloaded from the InfoScale Download Center. Commands are run from the master node of a Kubernetes cluster.

Note: You can form multiple InfoScale clusters. Each cluster can have upto 16 nodes. InfoScale is deployed on all the nodes you specify in the Custom Resource yaml file. With HyperConverged Architecture supported, a stateful application consuming storage from an InfoScale cluster must run on any of the nodes that are part of that InfoScale cluster.

Note: If you previously installed an older version of InfoScale and are reinstalling while reusing the same Cluster ID and DiskGroup, we recommend that you reboot all nodes with reboot -f before proceeding. This clears stale configurations and helps prevent conflicts during reinstallation.

Prerequisites

1. Be ready with the following information -
   - Names of all the nodes.

     Note: Run `kubectl get nodes -o wide` on the master node to obtain Names and IP addresses of the nodes.

     Use `NAME` and `INTERNAL-IP` from the output similar to the following -

     ```
     NAME     STATUS ROLES  AGE VERSION         INTERNAL-IP
     k8s-cp-1.lab.k8s.lan Ready  master 75d v1.20.0+558d959 192.168.22.201
     k8s-cp-2.lab.k8s.lan Ready  master 75d v1.20.0+558d959 192.168.22.202
     k8s-cp-3.lab.k8s.lan Ready  master 75d v1.20.0+558d959 192.168.22.203
     k8s-w-1.lab.k8s.lan  Ready  worker 75d v1.20.0+558d959 192.168.22.211
     ```

   - Operating system hardware path of the disks which are being managed by other storage vendors that need to be excluded from InfoScale disk group.
   - Optionally if you want to exclude boot disks, device path to the boot disks.

     Note: InfoScale recommends excluding boot disks.

   - Custom Registry address to set up registry where InfoScale images are pushed.
2. Ensure that all nodes are synchronized with the NTP Server.
3. Reserve network ports for exclusive use of InfoScale as under -

Component	Port
LLT over UDP	Serially onwards 50000 (as many as configured LLT links)
VVR (Needed only if you want to configure DR)	4145 (UDP), 8199 (TCP), 8989 (TCP)

4. Add local or shared storage to all the worker nodes before you proceed with the deployment.
5. Ensure that stale InfoScale kernel modules (`vxio`,`vxdmp`,`veki`,`vxspec`,`vxfs`,`odm`,`glm`,`gms`) from previous installation do not exist on any of the worker nodes.

   Note: You can reboot a worker node to unload all stale InfoScale kernel modules.

6. Hostnames of the InfoScale nodes must exactly match with the fully qualified domain names(FQDN) of the Kubernetes nodes.
7. SE Linux is disabled on all the nodes
8. Kubernetes cluster must be configured by using IPv4 only.
9. Optionally, add custom ciphers to `kube-apiserver`. Refer to Kubernetes documentation to know how to add ciphers.
10. `git` must be installed on the master node.

Installing Cert-Manager on Kubernetes

Complete the following steps to enable Cert-Manager on Kubernetes.

1. Run the following command on the master node to install Cert-Manager:

```
kubectl apply -f
https://github.com/cert-manager/cert-manager/releases/download/v1.12.0/cert-manager.yaml
```

Downloading Installer

Complete the following steps to download the installer file.

1 Download `infoscale-yamls-v9.1.2.tar.gz` from the InfoScale Download Center.

2 Unzip and untar the file. The folders `/infoscale-yamls-v9.1.2/openshift/`, `/infoscale-yamls-v9.1.2/DR`, and `/infoscale-yamls-v9.1.2/kubernetes` are created. Each folder contains files required for installation.

Tagging the InfoScale images on Kubernetes

Complete the following steps to upload and tag InfoScale images in the private registry/repository and prepare the cluster for installing InfoScale.

Prerequisites

- You must have a docker registry or you must set up a new docker registry.
- If the registry is insecure, your Kubernetes nodes must be configured to access the registry by using `http` method.

Complete the following steps

- Download `Arctera_InfoScale_9.1.2_Containers_<OS>_Linux.tar.gz` and `setup_vtas_registry.sh` from the InfoScale Download Center to a node where the private repository is configured.

  Note: `setup_vtas_registry.sh` is available in `infoscale-tools_v9.1.2.tar.gz` .

- Run the following command on this node
  ```
  setup_vtas_registry.sh -c <IP address of custom registry>:<port number>/vtas_test -t
  Arctera_InfoScale_9.1.2_Containers_<OS>_Linux.tar.gz
  ```

After this command is successfully run, you have the ISO image as `<IP address of custom registry>:<port number>/vtas_test/infoscale-sds-operator-9.1.2-<rhel>` and CR custom registry as `<IP address of custom registry>:<port number>/vtas_test`.

Alternatively, you can download `Arctera_InfoScale_9.1.2_Containers_<OS>_Linux.tar.gz` and individually load, tag, and push each image file. See the following steps.

1. Download `Arctera_InfoScale_9.1.2_Containers_<OS>_Linux.tar.gz` and run the following command to extract the tar file.

tar -xvf Arctera_InfoScale_9.1.2_Containers_<OS>_Linux.tar.gz

2. After you unzip and untar Arctera_InfoScale_9.1.2_Containers_<OS>_Linux.tar.gz, you get the following image files
   - infoscale-storage-operator-9.1.2-<os-version>.img
   - infoscale-dr-operator-9.1.2-<os-version>.img
   - infoscale-license-operator-9.1.2-<os-version>.img
   - infoscale-sds-toolset-9.1.2-<os-version>.img
   - infoscale-sds-core-9.1.2-<os-version>.img
   - infoscale-sds-csi-9.1.2-<os-version>.img
   - infoscale-sds-fencing-9.1.2-<os-version>.img

   You must load, tag, and push each image file into the custom registry.

   Note: The following commands are applicable to docker as the runtime environment. These commands change as per your container runtime environment. Refer to the documentation for the equivalent commands.

   ---

3. For infoscale-storage-operator-9.1.2-<os-version>.img, run the following commands-
   - docker load -i infoscale-storage-operator-9.1.2-<os-version>.img
   - docker tag localhost/arctera/infoscale-storage-operator-9.1.2-<os-version> <IP address of custom registry>:<port number>/infoscale-storage-operator-9.1.2-<os-version>
   - docker push <IP address of custom registry>:<port number>/ infoscale-storage-operator-9.1.2-<os-version>
4. For infoscale-license-operator-9.1.2-<os-version-major>.img, run the following commands-
   - docker load -i infoscale-license-operator-9.1.2-<os-version-major>.img
   - docker tag localhost/infoscale-license-operator:9.1.2-<os-version-major>

`<IP address of custom registry>:<port number>/ infoscale-license-operator-9.1.2-<os-version-major>`

- `docker push <IP address of custom registry>:<port number>/ infoscale-license-operator-9.1.2-<os-version-major>`

5. For `infoscale-sds-core9.1.2-<os-version>.img` , run the following commands

Note: You must be ready with the kernel version of the Operating system on each worker node. You can run `uname -r` to know the kernel version. If worker nodes have different kernel versions, you must run the following commands separately for worker nodes with identical kernel versions.

- `docker load -i infoscale-sds-core9.1.2-<os-version>.img`
- `docker tag localhost/infoscale-sds-core:9.1.2-<os-version>-<kernel-version> <IP address of custom registry>:<port number>/infoscale-sds-core:9.1.2-<os-version>-<kernel-version>`
- `docker push <IP address of custom registry>:<port number>/infoscale-sds-core:9.1.2-<os-version>-<kernel-version>`

6. For `infoscale-sds-fencing-9.1.2-<os-version-major>.img`, run the following commands -

- `docker load -i infoscale-sds-fencing-9.1.2-<os-version-major>.img`
- `docker tag localhost/arctera/infoscale-sds-fencing-9.1.2-<os-version-major> <IP address of custom registry>:<port number>/infoscale-fencing:9.1.2-<os-version-major>`
- `docker push <IP address of custom registry>:<port number>/ infoscale-sds-fencing-9.1.2-<os-version-major>`

7. For `infoscale-sds-csi-9.1.2-<os-version-major>.img`, run the following commands -

- `docker load -i infoscale-sds-csi-9.1.2-<os-version-major>.img`
- `docker tag localhost/arctera/infoscale-sds-csi-9.1.2-<os-version-major>`

```
<IP address of custom registry>:<port
number>/infoscale-sds-csi-9.1.2-<os-version-major>
```

- ```
  docker push
  localhost/arctera/infoscale-sds-csi-9.1.2-<os-version-major>
  <IP address of custom registry>:<port
  number>/infoscale-sds-csi-9.1.2-<os-version-major>
  ```

Downloading side car images

The following table lists the side car images for the CSI plugin and fencing containers. You must download the CSI plugin-related images from https://github.com/kubernetes-csi and the fencing-related images from https://github.com/kvaps/kube-fencing.

Note: Perform these steps if you are manually tagging images instead of using `setup_vtas_registry.sh` for tagging images.

Table 5-1 Image names and sources for side car containers

Image name	Source
csi-snapshotter	registry.k8s.io/sig-storage/csi-snapshotter
csi-provisioner	registry.k8s.io/sig-storage/csi-provisioner
csi-resizer	registry.k8s.io/sig-storage/csi-resizer
csi-node-driver-registrar	registry.k8s.io/sig-storage/csi-node-driver-registrar
kube-fencing-switcher	ghcr.io/kvaps/kube-fencing-switcher
kube-fencing-controller	ghcr.io/kvaps/kube-fencing-controller
ose-csi-external-snapshotter-rhel9	registry.redhat.io/openshift4/ose-csi-external-snapshotter-rhel9:v4.21
ose-csi-external-resizer-rhel9	registry.redhat.io/openshift4/ose-csi-external-resizer-rhel9:v4.21
ose-csi-external-provisioner-rhel9	registry.redhat.io/openshift4/ose-csi-external-provisioner-rhel9:v4.21
ose-csi-node-driver-registrar-rhel9	registry.redhat.io/openshift4/ose-csi-node-driver-registrar-rhel9:v4.21
  ```

Table 5-2 Image names with tags for side car containers

Image name	Tag	Required tag
csi-snapshotter	v8.5.0	<IP address of custom registry>:<port number>/csi-snapshotter:v8.5.0
csi-provisioner	v6.1.1	<IP address of custom registry>:<port number>/csi-provisioner:v6.1.1
csi-resizer	v2.1.0	<IP address of custom registry>:<port number>/csi-resizer:v2.1.0
csi-node-driver-registrar	v2.15.0 v2.16.0	<IP address of custom registry>:<port number>/csi-node-driver-registrar:v2.15.0 <IP address of custom registry>:<port number>/csi-node-driver-registrar:v2.16.0
kube-fencing-switcher	v2.4.0	
kube-fencing-controller	v2.4.0	<IP address of custom registry>:<port number>/kube-fencing-controller:v.2.4.0
ose-csi-external-snapshotter-rhel9	v4.19 v4.21	<IP address of custom registry>:<port number>/ose-csi-external-snapshotter-rhel9:v4.19 <IP address of custom registry>:<port number>/ose-csi-external-snapshotter-rhel9:v4.21
ose-csi-external-resizer-rhel9	v4.19 v4.21	<IP address of custom registry>:<port number>/ose-csi-external-resizer-rhel9:v4.19 <IP address of custom registry>:<port number>/ose-csi-external-resizer-rhel9:v4.21
ose-csi-external-provisioner-rhel9	v4.19 v4.21	<IP address of custom registry>:<port number>/ose-csi-external-provisioner-rhel9:v4.19 <IP address of custom registry>:<port number>/ose-csi-external-provisioner-rhel9:v4.21
ose-csi-node-driver-registrar-rhel9	v4.19 v4.21	<IP address of custom registry>:<port number>/ose-csi-node-driver-registrar-rhel9:v4.19 <IP address of custom registry>:<port number>/ose-csi-node-driver-registrar-rhel9:v4.21

You must pull these images, tag correctly, and subsequently push the tagged images to the custom registry. The commands are in the following format.

- Pulling the image

```
docker pull <source from the above table>:<tag from the above
table>
```

- Correctly tagging the image

  ```
  docker tag <source from the above table>:<tag from the above table>
  <Required tag from the above table>
  ```

- Pushing the image

  ```
  docker push <Required tag from the above table>
  ```

An example of commands for csi-snapshotter is as under -

- docker pull registry.k8s.io/sig-storage/csi-snapshotter:v8.4.0
- docker tag registry.k8s.io/sig-storage/csi-snapshotter:v8.4.0 <IP address of custom registry>:<port number>/csi-snapshotter:v8.4.0
- docker push <IP address of custom registry>:<port number>/csi-snapshotter:v8.4.0

Run these commands on all the images.

Applying licenses

To apply InfoScale licenses

1. Run kubectl apply -f /infoscale-yamls-v9.1.2/kubernetes/lico_rhel.yaml on the master node.
2. Run kubectl get pods -A|grep -i licensing on the master node to verify whether lico.yaml is successfully applied.

   An output similar to the following indicates that lico.yaml is successfully applied.

   ```
   infoscale-vtas infoscale-licensing-operator-cfff86577-kcpnl 1/1 Running 1
   ```

3. After lico.yaml is successfully applied, wait till the licensing endpoints are activated.

   Run kubectl describe service/lico-webhook-service -n infoscale-vtas|grep Endpoints on the master node and review the output.

4. Run the command again till you get an output in the following format.

   ```
   Endpoints: <IP address of the endpoint>:<Port number>
   ```

5 Edit `/infoscale-yamls-v9.1.2/kubernetes/vlic_v1_license.yaml` for the Enterprise license edition.

```
apiVersion: vlic.veritas.com/v1
kind: License
metadata:
name: <You can change the name here>
spec:
# valid licenseEdition values are Developer,
 Storage or Enterprise
licenseEdition: "Developer"
licenseServer: <Optional - IP address of
                  the VIOM server on your system>
```

6 Run `kubectl create -f /infoscale-yamls-v9.1.2/kubernetes/vlic_v1_license.yaml` on the master node.

7 Run `kubectl get licenses` on the master node to verify whether `license_cr.yaml` is successfully applied.

An output similar to the following indicates that `license_cr.yaml` with Developer as the `license-edition` is successfully applied.

```
NAME          NAMESPACE   LICENSE-EDITION   AGE
license-dev               ENTERPRISE        75m
```

Considerations for configuring cluster or adding nodes to an existing cluster

You can specify up to 16 worker nodes in `cr.yaml`. Although cluster configuration is allowed even with one Network Interface Card, InfoScale recommends a minimum of two physical links for performance and High Availability (HA). Number of links for each network link must be same on all nodes. Optionally, you can enter node level IP addresses. If IP addresses are not provided, IP addresses of Kubernetes cluster nodes are used.

By default, Flexible Storage Sharing (FSS) disk group is created. If shared storage is the only storage available across nodes you can set `isSharedStorage` to true and fully shared non-FSS disk group is created while configuring clusters. Mirroring is performed across enclosures thus ensuring redundancies. While using shared storage, InfoScale recommends changing the default majority-based fencing to disk-based fencing. With shared storage and majority-based fencing, storage becomes inaccessible when majority of the nodes go down; although all disks are connected to all the nodes. With disk-based fencing, storage is available to applications even when one node is up. To configure disk-based fencing, specify hardware path of the fencing disks for at least one node. InfoScale recommends using at least three disks for fencing purpose. Ensure that the number of disks is odd in number.

Note: For Disaster Recovery (DR) configuration, only Volume level encryption is supported. Disk group level encryption is not supported.

Creating multiple InfoScale clusters

After successfully installing InfoScale operator, you can create a cluster.

You can configure multiple clusters in different namespaces. After a cluster is configured and pods are created successfully, you can update `cr.yaml` with the name, namespace, Cluster ID, and node-related information of the next cluster. You can choose to rename `cr.yaml` or you can overwrite the same `cr.yaml`. You can then create the cluster and after a successful creation of pods, update `cr.yaml` for the subsequent cluster. Likewise, you can configure multiple clusters.

Each InfoScale cluster must have a unique name even when clusters are created across namespaces.

Installing InfoScale on Kubernetes

All information about the worker nodes must be added to the `cr.yaml` file. All worker nodes become part of InfoScale cluster after `cr.yaml` is applied. After you download, unzip, and untar `infoscale-yamls-v9.1.2.tar.gz` , all files required for installation are available.

Note: You must download images required for installation from the InfoScale Download Center and push those to the Custom registry.

Configure a new user - infoscale-admin, associated with a Role-based Access Control (RBAC) clusterrole defined in`infoscale-admin-role.yaml`, to deploy InfoScale and its dependent components. infoscale-admin as a user when configured has clusterwide access to only those resources needed to deploy InfoScale and its dependent components such as Cert Manager in the desired namespaces.

To provide a secure and isolated environment for InfoScale deployment and associated resources, the namespace associated with these resources must be protected from access of all other users (except super user of the cluster), with appropriate RBAC implemented.

Run the following commands on the master node to create a new user -infoscale-admin and assign a clusterrole to infoscale-admin. You must be logged in as a super user.

1. `kubectl config use-context kubernetes-admin@kubernetes`

   You have switched the context to `kubernetes-admin@kubernetes`.

2. `kubectl apply -f /infoscale-yamls-v9.1.2/kubernetes/infoscale-admin-role.yaml`

   Here, `infoscale-admin-role.yaml` is downloaded as a part of `infoscale-yamls-v9.1.2.tar.gz` .

3. `kubectl create clusterrolebinding infoscale-admin-clusterrolebinding --clusterrole=infoscale-admin-role --user=infoscale-admin`

   This step binds the cluster role `infoscale-admin-role.yaml` to a new user - `infoscale-admin`.

4. `openssl genrsa -out infoscale-admin.key 2048`

   You have generated a key for `infoscale-admin` by using 2048 bits.

   `openssl req -new -key infoscale-admin.key -out infoscale-admin.csr -subj "/CN=infoscale-admin"`

   You have created a certificate signing request for `infoscale-admin`.

   `openssl x509 -req -in infoscale-admin.csr -CA /etc/kubernetes/pki/ca.crt -CAkey /etc/kubernetes/pki/ca.key -CAcreateserial -out infoscale-admin.crt -days <Number of days>`

   The certificate is signed by Kubernetes, a crt file is created, and is valid for the number of days you specified in the command.

5. `mkdir /certs && mv infoscale-admin.crt infoscale-admin.key /certs`

   `infoscale-admin.crt` and `infoscale-admin.key` moved to `/certs` folder for the `lico_rhel.yaml` to reference.

6 `cd /certs`

```
kubectl config set-credentials infoscale-admin
--client-certificate=/certs/infoscale-admin.crt
--client-key=/certs/infoscale-admin.key
```

```
kubectl config set-context infoscale-admin.context
--cluster=kubernetes --user=infoscale-admin
```

```
kubectl config use-context infoscale-admin.context
```

Credentials are set for `infoscale-admin` and its context created.

7 `kubectl config current-context infoscale-admin.context`

Ensures that the current context is `infoscale-admin.context`.

You must perform all installation-related activities by logging in as infoscale-admin. A cluster super-user can also install InfoScale.

1. Edit `/infoscale-yamls-v9.1.2/kubernetes/iso_rhel.yaml` as under -

   Replace image: <org-registry>/arctera/infoscale-storage-operator:9.1.2-rhel with image: <IP address of custom registry>/infoscale-storage-operator:9.1.2-rhel.

2. Edit `/infoscale-yamls-v9.1.2/kubernetes/lico_rhel.yaml` as under -

   Replace image: <org-registry>/arctera/infoscale-license-operator:9.1.2-rhel with image: <IP address of custom registry>/infoscale-license-operator:9.1.2-rhel.

3. Edit `/infoscale-yamls-v9.1.2/kubernetes/dro_deployment.yaml` as under -

   Replace image: <org-registry>/arctera/infoscale-dr-operator:9.1.2-rhel with image: <IP address of custom registry>/arctera/infoscale-dr-operator:9.1.2-rhel.

4. Optionally, if you want to change the default kubelet path, edit `/infoscale-yamls-v9.1.2/kubernetes/iso_rhel.yaml` as under.

```
env:

   - name: KUBELET_PATH

     value: <enter the new path>
```

   The default path is `/var/lib/kubelet`.

   Note: Do not change the kubelet path after clusters are configured.

5. Run the following command on the master node to install InfoScale.

   ```
   kubectl create -f /infoscale-yamls-v9.1.2/kubernetes/iso_rhel.yaml
   ```

6. Run the following command on the master node to verify whether the installation is successful.

   ```
   kubectl get pods -n infoscale-vtas|grep infoscale-sds-operator
   ```

   An output similar to the following indicates a successful installation. `READY 1/1` indicates that Storage cluster resources can be created.

   ```
   NAME                                  READY   STATUS  RESTARTS AGE
   infoscale-sds-operator-6dc9bc8856-lh72f 1/1     Running 0        2d18h
   ```

Configuring cluster

Complete the following steps for each cluster.

Note: Ensure that you add a Kubernetes node only to a single InfoScale cluster.

---

1. Edit clusterInfo section of the sample `/infoscale-yamls-v9.1.2/kubernetes/cr.yaml` for InfoScale specifications as under -

   ```
   ---
   apiVersion: infoscale.veritas.com/v1
   kind: InfoScaleCluster
   metadata:
     # Please change cluster name if required
     name: infoscalecluster-dev
     namespace: infoscale-vtas
   spec:
     # This denotes version of the InfoScale release
     version: 9.1.2
     # (optional) This denotes the user-provisioned ID for InfoScale
    cluster
     # The value can range from 1 to 65535
     # clusterID: <Cluster_ID>
     clusterInfo:
     # Please change worker node names according to your cluster
    configuration
     # Mention additional worker node names and corresponding node
   ```

```
 specific
  # parameters, as applicable.
  - nodeName: "<Hostname_of_worker_node>"
    # (optional) Specifies node IP address(es) to be used for
 InfoScale cluster.
    # If omitted, InfoScale chooses available IP address(es)
 for cluster config.
    # Please change worker IP address(es) according to your
 cluster configuration
    # ip:
    # - "<IP_Address_1>"
    # - "<IP_Address_2>"
    # (optional) Specifies a node-specific list of devices
 for InfoScale configuration.
    #
    # Only one of the following fields can be used at a time:
    # - `includeDevices`: List of devices to be explicitly
included in InfoScale configuration.
    # - `excludeDevice`: List of devices to be excluded from
 InfoScale configuration.
    #
    # Please update the device names according to your cluster
 setup.
    # includeDevices:
    # - "<Hardware_Path_to_device_to_be_included>"
    # excludeDevice:
    # - "<Hardware_Path_to_device_to_be_excluded>"
    # (optional) Specifies node specific list of devices to be
 used for fencing purpose
    # It is sufficient to provide fencing device information
 from one node
    # (optional) Specifies node specific list of devices to be
 used for fencing purpose
    # It is sufficient to provide fencing device information
 from one node
    # Please change device names according to your cluster
 configuration
    # fencingDevice:
    # - "<Hardware_Path_to_device_to_be_used_for_fencing>"
  - nodeName: "<Hostname_of_worker_node>"
    # (optional) Specifies node IP address(es) to be used for
 InfoScale cluster.
    # If omitted, InfoScale chooses available IP address(es)
```

```
for cluster config.
   # Please change worker IP address(es) according to your
cluster configuration
   # ip:
   # - "<IP_Address_1>"
   # - "<IP_Address_2>"
   # (optional) Specifies node specific list of devices to be
excluded from
   # InfoScale configuration.
   # Please change device names according to your cluster
configuration
   # excludeDevice:
   # - "<Hardware_Path_to_device_to_be_excluded>"
 # Please change below customImageRegistry according to your
environment
 # This is mandatory for Kubernetes and Air gapped system
deployments
 # This is optional for OCP deployments
 # customImageRegistry: "<Custom_Registry_Address>"
 # (optional) Specifies whether SCSI-3 Persistent Reservation
should be enabled
 # If omitted, SCSI3-PR reserveation will be disabled by default.
 # Valid values are true or false.
 # enableScsi3pr : <true/false>
 # (optional) Specifies whether Disk Group Level Encryption
should be enabled
 # If omitted, Disk Group Level Encryption will be disabled
by default.
 # Valid values are true or false.
 # encrypted: <true|false>
 # (optional) Specifies whether Disk Group Level Encryption
Key should be same for all volumes in the DG
 # If omitted, different key will be created for each volume
by default.
 # Valid values are true or false.
 # sameEncKey: <true|false>
 # (optional) Specifies whether to create Shared NonFSS Disk
Group.
 # With this option only the full shared disks will be included
in Disk Group
 # If omitted, FSS Disk Group will be created by default.
 # Valid values are true or false.
```

```
  # isSharedStorage: <true|false>
```

You can add up to 16 nodes.

Note: Do not enclose parameter values in angle brackets (<>). For example, Primarynode is the name of the first node; for nodeName : <Name of the first node> , enter nodeName : Primarynode. InfoScale on Kubernetes is a keyless deployment.

---

2. You can choose to rename `cr.yaml`. If you rename the file, ensure that you use that name in the next step.

Note: InfoScale recommends renaming `cr.yaml` and maintaining a custom resource for each cluster. The renamed `cr.yaml` is used to add more nodes to that InfoScale cluster.

---

3. Run the following command on the master node.

```
kubectl create -f /infoscale-yamls-v9.1.2/kubernetes/cr.yaml
```

4. Run the following command on the master node to know the name and namespace of the cluster.

```
kubectl get infoscalecluster -A
```

Use the namespace from the output similar to the following -

```
NAMESPACE   NAME                VERSION   CLUSTERID   STATE   AGE
.
.
<Namespace> <Name of the
             InfoScale cluster> 9.1.2  <Cluster ID> Running 25h
.
.
```

5. Run the following command on the master node to verify whether the pods are created successfully.

```
kubectl get pods -n infoscale-vtas
```

An output similar to the following indicates a successful creation of nodes

```
NAME                                         READY   STATUS    RESTART
infoscale-csi-controller-7b4b7bd899-4c786    5/5     Running   0
infoscale-csi-node-5rlr7                     2/2     Running   0
infoscale-csi-node-fl57f                     2/2     Running   0
infoscale-csi-node-srnh5                     2/2     Running   0
infoscale-fencing-controller-654f549565-d56jk 1/1    Running   0
infoscale-fencing-enabler-4xxx7              1/1     Running   0
infoscale-fencing-enabler-hr26f              1/1     Running   0
infoscale-fencing-enabler-qdx4v              1/1     Running   0
infoscale-licensing-operator-cfff86577-kcpnl 1/1     Running   1 (43m
infoscale-sds-22222-a924e23b0ed1ca1d-6t4fs   1/1     Running   0
infoscale-sds-22222-a924e23b0ed1ca1d-qhqzn   1/1     Running   0
infoscale-sds-22222-a924e23b0ed1ca1d-w48vz   1/1     Running   0
infoscale-sds-operator-7fd45465cc-cssmm      1/1     Running   0
infoscale-toolset-22222-b7c8cbf78-b9xnv      1/1     Running   0
infoscale-toolset-22222-b7c8cbf78-fb8vz      1/1     Running   0
infoscale-toolset-22222-b7c8cbf78-x5cjj      1/1     Running   0
# kubectl get infoscalecluster -A
NAMESPACE        NAME                    VERSION   CLUSTERID   STATE     D
infoscale-vtas   infoscalecluster-dev    9.1.2     22222       Running   v
```

InfoScale SDS pods are created in the namespace you specify in `cr.yaml`. Fencing and CSI pods are created in the operator namespace.

After a successful InfoScale deployment, a disk group is automatically created. You can now create Persistent Volumes/ Persistent Volume Claims (PV / PVC) by using the corresponding Storage class.

See "Adding nodes to an existing cluster" on page 261.

See "Removing nodes from an existing cluster" on page 267.

Undeploying and uninstalling InfoScale

You can run the following command to undeploy and uninstall InfoScale on your Kubernetes cluster.

Note: If you want to retain the disk group and re-use the Volumes you created, ensure that you note the `ClusterID` before undeploying and uninstalling InfoScale. You must use this `ClusterID` while re-installing InfoScale. Else, clean the disks before uninstalling.

```
kubectl delete infoscalecluster -n <Namespace> <Name of the cluster>
```

Note: Run the command for each cluster.

The commands to clean up InfoScale components are as follows:

Note: Run these commands only after all InfoScale pods are terminated.

```
kubectl delete -f /infoscale-yamls-v9.1.2/kubernetes/iso_rhel.yaml

kubectl delete -f
/infoscale-yamls-v9.1.2/kubernetes/vlic_v1_license.yaml

kubectl delete -f /infoscale-yamls-v9.1.2/kubernetes/lico_rhel.yaml
```

Note: After uninstallation, ensure that stale InfoScale kernel modules (`vxio`/`vxdmp`/`veki`/`vxspec`/`vxfs`/`odm`/`glm`/`gms`) do not remain loaded on any of the worker nodes. Rebooting a worker node deletes all such modules. When fencing is configured, certain keys are configured. Those must also be deleted after uninstallation. Run `./clearkeys.sh <Path to the first disk>, <Path to the second disk>,...` to remove stale keys that might have remained.

Upgrading InfoScale

This chapter includes the following topics:

Overview

This section outlines the workflows for IKE upgrades and end-to-end upgrades. Users can upgrade InfoScale cluster in two steps:

1. Upgrade the InfoScale Operator.
2. Upgrade the InfoScale cluster version (change the CR version).

InfoScale upgrade workflows can change with additional preparatory steps based on the source version and known issues. These workflows allow IKE to ensure upgrades without downtime for applications and virtual machines in OpenShift and OpenShift Virtualization (OCP-V) platforms.

Note: Complete steps 1 and 2 before upgrading OpenShift, unless a combined upgrade is supported.

Update the CR version to match the target InfoScale release to begin the cluster upgrade.

Prerequisites

- Maintain a stable and responsive platform by avoiding API server saturation, excessive thrashing, or high node resource pressure.
- Ensure enough resources are available for VM migration if you are using OpenShift Virtualization.
- Clear all errors or pending remediations flagged by the Pre-flight CLI found in infoscale-tools-v9.1.2.tar.

  Note: Verify the InfoScale support matrix before you choose your OpenShift cluster upgrade version.

```
Before running the script, ensure that you have Bash
installed on your system.

- Bash: This script is intended to be run in a Bash shell.
If you're using a
different shell, consider switching to Bash.
Also ,make sure jq is installed since script uses it.

./preflight-cli.sh --target-ike <infoscale_version>
--target-ocp <ocp_version>

e.g.
./preflight-cli.sh --target-ike 9.1.2 --target-ocp 4.19.x


========== PRE-FLIGHT SUMMARY ==========
04-workload-sanity.sh : All checks passed
03-sourceclust.sh : All checks passed
01-platform.sh : All checks passed
02-ike-versions.sh : All checks passed
========================================
```

```
[INFO]  All output saved to: /opt/Preflight/preflight/
logs/preflight-20260317-095735/
preflight.log
```

- Complete all pending rollouts and ensure all configurations are in sync.
- Resolve any existing split-brain issues in the InfoScale cluster before starting the upgrade.

Note: This is applicable only when data disk fencing is configured.

```
Before OCP upgrade is executed on the OCP-V VIKE cluster
please execute following following commands  inside SDS pods

#oc exec -it infoscale-sds-21432-xxxx-xxxx  -n infoscale-vtas -- bash

#vxfenadm -s /dev/vx/rdmp/<dmpnodename>

No of keys should match the number of nodes X no of paths for each disk.

e.g  if no of nodes is 5 and each disk has 2 path  then you should see 10


[root@ocptest-01 /]# vxfenadm -s /dev/vx/rdmp/emc0_346f

Reading SCSI Registration Keys...

Device Name: /dev/vx/rdmp/emc0_346f
Total Number Of Keys: 10
key[0]:
        [Numeric Format]:   65,80,71,82,48,48,49,57
        [Character Format]: APGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 0   Node Name:
ocptest-01.test.int
key[1]:
        [Numeric Format]:   65,80,71,82,48,48,49,57
        [Character Format]: APGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 0   Node Name:
ocptest-01.test.int
key[2]:
        [Numeric Format]:   67,80,71,82,48,48,49,57
        [Character Format]: CPGR0019
```

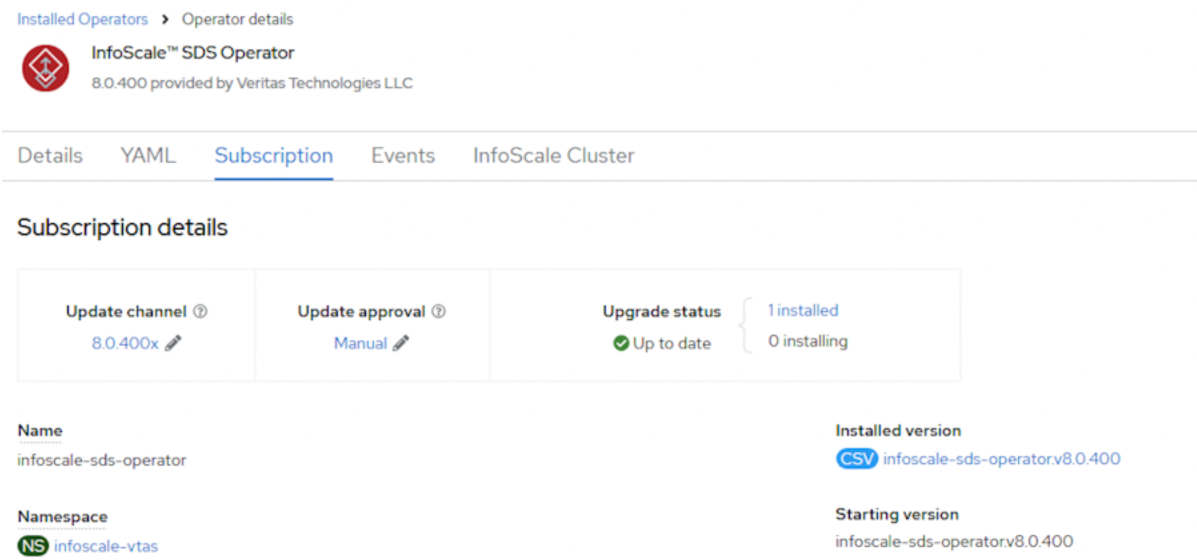
```
        [Node Format]: Cluster ID: unknown  Node ID: 2   Node Name:
ocptest-02.test.int
key[3]:
        [Numeric Format]:   67,80,71,82,48,48,49,57
        [Character Format]: CPGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 2   Node Name:
ocptest-02.test.int
key[4]:
        [Numeric Format]:   68,80,71,82,48,48,49,57
        [Character Format]: DPGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 3   Node Name:
ocptest-03.test.int
key[5]:
        [Numeric Format]:   68,80,71,82,48,48,49,57
        [Character Format]: DPGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 3   Node Name:
ocptest-03.test.int
key[6]:
        [Numeric Format]:   69,80,71,82,48,48,49,57
        [Character Format]: EPGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 4   Node Name:
ocptest-04.test.int
key[7]:
        [Numeric Format]:   69,80,71,82,48,48,49,57
        [Character Format]: EPGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 4   Node Name:
ocptest-04.test.int
key[8]:
        [Numeric Format]:   66,80,71,82,48,48,49,57
        [Character Format]: BPGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 1   Node Name:
ocptest-05.test.int
key[9]:
        [Numeric Format]:   66,80,71,82,48,48,49,57
        [Character Format]: BPGR0019
        [Node Format]: Cluster ID: unknown  Node ID: 1   Node Name:
ocptest-05.test.int
```

- Disable hard eviction settings in the kubeletconfig if system memory reservations are exceeded.

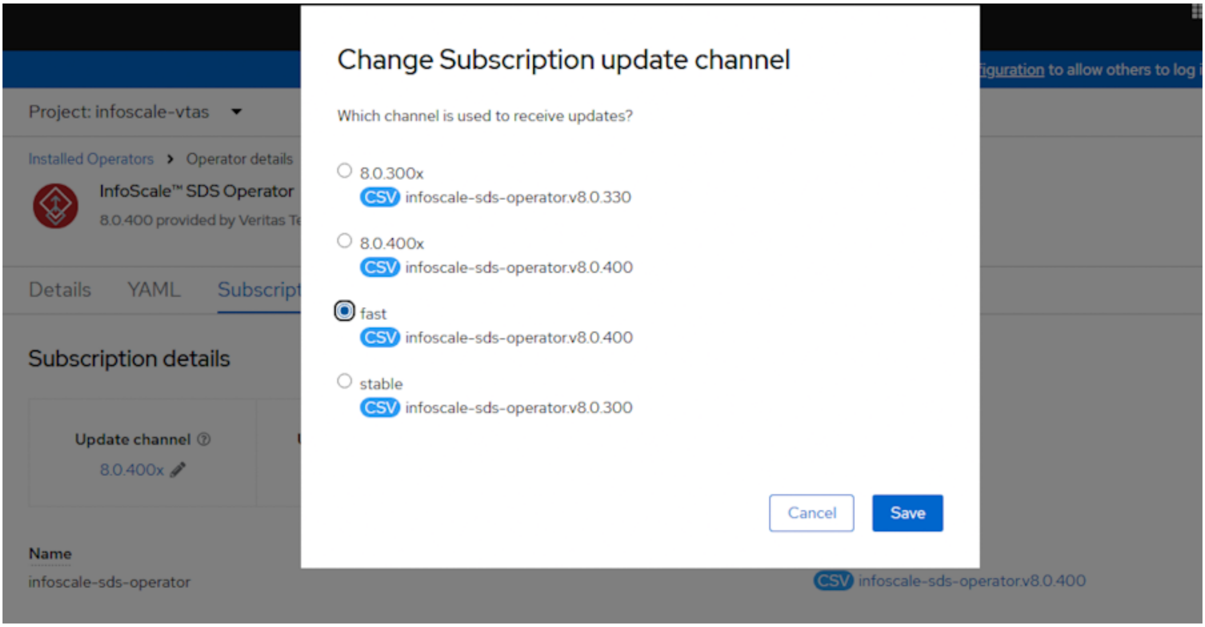
InfoScale operator upgrade

Complete the following steps to upgrade InfoScale operator.

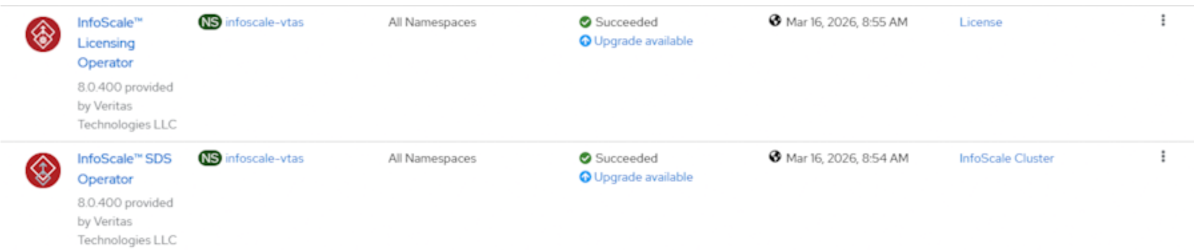
1. Log in to the OpenShift console and navigate to the correct page:
   - For versions below 4.20: Go to Operators > Installed Operators.
   - For versions 4.20 and above: Go to Ecosystem > Software Catalog.
2. Verify that all InfoScale operators are listed on this page.
3. Click the InfoScale SDS Operator and select the Subscription tab to open the next screen.



4. Click Update channel. Select fast and click Save.

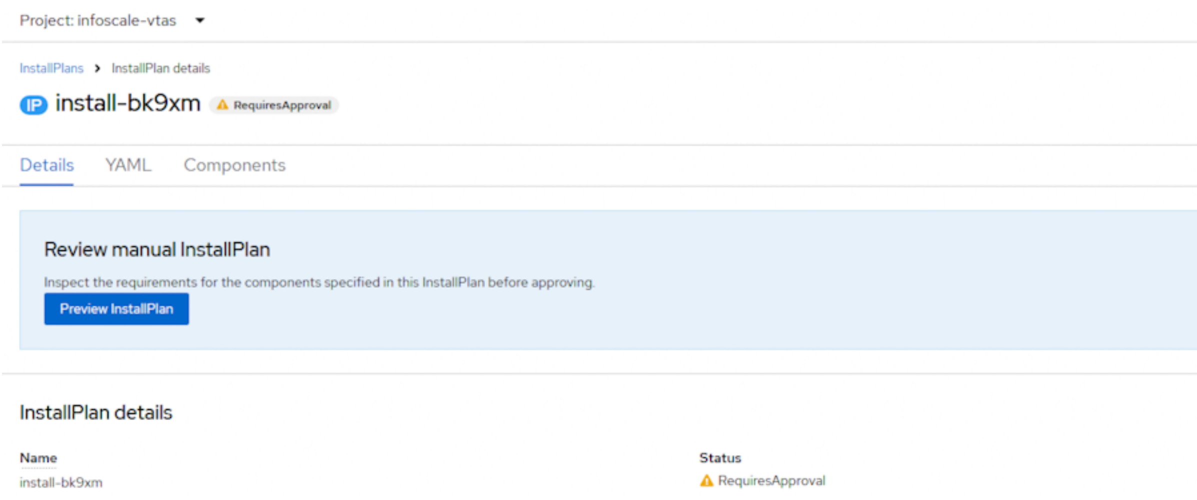


5. For the InfoScale operators, Upgrade Available is reported in Status as the following

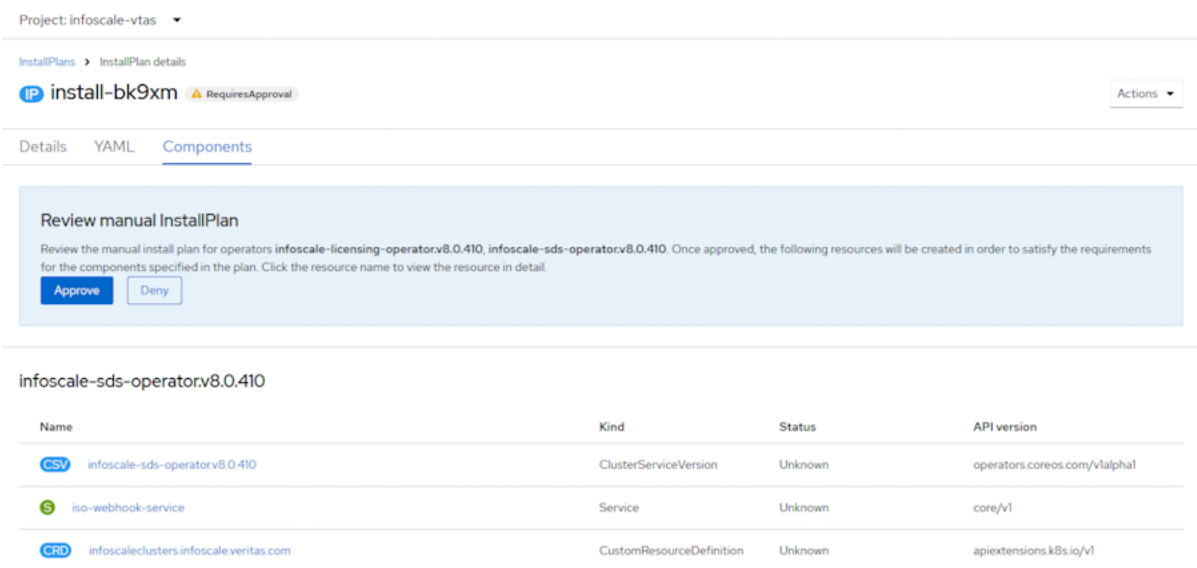


6. To upgrade InfoScale, click InfoScale SDS Operator or InfoScale Licensing Operator.

7. In Subscription, updates that need approval are displayed next to Upgrade Status.

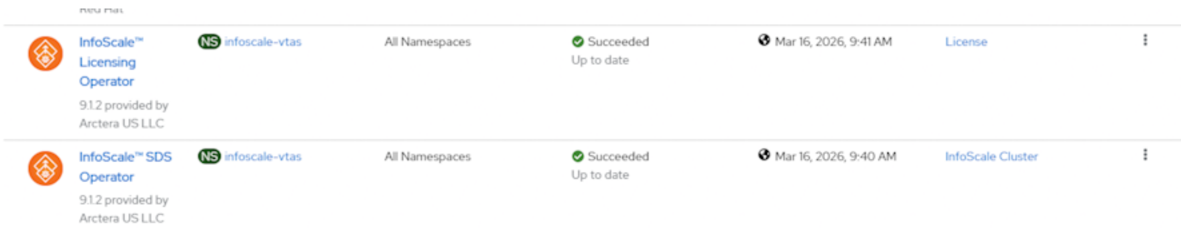


8. Review the details list. Click Approve to proceed with the upgrade.



Note: The operator upgrade may require multiple steps (hops) depending on your current version. For example, an upgrade from version 8.0.400 must follow this path: 8.0.410 > 9.1.0 > 9.1.2. Upgrading from 9.1.0 to 9.1.2 only requires one hop.

9. Wait till it is upgraded and listed in Operators > Installed Operators.



InfoScale Cluster version upgrade (Software upgrade)

To upgrade the InfoScale cluster version:

1. Verify that the kubelet configuration is applied and the rollout is complete before starting the software upgrade. Refer to See “Enabling kubelet inhibitor with systemd” on page 29.
   - If the recommended 9.x kubelet configuration was missed during deployment, apply it before upgrading the software.
2. Apply the additional required patch if you are upgrading InfoScale from 8.x to 9.1.2.
3. Change the InfoScale cluster version to the latest version in the Custom Resource (CR).
4. Wait for step 3 to complete and then trigger the platform or OpenShift upgrade, if applicable to your environment.

Upgrading InfoScale

Note: To configure SDS pod life cycle hooks, apply the additional kubelet configuration required for this major InfoScale version change. To minimize restarts caused by this configuration, pause the schedulable machine config pools before you begin.

Note: Make sure to un-pause the machine config pools as instructed. If you do not follow these steps correctly, it will cause a forced rollout of your nodes.

Note: Follow these same steps for version 9.x if you missed the pre-deployment requirement or if the preflight CLI flagged this issue.

```
oc get kubeletconfigs.machineconfiguration.openshift.io
is empty OR without following params
# oc get kubeletconfigs.machineconfiguration.openshift.io -oyaml |
grep -i grace
      shutdownGracePeriod: 15m
      shutdownGracePeriodCriticalPods: 5m
And if its present, check whether it's rolled out
replace the correct node below; it should show kubelet inhibitor.
if it's not present proceed with given configuration.

# ssh core@ocp-w-02.lab.ocp.lan sudo systemd-inhibit
WHO            UID USER PID  COMM           WHAT     WHY
NetworkManager 0   root 1486 NetworkManager sleep    NetworkManager
 needs to turn off networks  delay
kubelet        0   root 3112 kubelet        shutdown Kubelet needs
time to handle node shutdown delay
```

If these kubelet inhibitors are already applied, you can skip Steps A and B.

A. Follow if both workers and masters are schedulable and ANY of master nodes is part of InfoScale

```
Label the MCP

oc label mcp worker machineconfiguration.openshift.io/domain=sched
oc label mcp master machineconfiguration.openshift.io/domain=sched

# pause both MCPs
oc patch mcp worker --type=merge -p '{"spec":{"paused":true}}'
machineconfigpool.machineconfiguration.openshift.io/worker patched

oc patch mcp master --type=merge -p '{"spec":{"paused":true}}'
machineconfigpool.machineconfiguration.openshift.io/master patched

# Apply common systemd config
oc apply -f https://raw.githubusercontent.com/Arctera/
infoscale-kubernetes-enterprise/main/config/sysd/
99-common-sysd.yaml
# Apply common kubelet config
```

```
oc apply -f https://raw.githubusercontent.com/Arctera/
infoscale-kubernetes-enterprise/main/config/kubelet/
common-kubelet-config.yaml
```

Note: The Machine Config Operator will only roll out this configuration when you unpause it or set pause=false. Do not perform this step yet; follow the next instructions first.

B. Kubeletconfig – Follow if only workers are schedulable

```
# kubelet config should have the same label present in this case.
oc label mcp worker machineconfiguration.openshift.io/role=worker
machineconfigpool.machineconfiguration.openshift.io/worker labeled

oc patch mcp worker --type=merge -p '{"spec":{"paused":true}}'
machineconfigpool.machineconfiguration.openshift.io/worker patched

oc get mcp worker -oyaml  | grep pause
  paused: true

# Apply systemd config

oc apply -f https://raw.githubusercontent.com/Arctera/
infoscale-kubernetes-enterprise/main/config/sysd/99-worker-sysd.yaml
machineconfig.machineconfiguration.openshift.io/
99-worker-sysd-conf-override created

# Apply kubelet config for worker
oc apply -f https://raw.githubusercontent.com/Arctera/
infoscale-kubernetes-enterprise/main/config/kubelet/
kubelet-config.yaml
kubeletconfig.machineconfiguration.openshift.io/
custom-kubelet-config created

# Should show success as -
oc describe kubeletconfigs.machineconfiguration.openshift.io
custom-kubelet-config
Name:         custom-kubelet-config
Namespace:
Labels:       <none>
Annotations:  machineconfiguration.openshift.io/mc-name-suffix:
```

```
API Version:  machineconfiguration.openshift.io/v1
Kind:         KubeletConfig
Metadata:
  Creation Timestamp:  2026-01-16T13:15:00Z
  Finalizers:
    99-worker-generated-kubelet
  Generation:        1
  Resource Version:  19382539
  UID:               940d4930-c19d-4e8e-98f1-99d0fb65e2cd
Spec:
  Kubelet Config:
    Shutdown Grace Period:                15m
    Shutdown Grace Period Critical Pods:  5m
  Machine Config Pool Selector:
    Match Labels:
      machineconfiguration.openshift.io/role:  worker
<<< can change as per label
Status:
  Conditions:
    Last Transition Time:  2026-01-16T13:15:00Z
    Message:               Success
    Status:                True
    Type:                  Success
Events:                    <none>
```

Note: Note that the machine config pools are still paused at this stage. If your source version is 8.0.4x, apply the required patch now.

This step won't be needed if source InfoScale cluster version is 9.x.

```
# Contact InfoScale support team to download patch
infoscale-patch-8.0.4x.tar.gz
tar xvzf infoscale-patch-8.0.4x.tar.gz
cd infoscale-patch-8.0.4x

# Apply the patch
./apply-cluster-patch.sh infoscalecluster-dev
# Apply
./apply-cluster-patch.sh infoscalecluster-dev
[INFO] Using CLI: oc
[INFO] Locating InfoScaleCluster: infoscalecluster-dev
[INFO] Cluster namespace: infoscale-vtas
[INFO] Discovering pods for cluster infoscalecluster-dev
```

```
[INFO] Found pods:
  - infoscale-sds-1230-1eb7816cdb4bd7f3-9dlmv
  - infoscale-sds-1230-1eb7816cdb4bd7f3-fjpwt
  - infoscale-sds-1230-1eb7816cdb4bd7f3-vt2nj
  - infoscale-sds-1230-1eb7816cdb4bd7f3-w9ls8
[INFO] Collecting files from patch directory: /opt/infoscale-patch-8.0.4x
[INFO] Processing pod: infoscale-sds-1230-1eb7816cdb4bd7f3-9dlmv
[INFO]   -> /usr/sbin/vxassist
[INFO]   -> /etc/systemd/system/pod-prestop.service
Created symlink /etc/systemd/system/multi-user.target.wants/
pod-prestop.service → /etc/systemd/system/pod-prestop.service.
[INFO]   -> /sbin/vss-util
[INFO]   -> /sbin/vss-stop
[INFO] Processing pod: infoscale-sds-1230-1eb7816cdb4bd7f3-fjpwt
[INFO]   -> /usr/sbin/vxassist
[INFO]   -> /etc/systemd/system/pod-prestop.service
Created symlink /etc/systemd/system/multi-user.target.wants/
pod-prestop.service → /etc/systemd/system/pod-prestop.service.
[INFO]   -> /sbin/vss-util
[INFO]   -> /sbin/vss-stop
[INFO] Processing pod: infoscale-sds-1230-1eb7816cdb4bd7f3-vt2nj
[INFO]   -> /usr/sbin/vxassist
[INFO]   -> /etc/systemd/system/pod-prestop.service
Created symlink /etc/systemd/system/multi-user.target.wants/
pod-prestop.service → /etc/systemd/system/pod-prestop.service.
[INFO]   -> /sbin/vss-util
[INFO]   -> /sbin/vss-stop
[INFO] Processing pod: infoscale-sds-1230-1eb7816cdb4bd7f3-w9ls8
[INFO]   -> /usr/sbin/vxassist
[INFO]   -> /etc/systemd/system/pod-prestop.service
Created symlink /etc/systemd/system/multi-user.target.wants/
pod-prestop.service → /etc/systemd/system/pod-prestop.service.
[INFO]   -> /sbin/vss-util
[INFO]   -> /sbin/vss-stop
[INFO] Patch applied successfully to cluster: infoscalecluster-dev
```

Patch InfoScale cluster version to 9.1.2

With this step, you are triggering the software upgrade for the InfoScale cluster.

```
oc patch infoscalecluster <cluster-name>  -n <namespace>
--type=merge  -p '{"spec":{"version":"9.1.2"}}'
infoscalecluster.infoscale.veritas.com/<namespace> patched
```

```
Wait for cluster to be in running state  and rollout to be complete
# oc get po -n infoscale-vtas
NAME                                          READY  STATUS       RESTARTS
infoscale-csi-controller-569fb9fc4-k6ddn      5/5    Running      1 (51s ago
infoscale-csi-node-c85tf                      2/2    Running      0
infoscale-csi-node-cpwrt                      2/2    Terminating  0
infoscale-csi-node-gsx5d                      2/2    Running      0
infoscale-csi-node-xfgjn                      2/2    Running      0
infoscale-fencing-controller-5b86c59dfd-f7ptw 1/1    Running      0
infoscale-fencing-enabler-6mx7d               1/1    Running      0
infoscale-fencing-enabler-j8zgw               1/1    Running      0
infoscale-fencing-enabler-qrxtw               1/1    Running      0
infoscale-fencing-enabler-tjcbk               1/1    Running      0
infoscale-licensing-operator-7bb6cb7d48-5hz5b 1/1    Running      0
infoscale-sds-1230-1eb7816cdb4bd7f3-drhlq     1/1    Running      0
infoscale-sds-1230-1eb7816cdb4bd7f3-fzdzl     1/1    Running      0
infoscale-sds-1230-1eb7816cdb4bd7f3-gf62d     1/1    Running      0
infoscale-sds-1230-1eb7816cdb4bd7f3-hq6b9     1/1    Running      0
infoscale-sds-operator-66c66c6ccf-wfzv4       1/1    Running      0
infoscale-toolset-1230-6bc76b6b96-64flx       1/1    Running      0
infoscale-toolset-1230-6bc76b6b96-87jrp       1/1    Running      0
infoscale-toolset-1230-6bc76b6b96-mggxs       1/1    Running      0
infoscale-toolset-1230-6d9f474779-nwb26       0/1    Running      0
---
# oc get infoscaleclusters -A
NAMESPACE        NAME                   VERSION   CLUSTERID   STATE
infoscale-vtas   infoscalecluster-dev   8.0.400   1230        Upgrading

DISKGROUPS          STATUS    AGE
vrts_kube_dg-1230   Healthy   3d5h
```

Wait for completion of software upgrade

At this stage, you will see the changes roll out first for CSI, fencing, and the Toolset, followed by the SDS. The cluster status will switch between "Degraded" and "Running" because nodes leave the cluster one at a time to upgrade and then rejoin. The time this takes will depend on the workload running on your system.

```
infoscale-vtas   infoscalecluster-dev   8.0.400   1230
Upgrading   vrts_kube_dg-1230   Degraded   38m

infoscale-vtas   infoscalecluster-dev   8.0.400   1230
Upgrading   vrts_kube_dg-1230   Healthy    39m
```

```
infoscale-vtas   infoscalecluster-dev   9.1.2     1230
Running     vrts_kube_dg-1230   Healthy    39m
```

Enable paused pools

Note: Only follow this step if the worker/master pools are paused, as it applies to the kubelet config mentioned in Step A or B. If you've done so, unpause the pools one at a time, sequentially.

Unpause the master pools first if they are schedulable. Wait for the rollout to finish completely before unpausing the worker pools. Note that enabling both pools at the same time does not guarantee that your applications will remain available.

```
oc patch mcp worker --type=merge -p '{"spec":{"paused":false}}'
machineconfigpool.machineconfiguration.openshift.io/worker patched
---
Every 2.0s: oc get no
NAME                   STATUS  ROLES                AGE   VERSION
ocp348-m1.test.int     Ready   control-plane,master   41d   v1.30.7
ocp348-m2.test.int     Ready   control-plane,master   41d   v1.30.7
ocp348-m3.test.int     Ready   control-plane,master   41d   v1.30.7
ocp348-w01.test.int    Ready   worker                 41d   v1.30.7
ocp348-w02.test.int    Ready   worker                 41d   v1.30.7
ocp348-w03.test.int    Ready   worker                 41d   v1.30.7
ocp348-w04.test.int    ReadySchedulingDisabled
                               worker                 41d   v1.30.7
```

Wait for rollout and ready state

Wait for the rollout to finish and for the InfoScale cluster status to show as "Ready."

If a node takes too long to recover and shows a status of NotReady or SchedulingDisabled, try resetting the node.

```
# oc get infoscaleclusters -Aw
NAMESPACE        NAME                   VERSION   CLUSTERID
infoscale-vtas   infoscalecluster-dev   9.1.2     1230
infoscale-vtas   infoscalecluster-dev   9.1.2     1230

STATE     DISKGROUPS          STATUS     AGE
Running   vrts_kube_dg-1230   Degraded   117m
Running   vrts_kube_dg-1230   Healthy    121m
```

Platform or OpenShift upgrade

Refer to See “Supported platforms” on page 16. for more information.

IKE is not supported on below OpenShift versions.

Table 6-1 Unsupported platforms

OCP version	Restriction
4.20.4	All channels
4.19.19	All channels
4.18.29	All channels
4.17.44	All channels
4.16.53	All channels

Note: During an OpenShift upgrade, both the master and worker pools usually roll out at the same time. This can cause issues if your master nodes are schedulable. For this reason, IKE recommends that you do not have schedulable master nodes. Instead, you should upgrade the master pool first and wait for it to finish before you start the worker pool upgrade.

You can optionally re-run the pre-flight script to check the health of your InfoScale cluster. Once the cluster is healthy, you can proceed with the OpenShift platform upgrade.

During the upgrade, when the Machine Config Operator updates, the InfoScale cluster will show an OS-Upgrade status, as seen below:

```
#oc get infoscalecluster -A
NAMESPACE        NAME             VERSION    CLUSTERID    STATE
infoscale-vtas   sanity-ocp416    9.1.2      21432        OS-Upgrade

DISKGROUPS          STATUS      AGE
vrts_kube_dg-21432  Degraded    2d1h
```

After the OpenShift upgrade finishes, make sure the InfoScale cluster shows a "Running" status before you start any other tasks.

```
# oc get infoscalecluster -A
```

```
NAMESPACE        NAME            VERSION   CLUSTERID   STATE
infoscale-vtas   sanity-ocp416   9.1.2     21432       Running

DISKGROUPS           STATUS    AGE
vrts_kube_dg-21432   Healthy   2d1h
```

Troubleshooting

Paused Upgrade due to affined and unmanaged workloads

If a workload is not managed by a standard Kubernetes controller, the InfoScale operator will pause the upgrade and record the following events:

```
Warning  UpdatePaused  23m (x89 over 41m)   InfoScaleCluster
ErrorCode=10050
ErrorMsg=Resource is not managed by controller : pod test/redis
not managed statefulset,
replicaset, daemonset on <hostname>

Warning  UpdatePaused  14m (x4 over 15m)    InfoScaleCluster  to
resume upgrade
 with forced migration of such workloads, annotate InfoScaleCluster
with infoscale.veritas.com/
forceMigrate=true
```

If the InfoScale operator pauses due to unmanaged workloads, the administrator must handle them. This includes single pods or pods managed by other operators that do not automatically move to a different node.

- Back up and delete these resources if they can be safely removed.
- Annotate the InfoScaleCluster to force the upgrade if it is safe for these resources to be down. You can find the specific annotation in the event logs mentioned earlier.

For example:

```
oc annotate infoscaleclusters infoscalecluster-dev infoscale.veritas.com/
forceMigrate=true
infoscalecluster.infoscale.veritas.com/infoscalecluster-dev annotated
```

After the upgrade finishes using the force option, the administrator can remove the annotation from the InfoScaleCluster resource.

NFD crash in OCP upgrades from 4.19 to 4.20

The OCP upgrade was blocked because the NFD garbage collector pods were crashing. This was caused by a readiness failure, as shown in the logs below.

```
NAME                                      READY   STATUS    RESTARTS
nfd-controller-manager-658cccddf9-w9l54   1/1     Running   0
nfd-gc-6b7549bfc4-llv5w                   0/1     Running   1 (35s ago)
continesouly restarting

nfd-gc-fbbd48975-g2bw5                    1/1     Running   0
nfd-master-5bb757c8d4-hmdm6               0/1     Running   777 (2m33s ago)
nfd-master-7c585f4bf-tswdp                1/1     Running   0
nfd-worker-629wr                          1/1     Running   2 (2d16h ago)
nfd-worker-86hd9                          1/1     Running   2 (2d20h ago)
. . .
58s         Normal    Pulled            pod/nfd-gc-6b7549bfc4-llv5w
Successfully pulled image "registry.redhat.io/openshift4/ose-node-feature-dis
rhel9@sha256:
5459499aedd2ebd0245d79d6dfed353c5a24c12d54bc97cbc2cecfe2489b792e" in 3.513s
(3.513s including waiting). Image size: 617600938 bytes.

32s         Warning   Unhealthy         pod/nfd-gc-6b7549bfc4-llv5w
Liveness probe failed: Get "http://10.128.3.19:8080/healthz":
dial tcp 10.128.3.19:8080: connect: connection refused

27s         Warning   Unhealthy         pod/nfd-gc-6b7549bfc4-llv5w
Readiness probe failed: Get "http://10.128.3.19:8080/healthz":
dial tcp 10.128.3.19:8080: connect: connection refused

27s         Warning   ProbeError        pod/nfd-gc-6b7549bfc4-llv5w
Readiness probe error: Get "http://10.128.3.19:8080/healthz":
dial tcp <IP address>:8080: connect: connection refused...

22s         Warning   ProbeError        pod/nfd-gc-6b7549bfc4-llv5w
Liveness probe error: Get "http://10.128.3.19:8080/healthz":
dial tcp <IP address>:8080: connect: connection refused...

22s         Normal    Killing           pod/nfd-gc-6b7549bfc4-llv5w
Container nfd-gc failed liveness probe, will be restarted

22s         Normal    Pulling           pod/nfd-gc-6b7549bfc4-llv5w
Pulling image "registry.redhat.io/openshift4/ose-node-feature-discovery-
```

```
rhel9@sha256:
5459499aedd2ebd0245d79d6dfed353c5a24c12d54bc97cbc2cecfe2489b792e"

18s         Normal    Pulled            pod/nfd-gc-6b7549bfc4-llv5w
Successfully pulled image "registry.redhat.io/openshift4/ose-node-feature-dis
rhel9@sha256:
5459499aedd2ebd0245d79d6dfed353c5a24c12d54bc97cbc2cecfe2489b792e" in 3.457s
(3.457s including waiting). Image size: 617600938 bytes.

18s         Normal    Started           pod/nfd-gc-6b7549bfc4-llv5w
Started container nfd-gc

18s         Normal    Created           pod/nfd-gc-6b7549bfc4-llv5w
Created container: nfd-gc
```

The upgrade continued after the corresponding NFD garbage collector pod was deleted. It appears the NFD is interacting with the OCP cluster operator and blocking the upgrade. Below are the specific versions where this issue occurred:

```
  {
    "completionTime": "2026-02-06T13:02:05Z",
    "image": "quay.io/openshift-release-dev/ocp-release
@sha256:91606a5f04331ed3293f71034d4f480e38645560534805fe5a821e6b64a3f203",
    "startedTime": "2026-02-06T11:38:41Z",
    "state": "Completed",
    "verified": true,
    "version": "4.20.8"
  },
  {
    "completionTime": "2026-02-06T06:25:59Z",
    "image": "quay.io/openshift-release-dev/ocp-release
@sha256:7c2001c24aa550aa228cd2d0fc0b5d9ac6656cd4267cd7c156ec758d0687758e",
    "startedTime": "2026-02-06T05:03:33Z",
    "state": "Completed",
    "verified": true,
    "version": "4.19.21"
  },
```

Upgrades stalled: Operators not progressing after install plan approval

Since the InfoScale and License operators share the same operator group, they can sometimes cause inconsistencies. Typically, both operators create two separate Install Plans, but only one of them will list both components as "owners." When you approve the Install Plan, choose the one that shows both operator subscriptions in the preview section.

Transient node drain failures in OCP-V upgrades

You may see the following harmless error messages or alerts while upgrading the OCP-V platform.

```
error when evicting pods/"virt-launcher-<VMNAME>" -n "<NAMESPACE>"
(will retry after 5s):
```

Related details can be found in this Red Hat knowledge base.

Related articles state that these messages can be safely ignored. These pod disruption budgets allow more time for Virtual Machine resources to migrate during the upgrade.

Resource busy or Node busy error during InfoScale upgrade

An upgrade from 8.x to 9.x can get stuck with a "Resource Busy" error if there are stale or incorrect snapshot records and tags inside the InfoScale SDS pods. When this happens, the SDS operator will repeatedly report a "node busy" status or show that a volume sync is still in progress.

If pre-flight CLI reports such snapshots delete those using below stale snapshot cleanup script, this script is part of infoscale-tools-v9.1.2.tar

```
Usage:

./snapshot_cleanup.sh -n <namespace> -p <pod_name> [-g <diskgroup>] [-d]

Options:

-n <namespace>: The OpenShift namespace where the pod is located (required).

-p <pod_name>: The name of the InfoScale SDS pod (required).
```


```
Usage:

./snapshot_cleanup.sh -n <namespace> -p <pod_name> [-g <diskgroup>] [-d]

Options:

-n <namespace>: The OpenShift namespace where the pod is located (required).

-p <pod_name>: The name of the InfoScale SDS pod (required).
```


```
-g <diskgroup>: Specific disk group to process (optional). If not specified,
```

```
 all disk groups will be processed.

-d: Dry-run mode. Displays the commands that would be executed without

 performing any actual deletions (optional).

Example:

./snapshot_cleanup.sh -n infoscale-vtas -p infoscale-sds-21432-7820e9290fa0fc

-b9phd -g vrts_kube_dg-1121
-g <diskgroup>: Specific disk group to process (optional). If not specified,

 all disk groups will be processed.

-d: Dry-run mode. Displays the commands that would be executed without

 performing any actual deletions (optional).

Example:

./snapshot_cleanup.sh -n infoscale-vtas -p infoscale-sds-21432-

7820e9290fa0fc26-b9phd -g vrts_kube_dg-1121
```

Failed VM migrations due to timeouts - InfoScale upgrade is in a paused state

You may see these issues during an InfoScale cluster upgrade if:

- The cluster is overloaded, or there is not enough bandwidth set for VMs to migrate.
- A migration policy is overriding the default settings and limiting how quickly VMs can move to other nodes.
- There are too many parallel writes, causing "dirty pages" that cannot be cleared fast enough.
- Migrations are simply taking too long to finish.
- In any of these cases, the InfoScale operator will wait for you to take action. You must fix the VirtualMachineMigration resource that is in an error state. Check if it has timed out, as shown in the example below.

```
# oc describe vmim  infoscale-fio-vm-1-114880f98092cdab -n prod
Name:          infoscale-fio-vm-1-114880f98092cdab
Namespace:     prod
Labels:        kubevirt.io/vmi-name=fio-vm-1
Annotations:   kubevirt.io/latest-observed-api-version: v1
```

```
              kubevirt.io/storage-observed-api-version: v1
API Version:  kubevirt.io/v1
Kind:         VirtualMachineInstanceMigration
Metadata:
  Creation Timestamp:  2026-03-12T13:17:59Z
  Generation:          1
  Resource Version:    4628614
  UID:                 6b39e1cc-2e8f-4d44-a644-e559319185d4
Spec:
  Vmi Name:  fio-vm-1
Status:
  Migration State:
    Abort Status:    Succeeded
    Completed:       true
    End Timestamp:   2026-03-12T13:18:15Z
    Failed:          true
    Failure Reason:  Live migration is not completed after
#Num seconds and has been aborted <<<
    Migration Configuration:
      Allow Auto Converge:                    true
      Allow Post Copy:                        false
    . . .
# oc describe infoscaleclusters -n infoscale-vtas kubeburner  | grep Pause
  Warning  UpdatePaused  164m (x7 over 168m)  InfoScaleCluster
migration instance infoscale-fio-vm-8-c74edd9fa2a14506 is failed in prod,
upgrade is waiting for its completion; user intervention required

  Warning  UpdatePaused  164m (x7 over 168m)  InfoScaleCluster
migration instance infoscale-fio-vm-5-24c5edc43025ba88 is failed in prod,
upgrade is waiting for its completion; user intervention required
  . . .
```

To fix these timeouts, you can simply delete the specific VirtualMachineMigration resource. The operator will automatically try the migration again and continue with the upgrade.

To prevent this from happening again, we suggest making the following configuration changes:

- Set allowAutoConverge: true in the KubeVirt resource if it is currently set to false.
- If you see memory violation alerts in the web console showing that system memory usage is over the limit, you can increase the system-slice memory on your nodes. Note that this change will require a node rollout to take effect. Refer to this Red Hat documentation.

Although the previous page recommends a small amount of memory, a better general rule is to set the system reservation using this formula:

```
systemReserved.memory = min( max(2G, 5% of totalRAM of node), 8Gi )
```

The cluster administrator should decide on this final amount based on workload surges, network bandwidth, and node management needs.

Limitations

Consider the following limitations for the current release

- You cannot upgrade a single-node InfoScaleCluster. This is because there is no other node available to host your workloads during the process.
- Upgrading clusters deployed with Flexible Storage Sharing (FSS) is not supported in this version.

Configuring KMS-based encryption on an OpenShift cluster

This chapter includes the following topics:

Introduction

InfoScale in Containers Environment offers advanced security of data at rest with KMS-based (Key Management Server) encryption. You can use an external Key Management Server to manage the encryption key. Encryption is supported for volumes and disk groups. However, DR configuration is not supported for disk group level encryption. You must add an external CA certificate (custom CA certificate) first to enable connection with the Key Management Server (KMS).

Note: Whenever the InfoScale cluster is reconfigured, ensure that you generate the client certificate again and update the KMS server.

KMS-based encryption is not supported in the OpenShift-virtualization enviornment.

Adding a custom CA certificate

The following steps inform you how to add a custom CA certificate by using the `cfssl` tool. The steps are about a KubernetesCA certificate.

Note: As an Administrator, you can use any other compliant tool for signing. Refer to the procedure of that tool. Ensure that you set `"is_ca"` to `true`.

As a prerequisite, download the following packages from `github.com` repository by using python-pip.

`Tools`

- `cfssl`
- `cfssl-certinfo`
- `cfssljson`

If you are using external CA credentials, skip steps 1 to 6 and go to step 7. Ensure that you set `"is_ca"` to `true`.

Complete the following steps:

1. On the bastion node, create a directory - `custom-ca`. Navigate to this directory to perform the next steps.
2. Copy the following content into a file and save it as `vxconfig.json`.

```
{
  "signing": {
    "default": {
      "expiry": "43800h"
    },
    "profiles": {
      "cluster": {
        "expiry": "8760h",
        "usages": [
          "signing",
          "key encipherment",
          "cert sign",
          "server auth",
          "client auth"
        ],
        "ca_constraint": {
          "is_ca": true
        }
      }
    }
  }
}
```

3. Copy the following content into a file and save it as `csr_config.json`.

```
{
  "CN": "infoscale-ca",
  "key": {
    "algo": "rsa",
    "size": 2048
  },
  "hosts": [
    "kubernetes"
  ],
  "names": [
    {
      "O": "system:nodes",
      "OU": "vx"
    }
  ]
}
```

4. Run the following command to generate certificates:

```
cfssl genkey csr_config.json | cfssljson -bare infoscale-ca
```

Review output similar to the following output:

```
2022/02/10 15:09:27 [INFO] generate received request
2022/02/10 15:09:27 [INFO] received CSR
2022/02/10 15:09:27 [INFO] generating key: rsa-2048
2022/02/10 15:09:28 [INFO] encoded CSR
```

5. Now you must sign the certificate you just generated. Run the following command:

```
cfssl sign -ca /etc/kubernetes/pki/ca.crt -ca-key
/etc/kubernetes/pki/ca.key -hostname kubernetes -config
./vxconfig.json -profile cluster ./infoscale-ca.csr | cfssljson
-bare infoscale-ca
```

6 Run `ls` to list files in the folder.

Following files must be created:

```
infoscale-ca.csr  infoscale-ca-key.pem  infoscale-ca.pem
```

Here , `infoscale-ca.pem` is the external CA certificate.

7 Copy the following content into a file and save it as `custom-ca.yaml`.

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: infoscale-sds-operator
  name: infoscale-vtas
---
apiVersion: v1
kind: Secret
metadata:
  name: infoscale-ca
  namespace: <namespace of cert-manager>
type: kubernetes.io/tls
data:
  ca.crt: $(oc get cm kube-root-ca.crt -o
            jsonpath="{.data['ca\.crt']}"| base64 -w0)
  tls.crt: $(base64 ./infoscale-ca.pem | tr -d '\n')
  tls.key: $(base64 ./infoscale-ca-key.pem | tr -d '\n')
```

You have to replace content for `ca.crt`, `tls.crt`, and `tls.key`.

8 Run the following command:

```
oc get cm kube-root-ca.crt -o jsonpath="{.data['ca\.crt']}" |
base64 -w0
```

Copy the output of this command as:

```
<Content of ca.crt>
```

9 Modify `custom-ca.yaml` as under

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: infoscale-sds-operator
  name: infoscale-vtas
---
apiVersion: v1
kind: Secret
metadata:
  name: infoscale-ca
  namespace: <namespace of cert-manager>
type: kubernetes.io/tls
data:
  ca.crt: <Content of ca.crt>
  tls.crt: $(base64 ./infoscale-ca.pem | tr -d '\n')
  tls.key: $(base64 ./infoscale-ca-key.pem | tr -d '\n')
```

10 Similarly, run

`base64 ./infoscale-ca.pem | tr -d '\n'` and update `tls.crt` in `custom-ca.yaml` with the output of this command.

11 Run

`base64 ./infoscale-ca-key.pem | tr -d '\n'` and update `tls.key` in `custom-ca.yaml` with the output of this command.

12 Ensure that `custom-ca.yaml` is as follows:

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: infoscale-sds-operator
  name: <cert-manager namespace>
---
apiVersion: v1
kind: Secret
metadata:
  name: infoscale-ca
  namespace: <namespace of cert-manager>
type: kubernetes.io/tls
data:
  ca.crt: <Content of ca.crt>
  tls.crt: <Content of tls.crt>
  tls.key: <Content of tls.key>
```

13 Run `oc apply -f custom-ca.yaml`.

14 If you are configuring DR, copy this `custom-ca.yaml` to the DR cluster.

15 Run `oc apply -f custom-ca.yaml` on the DR cluster before applying license.

After `custom-ca.yaml` is successfully applied, you can apply `iso_rhel.yaml`. See the Installation section.

Note: This certificate is valid for 12 months. Ensure that you extend its validity in the eighth month.

Configuring InfoScale to enable transfer of keys

You must configure InfoScale to enable a connection with the Key Management Server (KMS) to transfer and save rest certs.

If you have created multiple InfoScale clusters, ensure you run steps 23 to 28 for every cluster.

The rest certs are renewed every three months and the renewed certs must be uploaded to the KMS server. Run steps 23 to 28 every three months for encryption to work.

Note: After a client rest cert is renewed, ensure that you add the renewed client cert to the client group on the KMS server.

Complete the following steps:

Note: The following steps inform you how to configure IBM Key Management Server. As an administrator, you can configure any KMIP-compliant server. Refer to the procedure of that KMIP-compliant server.

---

1. Be ready with the IP address and port number of the Key Management Server (KMS).
2. Run `echo "<IP address of the server >"| base64`

   Verify the output as under:

   ```
   Server output for base64
   ```

3. Run `echo "<Port number of the server >"| base64`

   Verify the output as under:

   ```
   Port number output for base64
   ```

4. Copy the following content into a file and save it as `infoscale-kmip-secret.yaml`.

   ```
   apiVersion: v1
   data:
     host: <Server output for base64>
     port: <Port number output for base64>
   kind: Secret
   metadata:
     name: infoscale-kmip-encrypt
     namespace: infoscale-vtas
   type: Opaque
   ```

5. Run `oc apply -f infoscale-kmip-secret.yaml` to deploy the InfoScale secret.
6. From another terminal, sign in to

   https://www.ibm.com/docs/en/sgklm/&4.1.1?topic=objects-registering-client-by-using-graphical-user-interface.

7 Select Advanced Configuration > Server Certificate. Click Add. The Add SSL/KMIP Certificate screen opens.

8 Select Request certificate from a third-party provider and enter values for Certificate label and Certificate description.

9 Click Add Certificate. The certificate is listed as Administer Server Certificates.

10 Review the Status of the certificate. The status is Certificate is pending.

11 From the bastion node, run `ssh root@<IP address of the KMS >`. Enter the password and login.

12 The certificate you just created is listed under `/opt/IBM/WebSphere/AppServer/products/sklm/data/` as `<Time stamp>_<Certificate name>.csr`.

Note: The path might vary depending on the KMS version.

---

13 Copy content of `/opt/IBM/WebSphere/Liberty/products/sklm/data/<Time stamp>_<Certificate name>.csr` into another file `<Copy of server cert content>.pem`.

14 Run `openssl x509 -req -in <Time stamp>_<Certificate name>.csr -CA infoscale-ca.pem -CAkey infoscale-ca-key.pem -CAcreateserial -out <server-certificate-name> -days 1024 -sha256`.

15 Review the output for the following message.

```
Signature ok
```

16 Copy `<Certificate name>.crt` to the root directory of the Key Management server.

17 On the Welcome screen of KMS, click Third-party certificates pending import.

18 In the Import Certificate screen, click Browse and navigate to the certificate you saved. Click Select.

19 Run `oc get secret -n infoscale-vtas`.

20 Review the output for the following:

```
NAME
infoscale-ca
```

21 Run `oc get secret -n infoscale-vtas`.

22 Review the output for the following:

```
NAME
infoscale-kmip-encrypt
```

23 Run `oc -n <namespace where cr is deployed> get secret infoscale-sds-rest-tls-cert-<cluster-id> -o jsonpath="{.data['tls\.crt']}" | base64 --decode > <kmip-device-client-cert>`.

24 Copy `<kmip-device-client-cert>` to the root directory of the KMS.

25 On the KMS, select Advanced Configuration > Client Device Certificates. Click Import.

26 In the Import SSL/KMIP Certificate for Clients window, assign a name and click Browse to select `<device-certificate>.crt` from the root directory.

27 Select the checkbox next to Allow the server to trust this certificate with the associated client device.

28 Click Import.

After a successful configuration, data is more secure and a need to back up keys required during Disaster Recovery is eliminated.

For a DR configuration

1 Complete steps 1 to 24 on one of the DR sites to configure `infoscale-kmip-encrypt` and the server certificate. Ensure that you configure `infoscale-kmip-encrypt` on all the sites. See steps 4 and 5.

2 Run the following command on the primary site to get the client certificate.

```
oc -n infoscale-vtas get secret
infoscale-sds-rest-tls-cert-<cluster-id> -o
jsonpath="{.data['tls\.crt']}" | base64 --decode >
<kmip-primary-cert>
```

3 Run the following command on the secondary site to get the client certificate.

```
oc -n infoscale-vtas get secret
infoscale-sds-rest-tls-cert-<cluster-id> -o
jsonpath="{.data['tls\.crt']}" | base64 --decode >
<kmip-secondary-cert>
```

4 Log on to the below URL and perform the following steps to register client and create client group.

https://www.ibm.com/docs/en/sgklm/4.1.1?topic=&objects-registering-client-by-using-graphical-user-interface

- Navigate to Clients > Clients (subsection) > Create > fill details. Enter <kmip-primary-cert> to register client.
- Similarly, enter <kmip-secondary-cert> to register client.

5 Log on to

https://www.ibm.com/docs/en/sgklm/4.1.1?&topic=mcgctco-creating-managing-client-group-by-using-graphical-user-interface

Navigate to Clients > Client Groups (subsection) > Add > Provide Client Group Name > Create . Select clients from the list and click Save.

6 Run steps from 23 to 28 again.

Renewing with an external CA certificate

External CA certificates typically get renewed a few months before the validity end date. A cluster administrator re-creates the new certificates and populates the InfoScale cluster.

The new CA certificates can then be applied before the validity end date, before the earlier certificates expire.

When the external certificate is issued by the intermediary of the CA and the issuer knows the intermediary, the content of `tls.crt` is a resulting certificate followed by a certificate chain. The certificate chain does not include a root CA certificate, as it is stored in `ca.crt`.

For InfoScale the external CA certificate is valid for 12 months and as an Administrator, you can initiate its renewal after the eighth month.

Note: Self-signing certificate is automatically renewed without any intervention. Validity of the self-signing certificate is four months and it is automatically renewed in the third month. However, external CA certificate needs to renewed.

Complete the following steps, ensuring that NTP is synchronized across nodes.

1. Run the following commands to renew the CA certificate.

   Note: This is an example of cfssl tool. You can also use any other tool to renew CA certificates. Refer to the procedure of that tool.

   For generating the certificate:

```
cfssl gencsr -key /infoscale-ca-key.pem /csr_config.json |
cfssljson -bare infoscale-ca
```

For signing the certificate:

```
cfssl sign -ca /etc/kubernetes/pki/ca.crt -ca-key
/etc/kubernetes/pki/ca.key -hostname kubernetes -config
./vxconfig.json -profile cluster ./infoscale-ca.csr | cfssljson
-bare infoscale-ca
```

2. Run the following command to generate the new secrets:

   ```
   sh gen-cert
   ```

3. Run the following command to update the new `infoscale-ca` secret with renewed `infoscale-ca.pem`:

   ```
   oc apply -f custom-ca.yaml
   ```

   Wait for upto five minutes.

4. Now you need to delete secrets of the following certificates:

   - infoscale-sds-rest-tls-cert-<value>
   - infoscale-csi-tls-cert
   - infoscale-fencing-tls-cert
   - iso-tls-cert
   - webhook-tls-cert
   - lico-tls-cert

   Run the following command to delete secrets of these certificates:

   ```
   oc delete secret -n infoscale-vtas <certificate name>
   ```

5. To enable encryption, perform the steps listed in the following section again:

   See "Configuring InfoScale to enable transfer of keys" on page 122.

After these certificates are deleted, the cert-manager automatically re-creates the new certificates. Wait for 15 minutes.

Note: If DR is configured, ensure that you run these commands on the secondary site immediately after running these commands on the primary site.

Chapter 8

Configuring KMS-based encryption on an Kubernetes cluster

This chapter includes the following topics:

Introduction

InfoScale in Containers environment offers advanced security of data at rest with KMS-based (Key Management Server) encryption. You can use an external Key Management Server to manage the encryption key. Encryption is supported for Volumes and disk groups. However, DR configuration is not supported for disk group level encryption. You must add an external CA certificate (custom CA certificate) first to enable connection with the Key Management Server (KMS).

Note: Whenever the InfoScale cluster is reconfigured, ensure that you generate the client certificate again and update the KMS server.

Adding a custom CA certificate

The following steps inform you how to add a custom CA certificate by using the `cfssl` tool.The steps are about a KubernetesCA certificate.

Note: As an Administrator, you can use any other compliant tool for signing. Refer to the procedure of that tool. Ensure that you set `"is_ca"` to `true`.

As a prerequisite, download the following packages from `github.com` repository by using python-pip.

`Tools`

- `cfssl`
- `cfssl-certinfo`
- `cfssljson`

If you are using external CA credentials, skip steps 1 to 6 and go to step 7. Ensure that you set `"is_ca"` to `true`.

Complete the following steps:

1. On the master node, create a directory - `custom-ca`. Navigate to this directory to perform the next steps.
2. Copy the following content into a file and save it as `vxconfig.json`.

```
{
  "signing": {
    "default": {
      "expiry": "43800h"
    },
    "profiles": {
      "cluster": {
        "expiry": "8760h",
        "usages": [
          "signing",
          "key encipherment",
          "cert sign",
          "server auth",
          "client auth"
        ],
        "ca_constraint": {
          "is_ca": true
        }
      }
    }
  }
}
```

3 Copy the following content into a file and save it as `csr_config.json`.

```
{
  "CN": "infoscale-ca",
  "key": {
    "algo": "rsa",
    "size": 2048
  },
  "hosts": [
    "kubernetes"
  ],
  "names": [
    {
      "O": "system:nodes",
      "OU": "vx"
    }
  ]
}
```

4 Run the following command to generate certificates:

`cfssl genkey csr_config.json | cfssljson -bare infoscale-ca`.

Review output similar to the following output:

```
2022/02/10 15:09:27 [INFO] generate received request
2022/02/10 15:09:27 [INFO] received CSR
2022/02/10 15:09:27 [INFO] generating key: rsa-2048
2022/02/10 15:09:28 [INFO] encoded CSR
```

5 Now you must sign the certificate you just generated. Run the following command.

```
cfssl sign -ca /etc/kubernetes/pki/ca.crt -ca-key
/etc/kubernetes/pki/ca.key -hostname kubernetes -config
./vxconfig.json -profile cluster ./infoscale-ca.csr | cfssljson
-bare infoscale-ca
```

6 Run `ls` to list files in the folder.

Following files must be created:

```
infoscale-ca.csr  infoscale-ca-key.pem  infoscale-ca.pem
```

Here , `infoscale-ca.pem` is the external CA certificate.

7 Copy the following content into a file and save it as `custom-ca.yaml`.

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: infoscale-sds-operator
  name: infoscale-vtas
---
apiVersion: v1
kind: Secret
metadata:
  name: infoscale-ca
  namespace: <namespace of cert-manager>
type: kubernetes.io/tls
data:
  ca.crt: $(kubectl get cm kube-root-ca.crt -o
             jsonpath="{.data['ca\.crt']}"| base64 -w0)

  tls.crt: $(base64 ./infoscale-ca.pem | tr -d '\n')
  tls.key: $(base64 ./infoscale-ca-key.pem | tr -d '\n')
```

You have to replace content for `ca.crt`, `tls.crt`, and `tls.key`.

8 Run

```
kubectl get cm kube-root-ca.crt -o jsonpath="{.data['ca\.crt']}"
| base64 -w0.
```

Copy the output of this command as:

```
<Content of ca.crt>
```

9 Modify `custom-ca.yaml` as under:

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: infoscale-sds-operator
  name: infoscale-vtas
---
apiVersion: v1
kind: Secret
metadata:
  name: infoscale-ca
  namespace: <namespace of cert-manager>
type: kubernetes.io/tls
data:
  ca.crt: <Content of ca.crt>
  tls.crt: $(base64 ./infoscale-ca.pem | tr -d '\n')
  tls.key: $(base64 ./infoscale-ca-key.pem | tr -d '\n')
```

10 Similarly, run

`base64 ./infoscale-ca.pem | tr -d '\n'` and update `tls.crt` in `custom-ca.yaml` with the output of this command.

11 Run

`base64 ./infoscale-ca-key.pem | tr -d '\n'` and update `tls.key` in `custom-ca.yaml` with the output of this command.

12 Ensure that `custom-ca.yaml` is as under:

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: infoscale-sds-operator
  name: infoscale-vtas
---
apiVersion: v1
kind: Secret
metadata:
  name: infoscale-ca
  namespace: <namespace of cert-manager>
type: kubernetes.io/tls
data:
  ca.crt: <Content of ca.crt>
  tls.crt: <Content of tls.crt>
  tls.key: <Content of tls.key>
```

13 Run `kubectl apply -f custom-ca.yaml`.

14 If you are configuring DR, copy this `custom-ca.yaml` to the DR cluster.

15 Run `kubectl apply -f custom-ca.yaml` on the DR cluster before applying the license.

After `custom-ca.yaml` is successfully applied, you can apply `iso_rhel.yaml`. See the Installation section.

Note: This certificate is valid for 12 months. Ensure that you extend its validity in the eighth month.

Configuring InfoScale to enable transfer of keys

You must configure InfoScale to enable a connection with the Key Management Server (KMS) to transfer and save rest certs.

If you have created multiple InfoScale clusters, ensure you run steps 23 to 28 for every cluster.

The rest certs are renewed every three months and the renewed certs must be uploaded to the KMS server. Run steps 23 to 28 every three months for encryption to work.

Note: After a client rest cert is renewed, ensure that you add the renewed client cert to the client group on the KMS server.

Complete the following steps:

Note: The following steps inform you how to configure IBM Key Management Server. As an administrator, you can configure any KMIP-compliant server. Refer to the procedure of that KMIP-compliant server.

---

1. Be ready with the IP address and port number of the Key Management Server (KMS).
2. Run `echo "<IP address of the server >"| base64`

   Verify the output as under:

   ```
   Server output for base64
   ```

3. Run `echo "<Port number of the server >"| base64`

   Verify the output as under:

   ```
   Port number output for base64
   ```

4. Copy the following content into a file and save it as `infoscale-kmip-secret.yaml`.

   ```
   apiVersion: v1
   data:
     host: <Server output for base64>
     port: <Port number output for base64>
   kind: Secret
   metadata:
     name: infoscale-kmip-encrypt
     namespace: infoscale-vtas
   type: Opaque
   ```

5. Run `kubectl apply -f infoscale-kmip-secret.yaml` to deploy the InfoScale secret.

6 From another terminal, log on to

https://www.ibm.com/docs/en/sgklm/4.1.1?topic=objects-registering-client-by-using-graphical-user-interface.

7 Select Advanced Configuration > Server Certificate. Click Add. The Add SSL/KMIP Certificate screen opens.

8 Select Request certificate from a third-party provider and enter values for Certificate label and Certificate description.

9 Click Add Certificate. The certificate is listed as Administer Server Certificates.

10 Review the Status of the certificate. The status is Certificate is pending.

11 From the master node, run `ssh root@<IP address of the KMS >`. Enter the password and login.

12 The certificate you just created is listed under `/opt/IBM/WebSphere/AppServer/products/sklm/data/` as `<Time stamp>_<Certificate name>.csr`.

Note: The path might vary depending on the KMS version.

---

13 Copy content of `/opt/IBM/WebSphere/Liberty/products/sklm/data/<Time stamp>_<Certificate name>.csr` into another file `<Copy of server cert content>.pem`.

14 Run `openssl x509 -req -in <Time stamp>_<Certificate name>.csr -CA infoscale-ca.pem -CAkey infoscale-ca-key.pem -CAcreateserial -out <server-certificate-name> -days 1024 -sha256`.

15 Review the output for the following message:

```
Signature ok
```

16 Copy `<Certificate name>.crt` to the root directory of the Key Management server.

17 On the Welcome screen of KMS, click Third-party certificates pending import.

18 In the Import Certificate screen, click Browse and navigate to the certificate you saved. Click Select.

19 Run `kubectl get secret -n infoscale-vtas`.

20 Review the output for the following:

```
NAME
infoscale-ca
```

21 Run `kubectl get secret -n infoscale-vtas`.

22 Review the output for the following:

```
NAME
infoscale-kmip-encrypt
```

23 Run `kubectl -n <namespace where cr is deployed> get secret infoscale-sds-rest-tls-cert-<cluster-id> -o jsonpath="{.data['tls\.crt']}" | base64 --decode > <kmip-device-client-cert>`.

24 Copy `<kmip-device-client-cert>` to the root directory of the KMS.

25 On the KMS, select Advanced Configuration > Client Device Certificates. Click Import.

26 In the Import SSL/KMIP Certificate for Clients window, assign a name and click Browse to select `<device-certificate>.crt` from the root directory.

27 Select the checkbox next to Allow the server to trust this certificate with the associated client device.

28 Click Import.

After a successful configuration, data is more secure and a need to back up keys required during Disaster Recovery is eliminated.

For a DR configuration

1 Complete steps 1 to 24 on one of the DR sites to configure `infoscale-kmip-encrypt` and the server certificate. Ensure that you configure `infoscale-kmip-encrypt` on all the sites. See steps 4 and 5.

2 Run the following command on the primary site to get the client certificate.

```
kubectl -n infoscale-vtas get secret
infoscale-sds-rest-tls-cert-<cluster-id> -o
jsonpath="{.data['tls\.crt']}" | base64 --decode >
<kmip-primary-cert>
```

3 Run the following command on the secondary site to get the client certificate:

```
kubectl -n infoscale-vtas get secret
infoscale-sds-rest-tls-cert-<cluster-id> -o
jsonpath="{.data['tls\.crt']}" | base64 --decode >
<kmip-secondary-cert>
```

4 Log on to the below URL and perform the following steps to register client and create client group.

https://www.ibm.com/docs/en/sgklm/4.1.1?topic=objects-registering-client-by-using-graphical-user-interface

- Navigate to Clients > Clients (subsection) > Create > fill details. Enter <kmip-primary-cert> to register client.
- Similarly, enter <kmip-secondary-cert> to register client.

5 Log on to

https://www.ibm.com/docs/en/sgklm/4.1.1?topic=mcgctco-creating-managing-client-group-by-using-graphical-user-interface

Navigate to Clients > Client Groups (subsection) > Add > Provide Client Group Name > Create . Select clients from the list and click Save.

6 Run steps 23 to 28 again.

Renewing with an external CA certificate

External CA certificates typically get renewed a few months before the validity end date. A cluster administrator re-creates the new certificates and populates the InfoScale cluster.

The new CA certificates can then be applied before the validity end date, before the earlier certificates expire.

When the external certificate is issued by the intermediary of the CA and the issuer knows the intermediary, the content of `tls.crt` is a resulting certificate followed by a certificate chain. The certificate chain does not include a root CA certificate, as it is stored in `ca.crt`.

For InfoScale the external CA certificate is valid for 12 months and as an Administrator, you can initiate its renewal after the eighth month.

Note: Self-signing certificate is automatically renewed without any intervention. Validity of the self-signing certificate is four months and it is automatically renewed in the third month. However, external CA certificate needs to renewed.

Complete the following steps, ensuring that NTP is synchronized across nodes.

1. Run the following commands to renew the CA certificate.

   Note: This is an example of cfssl tool. You can also use any other tool to renew CA certificates. Refer to the procedure of that tool.

   For generating the certificate:

   ```
   cfssl gencsr -key /infoscale-ca-key.pem /csr_config.json |
   cfssljson -bare infoscale-ca
   ```

   For signing the certificate:

   ```
   cfssl sign -ca /etc/kubernetes/pki/ca.crt -ca-key
   /etc/kubernetes/pki/ca.key -hostname kubernetes -config
   ./vxconfig.json -profile cluster ./infoscale-ca.csr | cfssljson
   -bare infoscale-ca
   ```

2. Run the following command to generate the new secrets:

   ```
   sh gen-cert
   ```

3. Run the following command to update the new `infoscale-ca` secret with renewed `infoscale-ca.pem`:

   ```
   kubectl apply -f custom-ca.yaml
   ```

   Wait for upto five minutes.

4. Now you need to delete secrets of the following certificates:

   - infoscale-sds-rest-tls-cert-<value>
   - infoscale-csi-tls-cert
   - infoscale-fencing-tls-cert
   - iso-tls-cert
   - webhook-tls-cert
   - lico-tls-cert

   Run the following command to delete secrets of these certificates:

   ```
   kubectl delete secret -n infoscale-vtas <certificate name>
   ```

5. To enable encryption, perform the steps listed in the following topic again:

   See “Configuring InfoScale to enable transfer of keys” on page 134.

After these certificates are deleted, the cert-manager automatically re-creates the new certificates. Wait for 15 minutes.

Note: If DR is configured, ensure that you run these commands on the secondary site immediately after running these commands on the primary site.

Chapter 9

InfoScale CSI deployment in Container environment

This chapter includes the following topics:

CSI plugin deployment

CSI (Container Storage Interface) is a standardized mechanism for Container Orchestrators (COs) to expose arbitrary storage systems to their containerized workloads. InfoScale CSI plugin is used to provide persistent storage to OpenShift or Kubernetes. InfoScale CSI also supports creation of storage classes for high availability, performance, and capacity. It also supports online expansion of capacity as well as snapshot and clone functionality.

InfoScale CSI is automatically deployed while installing InfoScale on OpenShift or Kubernetes.

After you download, unzip, and untar `infoscale-yamls-v9.1.2.tar.gz` , a folder `/infoscale-yamls-v9.1.2/infoscale-csi` is automatically created. Within `/infoscale-yamls-v9.1.2/infoscale-csi`, following sub folders are created and the files listed are saved.

Note: The commands listed in this chapter are applicable to OpenShift. If you are on Kubernetes, replace `oc` with `kubectl`.

- `-- dynamic-provisioning`
  - `-- csi-dynamic-pvc.yaml`
  - `-- csi-dynamic-snapshot-restore.yaml`
  - `-- csi-dynamic-snapshot.yaml`
  - `-- csi-dynamic-volume-clone.yaml`
  - `-- csi-pod.yaml`
  - `-- csi-dynamic-block-pvc.yaml`
  - `-- csi-block-pod.yaml`
- `-- snapshot-class-templates`
  - `-- csi-infoscale-snapclass.yaml`
- `-- static-provisioning`
  - `-- csi-pod.yaml`
  - `-- csi-static-pvc.yaml`
  - `-- csi-static-pv-k8s.yaml`
  - `-- csi-static-pv-ocp.yaml`

  - `-- csi-static-snapshot-content.yaml`
  - `-- csi-static-snapshot.yaml`
  - `-- csi-static-snapshot-restore.yaml`
  - `-- csi-static-block-pv-k8s.yaml`
  - `-- csi-static-block-pv-ocp.yaml`
  - `-- csi-static-block-pvc.yaml`
  - `-- csi-static-block-pod.yaml`
- `-- storage-class-templates`
  - `-- csi-infoscale-performance-sc.yaml`
  - `-- csi-infoscale-resiliency-sc.yaml`
  - `-- csi-infoscale-sc.yaml`
  - `-- csi-infoscale-topology-sc.yaml`
  - `-- csi-multicluster-app.yaml`
  - `-- csi-ephemeral-pod.yaml`

After CSI deployment is complete, you can create YAML files specific to your requirements and use these for:

- Dynamic provisioning of volumes
- Static provisioning of volumes
- Snapshot provisioning (Creating volume snapshots)
- Creating volume clones
- Enabling raw block volume support

InfoScale CSI supports static and dynamic provisioning of volumes on shared storage as well as shared nothing storage (FSS).

Note: Minimum volume size should be 1Gi when volumes are created either statically or dynamically.

Note: Only one disk group - `vrts_kube_dg_<cluster id>` is supported for all CSI operations, and the same disk group is used throughout the CSI plugin lifecycle. The command examples are applicable to OpenShift. For Kubernetes, replace `oc` by `kubectl`. `vrts_kube_dg` is created automatically during cluster creation by using disks which are not under any other File System or Logical Volume Manager.

An application container requests for the required storage through a Persistent Volume claim (PVC). The PVC uses the storage class to identify and provision the Persistent Volume that belongs to the storage class. After the volume is created, a Persistent Volume object is created and is bound to the PVC, and persistent storage is made available to the application.

While provisioning volumes, the InfoScale CSI plugin supports the following access modes that determine how the volumes can be mounted:

- ReadWriteOnce (RWO) -- the volume can be mounted as read-write by a single node.
- ReadOnlyMany (ROX) -- the volume can be mounted read-only by many nodes.
- ReadWriteMany (RWX) -- the volume can be mounted as read-write by many nodes.

Note: The permission in a Persistent Volume Claim is per node and not per pod. For example, a PVC with RWO mode does not prevent mounting same volume in more than one pod on same node.

Raw block volume support

Using raw block volume, the block device is directly exposed instead of exposing the mountpoint inside an application container. Applications can directly write or can create their own native filesystem on the provided block device.

Raw block volumes support is required for applications that are capable of accessing the device directly to store data. Database application is an example where data is directly accessed from the underlying storage, which is more efficient. To enable raw block volume support, `volumeMode` must be set to `Block`.

Following are the FileSystem-persistent volumes features which are supported on Block volumes.

- Create and Delete Volumes
- Expand Volumes
- Clone Volumes

- Create and Delete Snapshot
- Snapshot Restore

Note: Access mode RWO (ReadWriteOnce) is not supported for block volumes. Creation of a RWO block volume does not fail. The volume is created with RWX (ReadWriteMany) properties even when it is listed as RWO. For ROX (ReadOnlyMany) block volumes, the read-only condition is enforced on non-root users only.

For Block volumes, the CSI driver does not format the block device; it just binds the block device to the target path.

Static provisioning

You can use static provisioning if you want to make the existing persistent storage objects available to the cluster. You can statically provision a volume over shared storage (CVM) and shared nothing (FSS) storage.

Static provisioning allows cluster administrators to make existing storage objects available to a cluster. To use static provisioning, you must know the details of the storage object, its supported configurations, and mount options. To make existing storage available to a cluster user, you must manually create a Persistent Volume, and a Persistent Volume Claim before referencing the storage in a pod.

Note: In case you want to use filesystem-persistent volumes, ensure that the InfoScale File System is created before provisioning the volumes. If the InfoScale File System does not exist, you must create it manually by using the `mkfs` command from the InfoScale SDS pod.

Creating Static Provisioning

1 You can create a Storage Class by running the `csi-infoscale-sc.yaml` file which is as under:

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-infoscale-sc
annotations:
  storageclass.kubernetes.io/is-default-class: "false"
provisioner: org.veritas.infoscale
reclaimPolicy: Delete
allowVolumeExpansion: true
parameters: fstype: vxfs

  # (optional) Specifies a volume layout type.
  # Supported layouts: stripe, mirror, stripe-mirror,,mirror-stripe,
  #  concat, concat-mirror, mirror-concat
  # If omitted, InfoScale internally chooses the best suited layout
  #   based on the environment.
  # layout: "mirror"

  # (optional) Specifies the number of disk or host failures a storage
  #                              object can tolerate.
  # faultTolerance: "1"


# (optional) Specifies the number of stripe columns to use when
  # creating a striped volume.
  # nstripe: "3"

  # (optional) Specifies the stripe unit size to use
  #  for striped volume.
  # stripeUnit: "64k"

  # (optional) Specifies disks with the specified media type.
  #  All disks with the given mediatype are selected for volume creation.
  # Supported values: hdd, ssd
  # mediaType: "hdd"

  # (optional) Specifies whether to store encrypted data on disks or not.
  # Valid values are true or false
  # encryption: "false"
```

```
#(optional) Specifies how to initialize a new volume.
# Valid values are "active", "zero" and "sync"
# initType: "active"
```

Note: The supported `initType` values are `"sync"`, `"active"`, or `"zero"`.

Run `oc/kubectl create -f csi-infoscale-sc.yaml`.

2 You must be ready with the VxVM volume name to define the Persistent Volume object.

Run `oc/kubectl exec -ti -n <namespace> <InfoScale SDS> -- <cmd>` to list Volumes from the InfoScale SDS pod.

An example of this command is `oc/kubectl exec -ti -n infoscale-vtas infoscale-sds-rhel9-bwvwb -- vxprint -g vrts_kube_dg -vuh | grep -w fsgen`.

3 In the `csi-static-pv-ocp.yaml` or `csi-static-pv-k8s.yaml`, define the Persistent Volume object and specify the existing VxVM volume name in the volumeHandle attribute.

```
csi-static-pv-<ocp/k8s>.yaml
---

apiVersion: v1

kind: PersistentVolume

metadata:

  name: csi-infoscale-pv

  annotations:

    pv.kubernetes.io/provisioned-by: org.veritas.infoscale

spec:

  storageClassName: csi-infoscale-sc

  persistentVolumeReclaimPolicy: Retain

  capacity:

    storage: 2Gi

  accessModes:

    - ReadWriteOnce

  csi:

    driver: org.veritas.infoscale

    # Please provide pre-provisioned Infoscale volume name.

    volumeHandle: clust_<cluster_id>/vrts_kube_dg-<cluster_id>/testVol

    fsType: vxfs
```

Note: Here, `clusterid` is what you use while configuring `cr.yaml`.

---

4 Add the following to `csi-static-pv-ocp.yaml` or `csi-static-pv-k8s.yaml` for node affinity.

```
nodeAffinity:
      required:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.org.veritas.infoscale/cluster
            operator: In
            values:
            - <Name of the InfoScale cluster>
```

Note: You can skip this step and add nodeSelector: topology.org.veritas.infoscale/cluster: "new1" to `csi-pod.yaml`.

---

5 Create a Persistent Volume using the yaml.

```
oc create -f csi-static-pv-ocp.yaml
```

or

```
kubectl create -f csi-static-pv-k8s.yaml
```

6 Define the Persistent Volume Claim (PVC) with appropriate access mode and storage capacity.

```
csi-static-pvc.yaml
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-infoscale-pvc
spec:
  accessModes:
   - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: csi-infoscale-sc
```

7 Create a Persistent Volume Claim by using the yaml. This PVC automatically gets bound with the newly created PV.

```
oc/kubectl create -f csi-static-pvc.yaml
```

8 Update the application yaml file (`mysql-deployment.yaml`) and specify the persistent Volume Claim name.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql-deployment
  labels:
    app: mysql
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:latest
          ports:
            - containerPort: 3306
          volumeMounts:
            - mountPath: "/var/lib/mysql"
              name: mysql-data
          env:
            - name: MYSQL_ROOT_PASSWORD
              value: root123
      volumes:
        - name: mysql-data
          persistentVolumeClaim:
            claimName: csi-infoscale-pvc
```

9 Create the application pod.

```
oc/kubectl create -f mysql-deployment.yaml
```

10 Check that old data exists on the persistent volume. Run the following commands:

```
oc/kubectl get pods | grep mysql
```

```
oc/kubectl exec -it mysql-deployment<id> -- mysql -uroot
-pRoot12345!
```

Enabling raw block support with static provisioning

1 Run `oc/kubectl exec -ti -n <namespace> <InfoScale SDS> -- <cmd>` to list Volumes from the InfoScale SDS pod. Be ready with the names of the Volumes.

2 Update `volumeHandle` in `csi-static-block-pv-<ocp/k8s>.yaml` as under:

```
---
apiVersion: v1

kind: PersistentVolume
metadata:
  name: csi-infoscale-block-pv
  annotations:
    pv.kubernetes.io/provisioned-by: org.veritas.infoscale
spec:
  volumeMode: Block
  storageClassName: csi-infoscale-sc
  persistentVolumeReclaimPolicy: Retain
  capacity:
    storage: 5Gi

  accessModes:
    - ReadWriteOnce

  csi:
    driver: org.veritas.infoscale

    # Please provide pre-provisioned Infoscale volume name.
    volumeHandle: clust_<cluster_id>/vrts_kube_dg-<cluster_id>/testVol

    volumeAttributes:
      volumePath: "/dev/blk"
```

Note: Here, `clusterid` is what you use while configuring `cr.yaml`.

3 Run `oc create -f csi-static-pv-ocp.yaml` or `kubectl create -f csi-static-pv-k8s.yaml` to apply the yaml.

4 Define the Persistent Volume Claim (PVC) with appropriate access mode and storage capacity in `csi-static-block-pvc.yaml` as under:

```
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-infoscale-block-pvc
spec:
  volumeMode: Block
  accessModes:
   - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: csi-infoscale-sc
```

5 Add the following to `csi-static-block-pv-<ocp/k8s>.yaml` for node affinity.

```
nodeAffinity:
      required:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.org.veritas.infoscale/cluster
            operator: In
            values:
            - <Name of the InfoScale cluster>
```

Note: You can skip this step and add nodeSelector: topology.org.veritas.infoscale/cluster: "new1" to `csi-pod.yaml`.

---

6 Run `oc/kubectl create -f csi-static-block-pvc.yaml` to apply the yaml.

7 Update the Persistent Volume Claim in `csi-static-block-pod.yaml` as under:

```
---
apiVersion: v1
kind: Pod
metadata:
  name: redis
  labels:
    name: redis
spec:
  containers:
    - name: redis
      image: redis
      imagePullPolicy: IfNotPresent
      volumeDevices:
        - devicePath: "/dev/blk"
          name: vol1
  volumes:
    - name: vol1
      persistentVolumeClaim:
        claimName: csi-infoscale-block-pvc
```

8 Run `oc create -f csi-static-block-pod.yaml` to create the application pod.

Dynamic provisioning

You can dynamically provision a volume over shared storage (CVM) and shared nothing (FSS) storage. In dynamic provisioning, you must create a Storage Class that define the storage provisioner and the required parameters in the storage class yaml file and create the Persistent Volume Claim. The Pod references the Storage Class through an existing Persistent Volume Claim and dynamically allocates storage for the requesting Pod.

While allocating storage to pods dynamically, you can reclaim the storage when the previously provisioned storage is available for other applications to use. You can resize an existing volume using the Persistent Volume Claim (PVC) object.

Perform the following steps for allocating storage dynamically to container workloads:

1. Create a Storage Class using a yaml file.

```
oc create -f csi-infoscale-sc.yaml
```

2. Define the Persistent Volume Claim and specify the appropriate Storage Class, access mode, and the required storage size.

```
csi-dynamic-pvc.yaml
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-infoscale-pvc
spec:
  storageClassName: csi-infoscale-sc
  accessModes:
    - ReadWriteMany

  resources:
    requests:
      storage: 5Gi
```

3. Create a Persistent Volume Claim using the yaml.

```
oc create -f csi-dynamic-pvc.yaml
```

4. Update `csi-mysql-app.yaml` and specify the Persistent Volume Claim name.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql-deployment
  labels:
    app: mysql
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
```

```
        - name: mysql
          image: mysql:latest
          ports:
            - containerPort: 3306
          volumeMounts:
            - mountPath: "/var/lib/mysql"
              name: mysql-data
          env:
            - name: MYSQL_ROOT_PASSWORD
              value: root123
      volumes:
        - name: mysql-data
          persistentVolumeClaim:
            claimName: csi-infoscale-pvc
```

5. Create the application pod.

   `oc create -f csi-mysql-app.yaml`

   After the pod is created, start using the InfoScale PVC as a Persistent Storage.

Enabling raw block volume support with dynamic provisioning

1 Ensure that `volumeMode` is `Block` in as under in `csi-dynamic-block-pvc.yaml`
.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-infoscale-block-pvc
spec:
  volumeMode: Block
  storageClassName: csi-infoscale-sc
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 4Gi
```

2 Run the following command to set Block as the VolumeMode `oc apply -f csi-dynamic-block-pvc.yaml`

Ensure that you resize the file system and drain I/O's from the file system under snapshots and clones for crash consistent snapshots.

After enabling raw block volume support, applications can write to the block device or create filesystem at the block level. Online resizing of block without detaching the block storage is supported. However, Resizing can be to a higher value only.

Reclaiming provisioned storage

When a previously provisioned storage is no longer required by an application, you can delete the corresponding PVC objects from the APIs and reclaim the storage for other applications to use. The reclaim policy for a Persistent Volume states what action the cluster must take on the volume after it is released from the PVC. You can use the following command to delete a PVC:

```
oc delete pvc <pvc_name>
```

InfoScale supports the following reclaim policies. You must specify the reclaim policy while creating a storage class for dynamic provisioning.

- Retain: Indicates that the Persistent Volume must be reclaimed manually.
- Delete: (Default) Indicates that the Persistent Volume and the associated storage gets automatically deleted when the PVC is deleted.

For more information on reclaim policies, see Kubernetes - Persistent Volumes documentation.

Resizing Persistent Volumes (CSI volume expansion)

Using the persistent volume expansion feature, you can easily expand the storage capacity of a persistent volume by just updating the Persistent Volume Claim storage specification. However, to use this feature, you must set the `allowVolumeExpansion` attribute to true in their `StorageClass` object. Only the PVCs created by using such Storage Class allow volume expansion. When the storage attribute of such a PVC object is updated, Container Orchestrator interprets it as a change request and triggers automatic volume resizing.

The following sample Storage Class `yaml` shows the `allowVolumeExpansion` attribute definition.

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-infoscale-sc
  annotations:
```

```
    storageclass.kubernetes.io/is-default-class: "false"
provisioner: org.veritas.infoscale
reclaimPolicy: Delete
allowVolumeExpansion: true
parameters:
  fstype: vxfs
  layout: "mirror"
```

While performing the volume expansion operation, you must note the following:

- Ensure that the PVC is in use by some application pod while performing resize operation. InfoScale supports dynamic volume expansion in 'Online' mode.
- Do not perform the volume expansion operation from InfoScale SDS pods. In such case, OpenShift is not aware of these changes and updated volume size is not reflected in the PV and PVC objects.
- InfoScale does not support the shrinking of persistent volume as OpenShift does not support it.
- If this PVC is a part of any Disaster Recovery (DR) configuration, DR controller automatically detects this change and underlying InfoScale Volume is expanded on the secondary cluster also. If PVC is resized on a primary cluster and if PVC with an identical name is available on secondary cluster, you must manually update the PVC size on the secondary cluster. Volume is resized automatically on the secondary cluster.
- When a volume is used in any InfoScale configuration, make sure that you use only InfoScale commands to resize it. Do not use any other methods—like OS-specific commands—to resize such volumes. Doing so may disrupt the functioning of your InfoScale configuration.

After the volume is provisioned, create the container application pod, run the application, and access the volume. If the volume is full and must be resized, use one of the following ways:

- Edit the Persistent Volume Claim
- Use the `oc patch pvc` command

Note: Resize operation on volume is not supported when it is provisioned in ReadOnlyMany mode.

Resizing a Persistent Volume by editing the Persistent Volume Claim

1. Find the Persistent Volume Claim to resize.

   ```
   oc get pvc
   ```

   Output similar to this is displayed:

   ```
   NAME              STATUS  VOLUME   CAPACITY
   csi-infoscale-pvc Bound   pvc-<id> 5Gi

   ACCESS MODES STORAGECLASS     AGE
   RWX          csi-infoscale-sc 32m
   ```

2. To resize the storage capacity, edit the PVC.

   ```
   oc edit pvc csi-infoscale-pvc
   ```

3. From your text editor, change the storage capacity to the required larger value. For example, from 5Gi to 10Gi .

4. Check the status of the Persistent Volume Claim and Persistent Volume to verify if the size is updated.

   ```
   oc get pvc
   ```

   Output similar to this is displayed:

   ```
   NAME               STATUS VOLUME
   csi-infoscale-pvc  Bound  pvc-<id>

   CAPACITY ACCESS MODES STORAGECLASS     AGE
   10Gi     RWX          csi-infoscale-sc 32m
   ```

Resizing a Persistent Volume using the 'patch pvc' command

1. Find the Persistent Volume Claim to resize.

   ```
   oc get pvc
   ```

   Output similar to this is displayed:

   ```
   NAME               STATUS VOLUME   CAPACITY
   csi-infoscale-pvc Bound  pvc-<id> 5Gi


   ACCESS MODES STORAGECLASS     AGE
   RWX          csi-infoscale-sc 32m
   ```

2. To resize the storage capacity to the required larger value, for example, from 5Gi to 10Gi, run the following command.

   ```
    oc patch pvc csi-infoscale-pvc --patch
   '{"spec": {"resources": {"requests": {"storage": "10Gi"}}}}'
   ```

3. Check the status of the Persistent Volume Claim and Persistent Volume to verify if the size is updated.

   ```
    oc get pvc
   ```

   Output similar to this is displayed:

   ```
   NAME                 STATUS   VOLUME
   csi-infoscale-pvc    Bound    pvc-<id>


   CAPACITY   ACCESS MODES STORAGECLASS     AGE
   10Gi       RWX          csi-infoscale-sc 32m
   ```

Snapshot provisioning (Creating volume snapshots)

Volume snapshot represents a point-in-time and space-optimized copy of volume on storage system. The InfoScale CSI Plugin supports snapshot provisioning. You can create one or more snapshots of Persistent Volume that is provisioned dynamically or statically. You can also restore a Snapshot to reinstate the volume contents on a completely new Persistent Volume that you want to provision. The snapshots can also be consumed directly as PVC through static provisioning.

For using the point-in-time copies, InfoScale recommends that you:

- Use the space-optimized snapshots for read-intensive applications that run on top of either a source Persistent Volume or a snapshot copy. You can use full-instant snapshots for the write-intensive applications.
- Use the Volume Clones feature for write- intensive applications. The volume Clones makes the exact copy of a Persistent volume immediately available for the read, write, and update operations.

To create a snapshot, you must create the following objects by using the `yaml` files:

1. A `VolumeSnapshotContent` is a cluster resource to create a snapshot of a volume in the cluster that is provisioned by an administrator. This resource is similar to a `PersistentVolume`.
2. A `VolumeSnapshot` is a cluster resource for request to create a snapshot of a volume in the cluster that is provisioned by a user. This resource is similar to a `PersistentVolumeClaim`.
3. A `VolumeSnapshotClass` describe the storage classes when provisioning a volume snapshot. The `VolumeSnapshotClass` acts as a template for creating a snapshot and includes attributes like the type of snapshot, synchronization parameters, and other configuration parameters.

Note: The `VolumeSnapshot`, `VolumeSnapshotContent`, and `VolumeSnapshotClass` API objects are Custom Resource Definitions (CRDs) and not a part of the core API. These CRDs and snapshot-controller are pre-installed on OpenShift, but must be manually deployed on Kubernetes.

Ensure that you install CRDs version v1 only and snapshot-controller version 6.1.0 or lower. For more details, see https://github.com/kubernetes-csi/external-snapshotter.

In the beta version of `VolumeSnapshot`, you must deploy a snapshot controller into the control plane.

Dynamic provisioning of a snapshot

To perform Dynamic provisioning of a snapshot:

1. Define the `VolumeSnapshotClass` object using the `yaml` and specify the `deletionPolicy` and `snapType`.

```
csi-infoscale-snapclass.yaml
---
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshotClass
metadata:
  name: csi-infoscale-snapclass
  annotations:
    snapshot.storage.kubernetes.io/is-default-class: "true"
driver: org.veritas.infoscale
deletionPolicy: Delete
#parameters:
  # (optional) Specifies the type of the snapshot to be created.
  # If omitted,  by default creates space-optimized snapshot.
  # Supported values: space-optimized, full-instant
  # snapType: space-optimized

  # (optional) Specifies the size of the cache volume
    to be created for space-optimized snapshots.
  # If omitted, InfoScale internally chooses the
    cacheSize as 30% of orignial volume size.
  # cacheSize: 500m
```

2. Create Volume Snapshot Class:

```
oc create -f csi-infoscale-snapclass.yaml
```

3. Define the Volume Snapshot.

```
csi-dynamic-snapshot.yaml
---
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshot
metadata:
  name: csi-dynamic-snapshot
spec:
  volumeSnapshotClassName: csi-infoscale-snapclass
  source:
    persistentVolumeClaimName: csi-infoscale-pvc
```

4 Create Volume Snapshot:

```
oc create -f csi-dynamic-snapshot.yaml
```

5 On successful creation of a snapshot, the corresponding volume snapshot content is created and bound to the volume Snapshot object.

Static provisioning of an existing snapshot

To perform static Provisioning of an existing snapshot:

1 You must be ready with the VxVM volume name to define the Persistent Volume object.

Run

```
oc exec -ti -n <namespace> <InfoScale SDS> -- <cmd>
```

to list Volumes from the InfoScale SDS pod. You have to specify a Volume for `snapshotHandle` in `csi-static-snapshot-content.yaml`.

An example of this command is

```
oc exec -ti -n infoscale-vtas infoscale-sds-rhel9-bwvwb -- vxprint
-g vrts_kube_dg -vuh | grep -w fsgen
```

2 Define the volume snapshot content object using the `yaml` file and specify the `snapshotHandle`.

```
csi-static-snapshot-content.yaml
---
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshotContent
metadata:
name: csi-static-snapshot-content
spec:
deletionPolicy: Retain
driver: org.veritas.infoscale
source:
# Provide pre-provisioned Infoscale snapshot volume name
snapshotHandle: clus_<clusterid>/<dgname>/<volname>
volumeSnapshotRef:
name: csi-static-snapshot
namespace: default
```

3 Define the volume snapshot object using the `yaml` and specify the `volumeSnapshotContentName`.

```
csi-static-snapshot.yaml
---
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshot
metadata:
name: csi-static-snapshot
spec:
volumeSnapshotClassName: csi-infoscale-snapclass
source:
volumeSnapshotContentName: csi-static-snapshot-content
```

4 Create Volume Snapshot

```
oc create -f csi-static-snapshot.yaml
```

On successful creation of VolumeSnapshot object, the corresponding volume snapshot content is bound to the volume Snapshot object.

Using a snapshot

You can use the snapshots created using the `VolumeSnapshot` request by restoring them to a new PVC and provisioning that PVC with the pre-populated data from snapshot to an application pod.

You can also use the snapshot volumes as static Persistent Volumes by specifying the snapshot volume name as a value for the `volumeHandle` parameter while provisioning a static PV.

Restoring a snapshot to new PVC

If you want to use and update a point-in-time copy of the application data, you can restore the snapshot of that application's persistent volume to a new persistent volume that represents the previous state described by the snapshot. To restore a volume from a snapshot, you must specify the name of the `VolumeSnapshot` object that you want to restore as the value of the `dataSource` attribute.

Note: While restoring a snapshot or a clone to a new PVC, you must specify the exact same storage details as specified in the source PVC.

To restore a snapshot to a new PVC:

1. Define a Persistent Volume Claim object using the `yaml` and specify the name of the `VolumeSnapshot` object that you want to restore in the `dataSource` attribute:

```
csi-dynamic-snapshot-restore.yaml
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: csi-infoscale-snapshot-restore
spec:
storageClassName: csi-infoscale-sc
accessModes:
- ReadWriteMany
resources:
requests:
storage: 5Gi
dataSource:
name: csi-dynamic-snapshot
kind: VolumeSnapshot
apiGroup: snapshot.storage.k8s.io
```

2. Create the Persistent Volume Claim.

```
oc create -f csi-dynamic-snapshot-restore.yaml
```

To restore a raw block volume snapshot to a new PVC

- Add `volumeMode: Block` to `csi-static-snapshot-restore.yaml` or `csi-dynamic-snapshot-restore.yaml` as under:

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <csi-static-snapshot-restore/csi-dynamic-snapshot-restore>
spec:
  volumeMode: Block
  storageClassName: csi-infoscale-sc
  dataSource:
    name: csi-static-snapshot
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
    - ReadWriteMany
  # Please provide storage capacity exactly matching to the dataSource
  resources:
    requests:
      storage: 10Gi
```

Deleting a volume snapshot

You can delete one or more snapshots by deleting the volume snapshot object associated with snapshot. If you set the `DeletionPolicy` to Delete while defining the snapshot object, then the underlying storage snapshot is automatically deleted when the `VolumeSnapshotContent` object is deleted. Use the following command to delete the `VolumeSnapshot` object:

```
oc delete volumesnapshot csi-dynamic-snapshot
```

Note: For space-optimized snapshots, InfoScale maintains the association between the source PVC and the snapshot volume. Therefore, you must delete the snapshot objects before deleting the source PVC. For full-instant snapshots, you can delete the source PVC before deleting the snapshot object after the synchronization between these two is completed. Review CSI controller logs for details of events or observed errors.

Creating snapshot of a raw block volume

Add `sourceVolumeMode: Block` to `csi-static-snapshot-content.yaml` as under:

```
---
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: csi-static-snapshot-content
spec:
  deletionPolicy: Retain
  driver: org.veritas.infoscale
  source:
    # Provide pre-provisioned Infoscale snapshot volume name
    snapshotHandle: clus_<clusterid>/<dgname>/<volname>
  volumeSnapshotRef:
    name: csi-static-snapshot
    namespace: default
  sourceVolumeMode: Block
```

Managing InfoScale volume snapshots with Velero

Velero is a backup and recovery solution that assists in backing up and restoring the applications and their corresponding persistent volumes in an OpenShift or Kubernetes environment.

You can integrate InfoScale CSI plugin with Velero to backup and restore CSI-backed volumes across clusters. The following example shows how to configure and use Velero with InfoScale CSI plugin snapshots feature. This example uses MinIO object storage server for storing objects metadata.

Setting up Velero with InfoScale CSI

Prerequisite:

Download and install Velero CLI. For more information, see Velero documentation.

Perform these steps to configure Velero:

1. Set up the InfoScale CSI environment. See "CSI plugin deployment" on page 142.
2. Set up the `MinIO` server. The `00-minio-deployment.yaml` file to set up the `MinIO` server is included in the Velero package. You must edit the IP addresses and ports in the yaml as required.

```
oc apply -f 00-minio-deployment.yaml
```

3. Create the Velero secret file with the credentials to access the MinIO server.

```
[default]
aws_access_key_id=<user_id>
aws_secret_access_key=<passowrd>
```

4. Install Velero by running the following command:

```
velero install \
--provider aws \
--features=EnableCSI \
--plugins=velero/velero-plugin-for-csi:v0.1.0,
velero/velero-plugin-for-aws:v1.0.0 \
--bucket velero \
--secret-file ./credentials-velero \
--use-volume-snapshots=True \
--backup-location-config region=minio,s3ForcePathStyle="true",
s3Url=http://minio.velero.svc:9000,publicUrl=http://<ip>:<port> \
--snapshot-location-config region=default,profile=default
```

5. Deploy the application that uses the CSI backed InfoScale volumes.
6. Create a `VolumeSnapshotClass` for the CSI backed volumes using the `csi-infoscale-snapclass yaml`.

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshotClass
metadata:
  name: csi-infoscale-snapclass
  annotations:
    snapshot.storage.kubernetes.io/is-default-class: "false"
driver: org.veritas.infoscale
deletionPolicy: Retain
parameters:
  snapType: full-instant
```

Run the following command to create a VolumeSnapshotClass:

```
oc create -f csi-infoscale-snapclass.yaml
```

7. After `VolumeSnapshotClass` is created, set `velero.io/csi-volumesnapshot-class:`, set to `"true"`.

Velero chooses this to back up InfoScale PersistentVolumeClaims.

Taking the Velero backup

After you configure the Velero setup, you can back up all objects in your cluster, or you can filter objects by type, namespace, and label. For more information, see Velero documentation.

Use the velero backup create command to back up applications that are using the CSI volumes.

For example, to back up a namespace run the following command:

```
# velero backup create <backup_name>
--include-namespaces=<namespace_name> -wait
```

Note: When you back up by using Velero, the PVCs of the CSI Volumes are backed up as snapshots on the on-premises InfoScale host.

Creating a schedule for a backup

The schedule operation allows you to back up your data at specified periodic intervals. The first backup is performed when the schedule is created, and subsequent backups happen at the scheduled interval.

Scheduled backups are saved with the name <SCHEDULE NAME>-<TIMESTAMP>, where <TIMESTAMP> is formatted as YYYYMMDDhhmmss.

In an OpenShift or Kubernetes environment, the scheduled backup operation create snapshots of the CSI-backed volumes on a pre-defined time interval.

For example, use the following sample `yaml` file to create backup schedules of the nginx-app namespace after every 30 minutes that has a validity of 2 hours.

```
apiVersion: velero.io/v1
kind: Schedule
metadata:
  name: daily
  namespace: velero
spec:
  schedule: "*/30 * * * *"
  template:
    hooks: {}
    includedNamespaces:
    -   nginx-app
    ttl: 02h00m0s
```

Restoring from the Velero backup

The restore operation allows you to restore all objects and persistent volumes from a previously created backup. You can also restore only a subset of objects and persistent volumes. For more information, see Velero documentation.

The default name of a restore is `<BACKUP NAME>-<TIMESTAMP>`, where `<TIMESTAMP>` is formatted as `YYYYMMDDhhmmss`. You can also specify a custom name.

Use the velero restore create command to restore the OpenShift or Kubernetes objects and InfoScale CSI volumes from the previously created backup.

For example:

```
velero restore create <restore-name>  --from-backup <backup-name>
```

Volume cloning

The CSI volume clone feature duplicates an existing Persistent Volume at given point in time. Cloning creates an exact duplicate of the specified volume on the backend rather than creating a new empty volume. When a clone is created, it is an independent object that can be used as any other PVC. The data of the cloned volume is also in sync with the data of the original dataSource PVC. The cloned volume can be consumed, cloned, snapshotted, or deleted without affecting the original dataSource PVC.

You can clone a PVC only when the following conditions are met:

- The source and destination PVCs are in the same namespace.
- The source and destination Storage Class are the same.

Creating volume clones

The cloning feature enables you to specify an existing PVC as a `dataSource` while creating a new PVC. Prerequisites are as under:

- The source PVC is bound and available for use
- A valid Storage Class is available
- The source PVC is created using the InfoScale CSI driver that supports volume cloning

To clone a PVC from an existing PVC:

1 Identify the PVC that you want to clone.

```
oc get pvc
```

2 Define the PersistentVolumeClaim object using the yaml and specify the name of the PVC object to use as source:

```
csi-dynamic-volume-clone.yaml
---
  kind: PersistentVolumeClaim
  apiVersion: v1
  metadata:
    name: csi-infoscale-volume-clone
  spec:
    storageClassName: csi-infoscale-sc
    accessModes:
      - ReadWriteMany
    resources:
      requests:
        storage: 5Gi
    dataSource:
      kind: PersistentVolumeClaim
      name: csi-infoscale-pvc
```

3 Create a volume clone:

```
oc create -f csi-dynamic-volume-clone.yaml
```

On successful creation of a clone, it is pre-populated with the data from the specified PVC `dataSource` volume.

Deleting a volume clone

To delete a volume clone, use the following command:

```
oc delete pvc csi-infoscale-volume-clone
```

Verify that the volume clone is deleted and not displayed in the command output.

Using InfoScale with non-root containers

While using InfoScale with containers that are not running as the root user, the storage ownership might need to be changed to ensure that the containers are able to read or write to the file system. You can specify an `fsGroup` attribute in the pod security context to enable read or write. Using the `fsGroup` attribute instructs OpenShift or Kubernetes to change the ownership of the file system to the specified group. It also instructs runtime to add the specified group to the supplemental groups the container is run with. This ensures that the container processes are able to read and write files in the volume. In the following example `securityContext` includes an explicit `fsGroup`

```
securityContext:
     runAsUser: 1000
     runAsGroup: 3000
     fsGroup: 5000
     fsGroupChangePolicy: "OnRootMismatch"
```

Using InfoScale in SELinux environments

If InfoScale CSI is used to provision volumes in an environment where SELinux is enabled in enforcing mode, the pod definition must explicitly specify a SELinux label. Files in the provisioned volume are then re-labeled and the containers associated with the pod are started in the appropriate SELinux context.

For example, the following `securityContext` includes explicit SELinux options:

```
securityContext:
     runAsUser: 1000
     runAsGroup: 3000
     fsGroup: 5000
     fsGroupChangePolicy: "OnRootMismatch"
     seLinuxOptions:
       level: "s0:c447,c946"
```

To avoid weakening security posture, ensure that you do not reuse the same label for pods that are not expected to access the same volume. Without explicit labels specified, pods may lose access to previously created files, or files that were created from a different node, for the case of `ReadWriteMany` volumes.

CSI Drivers

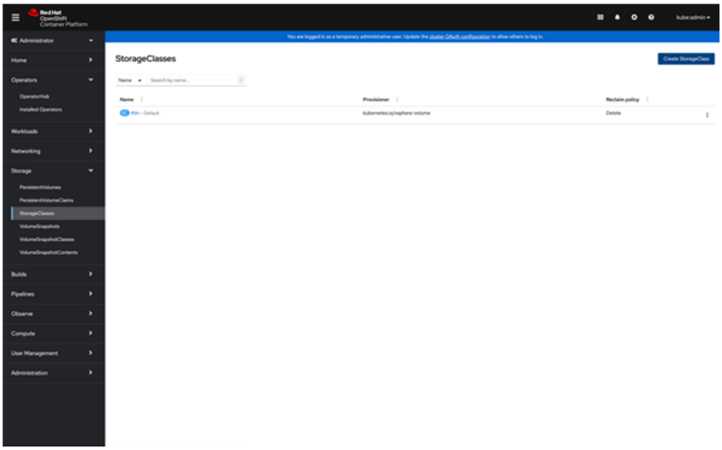
InfoScale CSI Driver is a Production Driver. See Kubernetes Drivers for a complete list of Production Drivers.

Creating CSI Objects for OpenShift

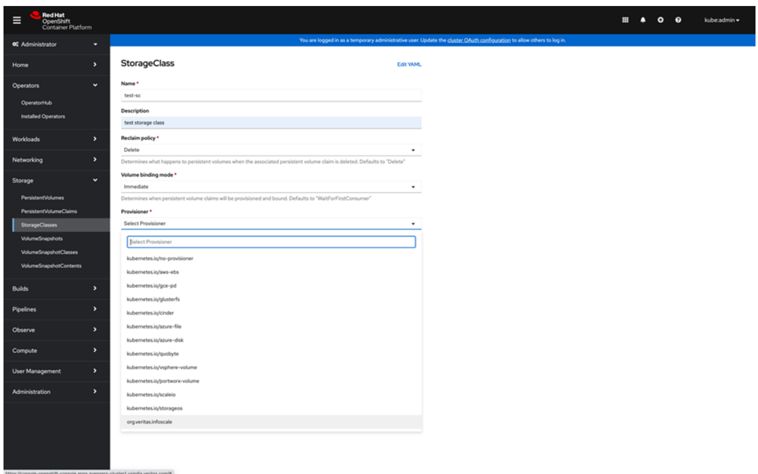
Complete the following steps to install InfoScale operator .

Creating StorageClass

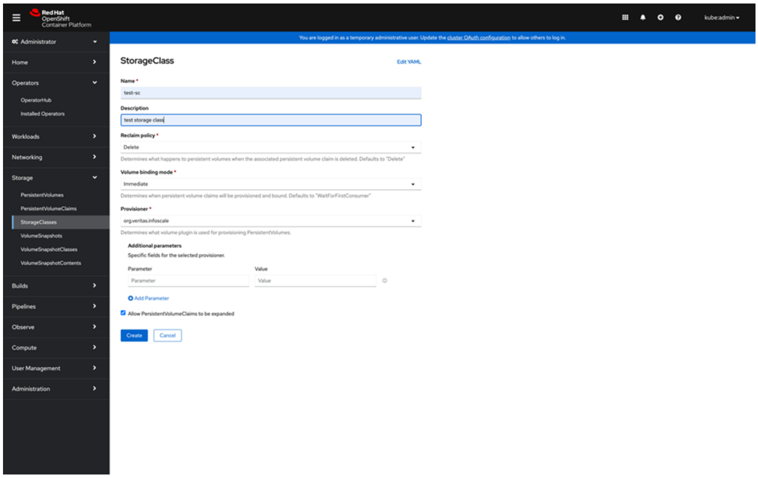
1. Connect to the OpenShift console and access the Catalog menu.
2. In the left frame, click Storage > StorageClasses. Click Create StorageClass in the upper-right corner of the screen.



3. Assign a Name to the StorageClass in the following screen. Optionally, you can enter Description.



4. Select Immediate as the Volume binding mode and org.veritas.infoscale as the Provisioner.
5. Click Add Parameter in Additional Parameters.



6. For fstype as the Parameter, enter vxfs as its Value.
7. Optionally, you can enter the following parameters.

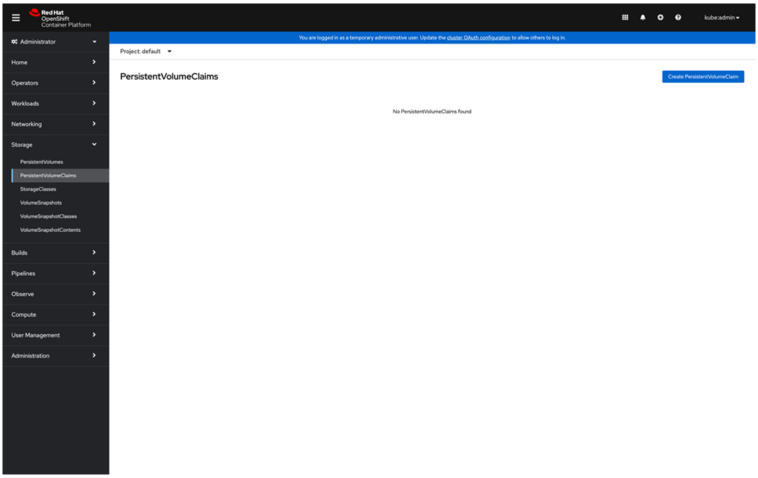
Parameter	Value
layout	Enter one of the following - stripe, mirror, stripe-mirror, mirror-stripe, concat, concat-mirror, and mirror-concat. By default, a best-suited layout is selected based on the environment
faultTolerance	Number of disk or host failures a storage object can tolerate.
nstripe	Number of stripe columns to use when creating a striped volume
stripeUnit	Stripe unit size to use for striped volume
mediaType	Type of disks to be used for Volume creation (HDD or SSD).

8. Click Create. Wait for the StorageClass to be created.
9. Review all details for the StorageClass.

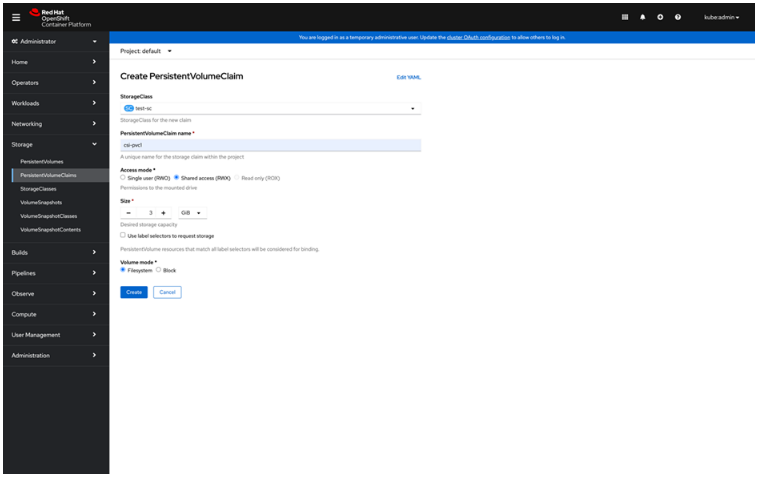
Now you can create a PersistentVolumeClaim.

Creating PersistentVolumeClaims

1 In the left frame, click Storage > PersistentVolumeClaims. Click Create PersistentVolumeClaim in the upper-right corner of the screen.



2 Select the StorageClass you just created.



3 Enter the following details

Parameter	Value
PersistentVolumeClaim	Assign a name.
Access Mode	Choose Single User (RWO) or Shared Access (RWX).
Size	Select the Storage capacity you want to assign to this Volume.
Volume mode	Choose Filesystem or Block.

4 Click Create. Wait for its successful creation.

5 Review information on the screen that is displayed.

After installing InfoScale and creating this PersistentVolumeClaim, Persistent Storage is now available for the Applications.

You can similarly create other storage objects or clones from the OpenShift web console.

Creating ephemeral volumes

Some applications like caching services do not require persistent storage. Ephemeral volumes are temporary volumes which such applications can utilize. You can use InfoScale CSI drivers to create generic ephemeral volumes.

Ephemeral volumes can be created in block mode also by setting volumeMode to 'Block'. Snapshotting, cloning, and resizing are also supported. See the sections above to know more.

Complete the following steps to create ephemeral volumes by using an InfoScale CSI driver storage class.

1. Copy the following content and save as `ephemeral_volume_pod.yaml`.

```
kind: Pod
apiVersion: v1
metadata:
  name: test-pod
spec:
  containers:
    - name: test-pod-container
```

```
    image: busybox
    volumeMounts:
    - mountPath: "/ephemeral"
      name: ephemeral-volume
    command: [ "sleep", "1000000" ]
volumes:
  - name: ephemeral-volume
    ephemeral:
      volumeClaimTemplate:
        metadata:
          labels:
            type: test-pod-volume
        spec:
          accessModes: [ "ReadWriteOnce" ]
          storageClassName: "vxvmsc-mirror"
          resources:
            requests:
              storage: 1Gi
```

2. Run the following command to create an application pod with the ephemeral volume.

   ```
   kubectl/oc apply -f ephemeral_volume_pod.yaml
   ```

3. Run the following command to verify.

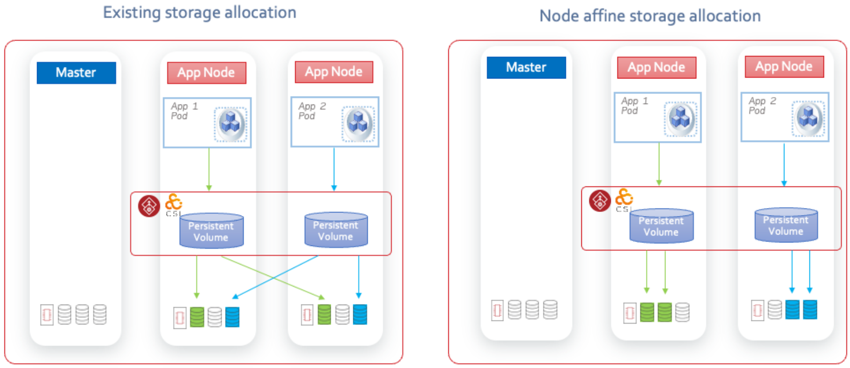
   ```
   kubectl/oc get pvc
   ```

Creating node affine volumes

Overview

Node affine volumes are a specialized type of striped only (no mirrors) volume that allocate storage exclusively to a specific node, ensuring that all I/O operations are served from local storage. This eliminates the need for network-based storage access. This approach is particularly effective for OLAP applications that require high-performance file storage, where traditional solutions like NFS or iSCSI may not perform well in such use-cases.

VIKE, as a hyper-converged solution, leverages node affine volumes to storage allocation, thereby achieving high performance through local I/O and efficient striping across multiple disks.



Enabling node affinity

To enable node affinity, specify the nodeAffinity parameter in the storage class. This parameter can be set to either:

- true – Enables node affinity, with node selection based on the volume binding mode.
- Node name – Specifies the Kubernetes node to which the volume should be affined.

Volume binding modes

The node selection for storage allocation through node affinity depends on the volume binding mode configured in the storage class:

1. Immediate binding
   - Storage is allocated as soon as the PersistentVolumeClaim (PVC) is created, potentially before the application pod is created.
   - The InfoScale CSI driver selects the node for storage allocation based on the policy specified in the storage class.
   - Once the volume is created, the appropriate nodeAffinity is set in the PersistentVolume (PV) object, ensuring that Kubernetes schedules the application pod on the same node.
2. Delayed binding (wait for first consumer)
   - Storage allocation is delayed until the application pod is scheduled.
   - Kubernetes first selects the node for the application pod and then passes the node information to the InfoScale CSI driver for storage allocation on that node.

- If the selected node does not have enough storage, the driver notifies the Kubernetes scheduler to reschedule the pod on a different node.

Supported allocation policies

Specify the policy using the nodeAffinityType parameter. The available options are:

- bestspread (default) – Selects the node with the maximum available storage among the nodes that can fulfill the request. This policy evenly spreads application pods but may result in suboptimal storage utilization.
- bestfit – Selects the node with the least available storage that can still satisfy the request. This policy optimizes storage utilization but may lead to uneven pod distribution.

Configuration parameters

The following parameters are used to configure node affinity:

Table 9-1 Configuration parameters for node affinity

Parameter	Description	Default value
nodeAffinity	Set to 'true' or the name of the Kubernetes node to which the volume should be affined.	N/A
nodeAffinityType	Set to 'bestspread' or 'bestfit'.	'bestspread'

Chapter 10

Installing and configuring InfoScale DR Manager on OpenShift

This chapter includes the following topics:

Introduction

This section informs you how to install Custom Resource (CR) files related to Disaster Recovery (DR).

Note: Multiple InfoScale clusters in different namespaces can be configured within a single OpenShift cluster. However, DR configuration is supported for a single InfoScale cluster per OpenShift cluster only.

Prerequisites

1. InfoScale must be installed and InfoScale pods must be configured on the clusters.
2. Load balancer must be installed and configured. If you choose to install Metallb as the load balancer, steps are listed in `Installing InfoScale DR`.
3. Be ready with the following information for every member cluster
   - GlobalClusterMembership: Virtual IP address and a port for each member cluster in GCM. This information is captured in GCM custom resource.
   - Data Replication: Virtual IP address, port, netmask and network interface (NIC) required for each cluster to be specified in DataReplication CR. This information is exclusive for a DataReplication CR and is used to configure InfoScale Volume Replicator.
4. Ensure that stale Custom Resources (CR) and Custom Resource Definitions (CRD) related to DR do not exist on the clusters.
5. Optionally if you want to configure DNS resource, DNS key and DNS private key. To know how to obtain keys, see https://www.veritas.com/content/support/en_US/doc/129694359-129694362-0/uxrt-731_id-SF1J0175244-129694362.
6. Multi-homed Bare-Metal Nodes: You must strictly define the interfaces array in the MetalLB L2Advertisement object. Limit the advertisement only to the interface connected to the OpenShift CNI bridge (usually br-ex or eno1).

```
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: vike-vvr-l2-adv
  namespace: metallb-system
spec:
  ipAddressPools:

<vvr-vip-pool-name>
interfaces:

br-ex    # Must match the OVN-Kubernetes bridge or primary CNI interface
which the traffic would leave.
```

Creating Persistent Volume for metadata backup

Ensure that you run the following steps on the primary and secondary clusters to create a PVC for backing up and restoring application metadata across clusters.

- Create a Storage Class by running the following `sc.yaml`.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: infoscale-dr-csi-sc
parameters:
  fstype: vxfs
  layout: mirror
allowVolumeExpansion: true
provisioner: org.veritas.infoscale
reclaimPolicy: Delete
```

- Create a Persistent Volume Claim (PVC) by running the following `pvc.yaml`.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: infoscale-dr-meta-bkp-pvc
  namespace: infoscale-vtas
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 4Gi
  storageClassName: infoscale-dr-csi-sc
  volumeMode: Filesystem
```

External dependencies

- Load balancer service must be installed on all clusters that are a part of Disaster Recovery (DR) to allocate Virtual IP addresses for cluster-to-cluster communications. Virtual IP addresses ensure a resilient communication between the clusters. See Installing Metallb to install Metallb. Alternatively, you can install any other load balancer service. Refer to its documentation. If you are installing in a cloud environment, ensure that you use native load balancers.

Installing InfoScale DR Manager by using OLM

You can connect to the Red Hat portal by using web console or the Command Line Interface (CLI).

Installing InfoScale DR Manager by using web console

Complete the following steps:

On the primary cluster and the secondary cluster

1. Install InfoScale DR Manager.

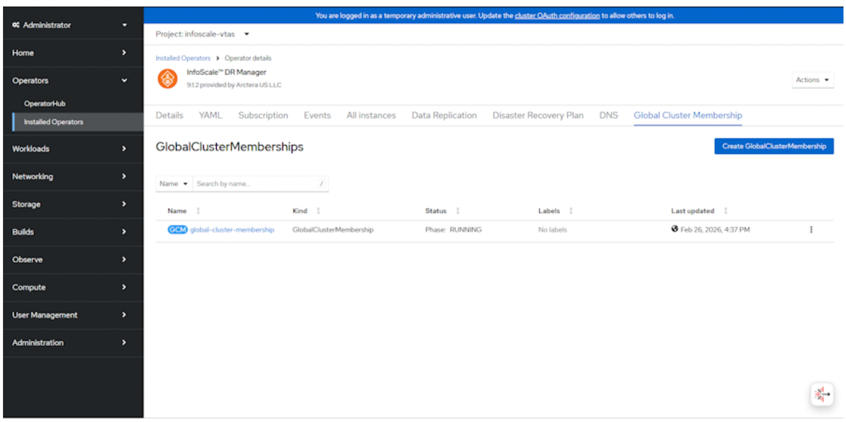
   See " Installing from OperatorHub by using web console" on page 38.

2. After the InfoScale DR Manager is installed, you can configure Global Cluster Membership, Data Replication, and Disaster Recovery Plan.

Configuring InfoScale DR Manager by using web console

Complete the following steps:

1. Click Operators > Installed Operators. Select InfoScale DR Manager.
2. In the main menu, click Global Cluster Membership. You can create a cluster membership.
3. Click Create GlobalClusterMembership in the upper-right corner of the screen as under.



4. Assign a Name and click Global Member Clusters. You can enter cluster details here.

5 Enter Cluster ID of the primary cluster first and its IP address in DR Controller Address. Optionally, you can enter its port number in DR Controller Port.

Note: The IP address and port number are generated by using Load Balancer. An example of the command to generate the IP address is - `oc -n infoscale-vtas expose deployment infoscale-dr-manager --name dr-lb-service --type LoadBalancer --protocol TCP --port 14155 --target-port 14155`

The default port number for Load Balancer service in this example is 14155.

6 Enter Cluster ID of the secondary cluster and its IP address in DR Controller Address. Optionally, you can enter its port number in DR Controller Port.

Note: This is the IP address and port number of the peer cluster generated by using Load Balancer. Use the command mentioned in the above step.

7 Enter the Cluster ID of the primary cluster in Local Cluster Name.

8 Click Create. Wait till it is created. The newly created cluster is listed under Global Cluster Membership.

9 Repeat steps 1 to 8 on the secondary cluster. In step 7, ensure that you enter the cluster ID of the secondary cluster. After a successful configuration on the secondary cluster, the secondary cluster is the DR cluster.

10 Run the following command on the primary cluster to verify whether GCM is successfully configured - `oc get gcm`

An output similar to the following indicates a successful configuration.

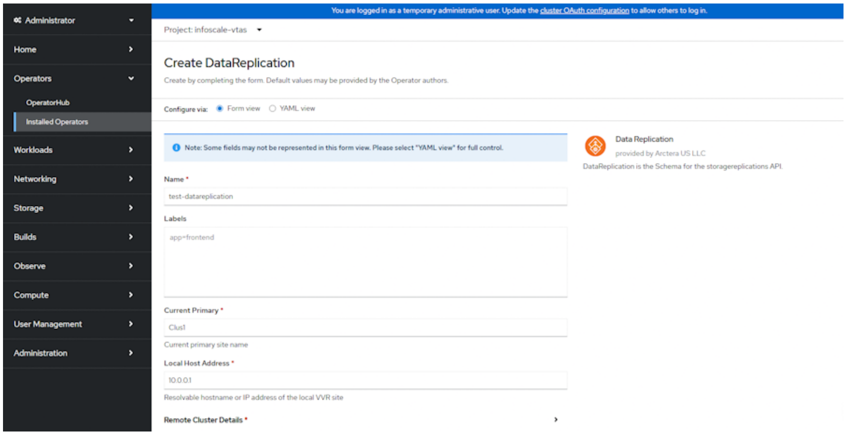
```
NAME                       CLUSTER NAME   CLUSTER STATE   PROTOCOL
global-cluster-membership  Clus1          RUNNING         10



PEER LINK STATE
{"Clus1":"CONNECTED","Clus2":"CONNECTED"}
```

11 You can now configure Data Replication. Select the appropriate project and click Data Replication.

12 Click Create DataReplication in the upper-right corner of the screen. The following screen opens.



13 For the primary cluster, enter the Cluster Name, its IP address (Can be any Virtual IP address available in the same subnet) in Local Host Address, and its corresponding Net Mask in Local Net Mask.

14 For the secondary cluster, enter its name in Cluster Name, its IP address (Can be any Virtual IP address available in the same subnet) in Remote Host Address, and its corresponding Net Mask in Remote Netmask. Enter the network interface in Remote NIC.

Note the following:

- When configuring Data Replication, the network interface to be used for the Remote NIC and the Local NIC should be the one where the worker node IPs are assigned.
- If the OCP cluster is set up with the OVNKubernetes Network Plugin, the worker node IP is typically assigned to the br-ex network interface. Therefore, during the data replication configuration, you should enter "br-ex" as the network interface for the Remote NIC and the Local NIC.

15 Enter the Namespace for which you want to configure data replication.

16 In Local NIC, enter the network interface of the primary cluster.

17 Click Create. Wait till Data Replication is configured and gets listed.

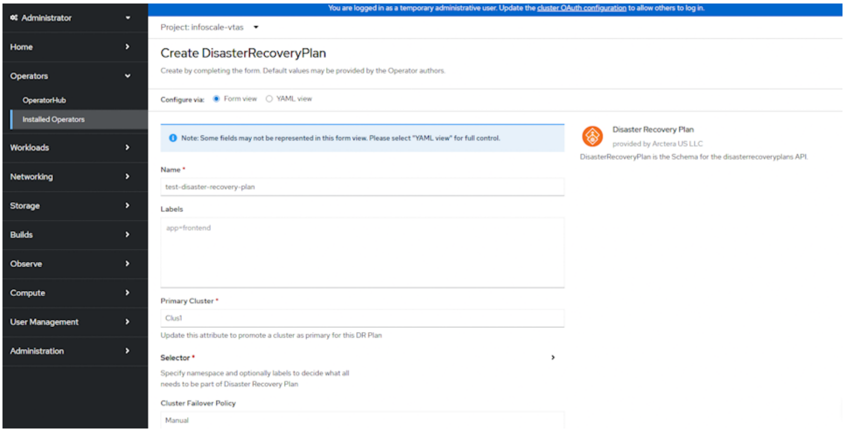
18 Run the following command on the primary cluster to verify whether Data Replication is successfully configured - `oc get datarep -o wide`

19 Review output for

```
consistent,up-to-date
```

20 Click Disaster Recovery Plan in the main menu to create a plan.

21 Click Create DisasterRecoveryPlan in the upper-right corner of the screen. The following screen opens.



22 Assign a Name to this plan.

Note: The names of clusters and the Data replication plan appear here.

23 Review the Primary Cluster. Enter the Namespace that you want to be a part of this plan.

24 Review the Data Replication Pointer and Preferred Cluster List.

25 Click Create and wait till the Disaster Recovery Plan is created and listed.

After these successful configurations, Disaster Recovery (DR) is ready.

Installing from OperatorHub by using Command Line Interface (CLI)

After you download `infoscale-yamls-v9.1.2.tar.gz` from InfoScale Download Center and unzip and untar it, a folder `/infoscale-yamls-v9.1.2/openshift/olm/` is created and all files required for installation are available in the folder. An OpenShift cluster already has a namespace `openshift-operators`. You can choose to install InfoScale DR Manager in `openshift-operators`.

Complete the following steps on the primary and the secondary clusters.

1. Copy the following content to a file and save the file as `infoscale-dr-sub.yaml`.

```
---
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: infoscale-dr-manager-sub
  # Namespace should be either infoscale-vtas
  # or openshift-operators
  namespace: infoscale-vtas
spec:
  channel: fast
  name: infoscale-dr-manager
  installPlanApproval: Manual
  # Source can be infoscale-dr-manager-catalog
  # if infoscale-dr-catsrc.yaml is used
  source: certified-operators
  sourceNamespace: openshift-marketplace
  startingCSV: infoscale-dr-manager.v9.1.2
```

2. Run the following command on the bastion node to create a namespace infoscale-vtas. Ignore if you want to install in openshift-operators.

   `oc create namespace infoscale-vtas`

   Review output similar to the following to verify whether the namespace is created successfully.

```
namespace/infoscale-vtas created
```

3 Run the following command on the bastion node to create a subscription:

Note: If you want to install InfoScale DR Manager in `openshift-operators` namespace, edit `infoscale-dr-sub.yaml`. Change namespace from `infoscale-vtas` to `openshift-operators`. To install the latest bundle, modify startingCSV to infoscale-dr-manager.v9.1.2.

```
oc create -f infoscale-dr-sub.yaml
```

The following output indicates a successful command run:

```
subscription.operators.coreos.com/infoscale-dr-manager-sub created
```

4 Run the following command on the bastion node:

```
oc get subscriptions -n infoscale-vtas
```

The following output indicates a successful command run:

```
NAME                                PACKAGE
infoscale-dr-manager                infoscale-dr-manager
infoscale-licensing-operator        infoscale-licensing-operator
infoscale-operator                  infoscale-operator
nfd-stable-redhat-operators
             openshift-marketplace  nfd

SOURCE                               CHANNEL
infoscale-dr-manager-catalog         fast
infoscale-lic-operator-test-catsrc   fast
infoscale-sds-operator-catalog       fast
redhat-operators                     fast
```

5 Run the following command on the bastion node.

Change `infoscale-vtas` to `openshift-operators`, if you are using that namespace.

```
oc get installplans --namespace infoscale-vtas
```

Use `<installation-name>` from the output similar to the following output:

```
NAME                  CSV                              APPROVAL   APPROVED
<installation-name>   infoscale-dr-manager.v9.1.2   Manual     false
```

6 Run the following command on the bastion node:

```
oc patch installplan <installation-name> --namespace
infoscale-vtas --type merge --patch '{"spec":{"approved":true}}'
```

The following output indicates a successful command run:

```
installplan.operators.coreos.com/<installation-name> patched
```

7 Run the following command on the bastion node:

```
oc get installplans --namespace infoscale-vtas
```

Review output similar to the following. Check if APPROVED is true.

```
NAME                CSV                     APPROVAL APPROVED
<installation-name> infoscale-dr-manager.  Manual   True
                                  v9.1.2
```

8 Run the following command on the bastion node to check the status of csv.

```
oc get csv
```

Components which are being installed or are pending installation are listed. Run the command again till you see the phase for all components as 'Succeeded'

```
NAME                                  DISPLAY
cert-manager-operator.v1.18.0         cert-manager Operator for Red Hat O
infoscale-dr-manager.v9.1.2           InfoScale™ DR Manager
infoscale-licensing-operator.v9.1.2   InfoScale™ Licensing Operator
infoscale-sds-operator.v9.1.2         InfoScale™ SDS Operator
```

```
VERSION  REPLACES                               PHASE
1.18.0   cert-manager-operator.v1.17.0          Succeeded
9.1.2    infoscale-dr-manager.v8.0.410          Succeeded
9.1.2    infoscale-licensing-operator.v8.0.410  Succeeded
9.1.2    infoscale-sds-operator.v8.0.410        Succeeded
```

9 After InfoScale-dr-manager CSV is in a 'Succeeded' phase, proceed with the custom resource configuration and creation.

10 Perform the steps that are listed in the following topics:

See “Configuring Global Cluster Membership (GCM)” on page 193.

See “Configuring Data Replication” on page 197.

See “Configuring DNS” on page 202.

See “Configuring Disaster Recovery Plan” on page 208.

Installing InfoScale DR Manager by using YAML

This section informs you how to install and configure Disaster Recovery for your InfoScale cluster by using YAML.

Note: When you download, unzip, and untar `infoscale-yamls-v9.1.2.tar.gz` , all files required for installation are available.

Complete the following steps to install the InfoScale DR Manager on the source and the target DR cluster.

Creating application user role

1. Run the following command on the bastion node:

   ```
   oc apply -f infoscale-yamls-v9.1.2/DR/infoscale-dr-admin-role.yaml
   ```

2. Copy the following content to `infoscale-yamls-v9.1.2/DR/ClusterRoleBinding .yaml`.

   ```
   ---
   apiVersion: rbac.authorization.k8s.io/v1
   kind: ClusterRoleBinding
   metadata:
      name: infoscaledr-role-binding
   roleRef:
      apiGroup: rbac.authorization.k8s.io
      kind: ClusterRole
      name: infoscaledr-admin-role
   subjects:
   -  kind: User
      name: postgres-admin
      apiGroup: rbac.authorization.k8s.io
   ```

   This is an example for a ClusterRoleBinding role.

3. Run the following command on the bastion node:

   ```
   oc apply -f infoscale-yamls-v9.1.2/DR/ClusterRoleBinding.yaml
   ```

For DR controller installation and Global Cluster Membership (GCM) configuration, you must use the kubeadmin role only. To configure DR plan and data replication, you can use this role or the kubeadmin role.

1. Run the following command on the bastion node of each cluster:

   ```
   oc apply -f
   /infoscale-yamls-v9.1.2/DR/OpenShift/dro_deployment.yaml
   ```

2. Wait till the command execution is complete.

3 Run the following command on the bastion node to verify if the deployment is successful:

```
oc -n infoscale-vtas get pods
```

See the Status in the output similar to the following:

```
NAME                          READY   STATUS     RESTARTS    AGE
infoscale-dr-manager-xxxx     1/1     Running    0           114m
```

Status must change from `ContainerCreating` to `Running`.

4 Run the following commands to configure ExternalIP for DR pod.

Note: Run these steps only if you want Metallb as the load balancer. If you choose any other load balancer, refer to its documentation for installation and configuration.

```
oc -n infoscale-vtas expose deployment infoscale-dr-manager --name
my-lb-service --type LoadBalancer --protocol TCP --port 14155
--target-port 14155
```

Here, DR controller uses port 14155 internally to communicate across peer clusters.

5 After a successful installation and configuration, you can verify by running the following command:

```
oc get svc my-lb-service
```

An output similar to the following indicates that installation and configuration is successful:

```
NAME          TYPE        CLUSTER-IP   EXTERNAL-IP  PORT(S)          AGE
my-lb-service LoadBalancer<IP address> <IP address> 14155:14155/TCP 13h
```

Run this command on both the clusters and verify if installation and configuration is successful. Verify whether `EXTERNAL-IP` is accessible from one cluster to the other cluster.

Configuring Global Cluster Membership (GCM)

With Global Cluster Membership (GCM), you can define membership of clusters for disaster recovery. The GCM CR must be configured and applied on all clusters. When configured, the Global Cluster Membership forms a logical notion called 'Global

Cluster' with all underlying clusters as 'Member Clusters'. Member clusters are OpenShift clusters providing disaster recovery capabilities to application components.

To provide DR, these member clusters:

- Send heartbeats with each other periodically.
- Exchange information like state, configuration, operation.
- Perform/participate in operation like migration.

Complete the following steps:

1. Edit `/infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml` as under:

```
apiVersion: infoscale.veritas.com/v1
kind: GlobalClusterMembership
metadata:
  name: global-cluster-membership
spec:
  # Local cluster name in the global membership
  localClusterName: <Local cluster
                           where you want to apply this YAML>
  globalMemberClusters:
      # Cluster ID of each member of global cluster membership
    - clusterID: <A unique ID of the primary cluster>
      # Address Used For Communicating With Peer Cluster's DR Controller
      drControllerAddress: "<Load balancer IP address or haproxy
                                           of the local cluster>"
      # Port used for DR controller
      drControllerPort: "<Load balancer port number>"
    - clusterID: <A unique ID of the secondary cluster>
      drControllerAddress: "<Load balancer IP address or haproxy
                                            of the DR site>"
      drControllerPort: "<Load balancer port number>"
  # If heartbeat with peer cluster missed more than CounterMissTolerance
         times,then cluster will be moved to FAULTED state
  counterMissTolerance: 5
  globalClusterOperation: "none"
  # Application metadata backup sync frequency to DR site(s) in minutes
  metadataBackupInterval: 15
  # Refresh data replication status after specified minutes
  datarepRefreshStatusFrequency: 10
  # Include cluster-scoped Custom Resource Definitions (CRDs)
  #                           in disaster recovery plan backup
```

```
  backupClusterScopeCRD: true
  # Maximum metadata backup copies stored per DR plan
  maximumMetadataCopies: 5
```

Note: Do not enclose the parameter values in angle brackets(< >) . For example, if 8334 is the Load balancer port number; enter drControllerPort: "8334" for drControllerPort: "<Load balancer port number>". localClusterName and clusterID can have maximum 20 characters.

---

2. Run the following command on the bastion node of the source cluster:

   ```
   oc apply -f
   /infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml
   ```

3. Edit another instance of `/infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml` to add DR site as under:

   ```
   apiVersion: infoscale.veritas.com/v1
   kind: GlobalClusterMembership
   metadata:
     name: global-cluster-membership
   spec:
     # Local cluster name in the global membership
   localClusterName: <DR site cluster name
                          where you want to apply this YAML>

     globalMemberClusters:
     # Cluster ID of each member of global cluster membership
      - clusterID: <A unique ID of the primary cluster>
     # Address Used For Communicating With Peer Cluster's DR Controller
       drControllerAddress: "<Load balancer IP address or haproxy
                                     of the local cluster>"

     # Port used for DR controller
        drControllerPort: "<Load balancer port number>"
         - clusterID: <A unique ID of the secondary cluster>
        drControllerAddress: "<Load balancer IP address or haproxy
        of the DR site>"
         drControllerPort: "<Load balancer port number>"
     # If heartbeat with peer cluster missed more than CounterMissTolerance
     # times, then cluster will be moved to FAULTED state
   ```

```
  counterMissTolerance: 5
  globalClusterOperation: "none"
  # Application metadata backup sync frequency to DR site(s) in minutes
  metadataBackupInterval: 15
  # Refresh data replication status after specified minutes
  datarepRefreshStatusFrequency: 10
  # Include cluster-scoped Custom Resource Definitions (CRDs) in
  # disaster recovery plan backup
  backupClusterScopeCRD: true
  # Maximum metadata backup copies stored per DR plan
  maximumMetadataCopies: 5
```

4. Copy this file to the DR site and Run the following command again on the bastion node of the DR site:

   ```
   oc apply -f
   /infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml
   ```

5. Manually verify on all clusters whether the GLOBALCLUSTERSTATE is DISCOVER_WAIT by running `oc get gcm`.

   Various states are as follows:

State	Description
UNKNOWN	A transient default Global-Cluster state. After initial configuration/setup, cluster state must transition to DISCOVER_WAIT. Prolonged UNKNOWN state indicates errors in initial configuration/setup. Review DR Controller log for the ongoing activities.
DISCOVER_WAIT	Although local cluster has a copy of GCM and member cluster details, it is not certain whether local copy of GCM and member cluster is up-to-date. If GCM and member cluster details are identical on all peer clusters then all clusters automatically transition to RUNNING state. If the details are not identical, waits till you seed the cluster by updating GlobalClusterOperation to localbuild. When a member cluster transitions to RUNNING state, all peer clusters with identical membership transition to RUNNING state.
ADMIN_WAIT	If local membership definition does not match with peer cluster's membership definition, clusters transition to this state. Update membership on peer clusters and ensure that it is identical. Peer clusters then transition to RUNNING state.

State	Description
RUNNING	Cluster transitions to RUNNING state if you seed cluster membership by updating GlobalClusterOperation to localbuild. Cluster transitions to RUNNING state even when local copy of membership matches with peer clusters.
EXITING	You have initiated DR Controller stop.
EXITED	DR Controller stopped.

6. To verify whether the Global Cluster is successfully created, run the following command on the bastion node:

   ```
   oc get gcm
   ```

7. Review the cluster names, GlobalClusterState, and PeerLinkState in the output similar to the following. GlobalClusterState must be Running and PeerLinkState must be Connected.

   ```
   NAME                      CLUSTER NAME   CLUSTER STATE   PROTOCOL
   global-cluster-membership Clus1          RUNNING         10


   PEER LINK STATE
   {"Clus1":"CONNECTED","Clus2":"CONNECTED"}
   ```

   Here, `NAME` is the `Name of GlobalClusterMembership custom resource`, `CLUSTER NAME` is the `Local Cluster ID`, and `Clus1`, `Clus2` are the Cluster IDs that you defined for global membership.

Configuring Data Replication

Using Data Replication custom resource you can configure replication for persistent data(PVs and PVCs) associated with application components in a namespace. Custom resource created on a cluster is automatically synchronized on all peer clusters. Hence, this CR needs to be configured on the primary cluster only. After CR is configured, replication is set up. InfoScale Volume Replicator(VVR) is responsible for performing replication. You can check status of underlying replication and perform operations like stop, pause, resume, and migrate data replication.

If you are configuring data replication for an on-premise and cloud combination, ensure that you select the appropriate value for `cloudVendor` and for load balancer-based network traffic management, set `cvmaster:true`. On ARO, load balancer is not supported.

Note: If a node specified in the data replication custom resource is down, data replication configuration is unresponsive. After the node is back, data replication is configured.

Complete the following steps:

1. Edit `/infoscale-yamls-v9.1.2/DR/SampleDataReplication.yaml` on the primary cluster as under for replication of persistent data(PVs and PVCs) associated with application components in the specified namespace and labels.

```
apiVersion: infoscale.veritas.com/v1
kind: DataReplication
metadata:
  name: <Name for data replication>
  #namespace: <Enter the namespace>
spec:
  # In case of load balancer based n/w traffic management,
  # set lbEnabled to true.
  #  else the default value should be always kept false.
  lbEnabled : false
  # Virtual IP address to configure VVR
  localHostAddress: <Any free Virtual IP address
                to configure VVR for the primary cluster>
  # Corresponding netmask to configure VVR
  localNetMask: <Corresponding netmask to configure VVR>
  # Corresponding network interface to configure VVR
  # (If NIC name is identical for all nodes)
  localNIC: eth0
  # Corresponding network interface list (hostname and NIC name )
  # to configure VVR
  # (If NIC name is not identical for all nodes)
  #localNICList:
  #  - "host1=eth0"
  #  - "host2=eth0"
  #  - "host3=eth0"
  #  - "host4=eth1"

  # (optional) Cloud Vendor (e.g Azure/Aws) on Primary VVR site.
  cloudVendor: Local

  # (optional) Applicable for Cloud-Vendor based environments,
  # If "localHostAddress" value is an Overlay N/w IP, then specify
```

```
#  all applicable Route table resource ids.
#routeTableResourceIds:
#  - "rtb-fb97ac9d"
#  - "rtb-f416eb8d"
#  - "rtb-e48be49d"

# Namespace and optionally labels for which you want
#  to configure data replication
selector:
  namespace: mysql
  #labels:
  #  - "app=db,env=dev"

# Current primary cluster name - Name of the cluster
#    you want to back up
currentPrimary: <Current primary cluster name
                -Name of the cluster you want to back up>

# (optional) In case of takeover operation,
# specify force to true along with
# the updated currentPrimary value.
# In case of migrate operation,
# force should be specified as false and
#  only currentPrimary needs
# to be updated.
force: false

# Secondary cluster details
remoteClusterDetails:
    # ID of the Cluster to be used for a backup
  - clusterName: <ID of the cluster to be used for a backup>
    # In case of load balancer based n/w traffic management,
    #  set remoteLbEnabled to true.
    #  else the default value should be always kept false.
    remoteLbEnabled : false
    # Virtual IP address for VVR configuration of this cluster
    remoteHostAddress: <Any free Virtual IP address for
                       VVR configuration of the remote cluster>
    # Corresponding Netmask of this cluster
    remoteNetMask: <Corresponding netmask of the remote cluster>
    # Corresponding Network interface of this cluster
    remoteNIC: eth0
    # Corresponding Network interface list of this cluster
```

```
#remoteNICList:
#  - "host5=eth1"
#  - "host6=eth0"
#  - "host7=eth0"
#  - "host8=eth1"

# (optional) Cloud Vendor (e.g Azure/Aws) on remote VVR site.
remoteCloudVendor: Local

# (optional) Applicable for Cloud-Vendor based environments,
# If "remoteHostAddress" value is an Overlay N/w IP,
# then specify all applicable Route table resource ids.
#remoteRouteTableResourceIds:
#  - "rtb-fb97ac9d"
#  - "rtb-f416eb8d"
#  - "rtb-e48be49d"

# (optional) replication type can be sync or async.
# default value will be async if not specified.
replicationType: async

# (optional) replicationState can have values start, stop, pause
#     and resume.
# This field can be updated to start/stop/pause/resume replication.
# Default value will be set to start during initial configuration.
replicationState: start

# (optional) network transport protocol can be TCP or UDP.
# Default value will be set to TCP during initial configuration and
# can be later changed to UDP.
networkTransportProtocol: TCP

# (optional) By default, it will be set to N/A during
# initial configuration, which means the available bandwidth
# will be used.It can be later changed to set the
# maximum network bandwidth (in bits per second).
bandwidthLimit: N/A

# (optional) Supported values for latency protection are:
#  fail, disable and override.
# By default it will be set to disable during initial configuration
# and can be changed later.
latencyProtection: disable
```

```
# (optional) Supported values log (SRL) protection are:
# autodcm, dcm, fail, disable and override.
# By default it will be set to autodcm during initial configuration
# and can be changed later.
logProtection: autodcm
```

Note: Ensure that the current primary cluster name you enter here must be the same that you plan to specify in `DisasterRecoveryPlan.yaml`. For every Disaster Recovery Plan, you must create a separate Data Replication CR. Ensure that namespace and labels in Disaster Recovery Plan and its corresponding Data Replication CR are identical. If you want to exclude any PVCs within the same namespace from replication, label the PVCs as `` `org.veritas.infoscale/exclude-from-backup=true` ``.

2. Run the following command on the bastion node:

   ```
   oc apply -f /infoscale-yamls-v9.1.2/DR/SampleDataReplication.yaml
   ```

3. After these commands are executed, run the following commands on the bastion node:

   ```
   oc get datarep
   ```

   Review the output similar to the following:

   ```
   NAME PROPOSED PRIMARY CURRENT PRIMARY NAMESPACE LABELS
   <Name of data replication resource> <proposed cluster ID>
        <current cluster ID> <namespace> <labels if any>
   ```

   ```
   oc get datarep -o wide
   ```

   Review the output similar to the following:

   ```
   NAME PROPOSED PRIMARY CURRENT PRIMARY NAMESPACE LABELS
   postgres-rep Clus1 Clus1 postgres <none>
    | replicating (connected) | behind by 0h 0m 0s


   REPLICATION   SUMMARY
   asynchronous | consistent,up-to-date
   ```

4. Wait for the initial synchronization of the application Persistent Volumes to complete on the DR site. Run the following command on the bastion node of the DR site.

   ```
   oc describe datarep <Data rep name for the application>
   ```

   Review the status in the output similar to the following. Data Status must be consistent up-to-date.

   ```
   Spec:
   ..
   ..
   Status:
   ..
   ..
      Primary Status:
      ..
      ..
      Secondary Status:
      ..
      Data Status:  consistent,up-to-date
   ```

Configuring DNS

Optionally, using DNS custom resource you can configure a DNS resource that updates the DNS server entries in the event of a failover or migration. The DNS CR must to be separately applied on all DR clusters. When configured, the DNS CR monitors the resource records for the hostname and IP address mappings on the DNS servers. When the Disaster Recovery Plan is configured, the DNS pointer can be provided in the Disaster Recovery Plan CR. Whenever, the DR plan is activated on any primary cluster, the configured DNS is also activated with the provided hostname and IP addresses. When the disaster recovery plan is migrated, the DNS entries from the primary site are removed and the DNS entries on the secondary site are updated.

DNS resource states can be as follows:

State	Description
INIT	Default state, not active.
OFFLINE	Corresponding DNS resource is offline. State on non-active cluster.
ONLINE	DNS entries are configured and DNS resource is online. State on the active primary cluster.

State	Description
FAULTED	Underlying DNS resource is faulted

Applicable only to Secure Linux DNS

1 Following steps are the prerequisites for `SampleDNS.yaml`. DNS private key and DNS key must be added to `infoscale-dns-secret`.

- Run the following command on the bastion node

  `cat dns.private | base64`

  Copy and remove all unnecessary spaces from the <dns private key> that is displayed.

- Run the following command on the bastion node

  `cat dns.key | base64`

  Copy and remove all unnecessary spaces from the <dns key> that is displayed.

- Run the following command on the bastion node to generate the keys

  `oc get secret -n infoscale-vtas |grep infoscale-sds-dns-secret`

  Use the output in the following command.

- Run the following command on the bastion node and add the keys

  `oc edit secret <output for infoscale-sds-dns-secret> -n infoscale-vtas`

  ```
  apiVersion: v1
  data:
   dns.private: <dns private key>
   dns.key: <dns key>
  kind: Secret
  ```

  Note: You can add the data: section if it is not present in the file.

- Save and close the file.

- Run the following command to verify whether addition of keys is successful

  `oc get secret infoscale-dns-secret -n infoscale-vtas -o json`

  Review the output as under

  ```
  {
      "apiVersion": "v1",
      "data": {
          "dns.key": "<dns key>",
  ```

```
        "dns.private": "<dns private key>"
    },
    "kind": "Secret",
```

- The private key files are created in `/etc/vx/dns-certs/` . You can run the following command on any of the InfoScale pods.
  `ls -l /etc/vx/dns-certs/dns.*`
  Review the output as under

```
lrwxrwxrwx. 1 root root 18 Oct 18 05:10 /etc/vx/dns-certs/dns.key
                                          -> ..data/dns.key
```

```
lrwxrwxrwx. 1 root root 18 Oct 18 05:10 /etc/vx/dns-certs/dns.private
                                     -> ..data/dns.private
```

2 Edit `/infoscale-yamls-v9.1.2/DR/SampleDNS.yaml` as under:

Note: Ensure that this resource is applied in the same namespace as that of data replication configuration resource.

```
apiVersion: infoscale.veritas.com/v1
kind: DNS
metadata:
  name: <Name of DNS>
  #namespace: <Enter the namespace>
spec:
  # Domain name for the DNS
  domain: "<Add example.com here>"

  # (optional) Path for the file containing private TSIG key,
  # required for secure DNS updates.
  # Configure only for UNIX based DNS server
  #tsigKeyFile: "/etc/vx/dns-certs/dns.private"

  # (optional) The list of primary master name servers in the domain.
  #stealthMasters: ["1.2.3.4"]

  # (optional) An association of DNS resource record value
  # Specify the key values in map format
  #resRecord:
        #"HostName_1" : "100.200.30.41"
        #"HostName_2" : "100.200.30.42"
        #"HostName_3" : "100.200.30.43"
        #"www" : "HostName_1"
        #"abc" : "HostName_2"
        #"xyz" : "HostName_3"

  # (optional) Time to Live value,
  # in seconds for DNS entries in the zone
  # default value is 86400
  #ttl: 86400

  # (optional) Time in seconds after which DNS agent
  # attempts to refresh resrecords on DNS server
  #refreshInterval: 0
```

```
# (optional)  Set to "true" if the DNS server that
# you have configured is a Windows DNS server and
# only if it accepts secure dynamic updates
# default is false
#useGSSAPI: false

# (optional) Set to "true" if you want to clean up all
# the existing DNS records for the configured keys
# before adding new records default is false
#cleanRRKeys: false

# (optional) Set to "true" if DNS online should create PTR records
# for each RR of type A or AAAA
# default is false
#createPTR: false

# (optional) Set to "true" if if DNS offline should remove all records
# defined by ResRecord
# default is false
#offDelRR: false
```

Note: name, namespace, and domain are mandatory here. Update tsigKeyFile for secure DNS only.

---

3. Run the following command on the bastion node::

   `oc apply -f /infoscale-yamls-v9.1.2/DR/SampleDNS.yaml`

4. To verify whether DNS resource is created successfully, run the following command on the bastion node:

   `oc -n infoscale-vtas get dns.infoscale.veritas.com/Name of DNS`

5. Review output similar to the following:

```
NAME        DOMAIN        STATE
Name of DNS example.com   INIT
```

Note: You must create a DNS resource with its attributes on each member cluster as DNS CR is not synchronized across peer clusters.

Configuring Disaster Recovery Plan

With a Disaster Recovery Plan (DR Plan) you can enable disaster recovery for a particular namespace. For a more granular control, you can selectively label components in the namespace and create a DR Plan with namespace and labels. A DR Plan cannot span multiple namespaces. DR Plan must be created only on the primary cluster. DR Plan is automatically created and synchronized on all peer clusters after its creation on the primary cluster. Migration and other operations on the namespace can be triggered by updating certain attributes.

Note: Clusterwide(across namespaces) RBAC permissions are granted to the service account of `infoscale-dr-manager` pod as required. As a Storage Administrator, ensure that a non-privileged user does not have `oc exec` permissions on the infoscale-dr-manager pod.

---

1 Edit `/infoscale-yamls-v9.1.2/DR/SampleDisasterRecoveryPlan.yaml` as under to create DR plan for application components in a given namespace.

Note: Ensure that this resource is applied in the same namespace as that of data replication configuration resource.

```
apiVersion: infoscale.veritas.com/v1
kind: DisasterRecoveryPlan
metadata:
  name: <Name of the Disaster Recovery Plan>
  #namespace: <Enter the namespace>
spec:
  # Name of cluster that should be treated as primary for this DR plan
  primaryCluster: <ID of the cluster you want to back up>

  # (optional) Set force to True if peer cluster(s) is not reachable
  # and local cluster needs to perform takeover
  force: false

  # List of member cluster(s) where this DRPlan can failover.
  # Sequence of MemberCluster specified in this list denotes
  # relative preference of member cluster(s)
  # Must be subset of Global Cluster Membership
  preferredClusterList: ["<ID of the cluster you want to back up>",
                         "<ID of the cluster where you want to back up>"]

  # Kind of corrective action in case of disaster
  # default value will be "Manual" if not specified
  clusterFailOverPolicy: Manual

  # Specify namespace and optionally labels to decide what
  # all needs to be part of the disaster recovery plan
  selector:
    namespace: mysql
    #labels:
    #  - "app=db,env=dev"

  # (optional) Pointer to manage data replication
  #dataReplicationPointer: test-datareplication
```

```
# (optional) Pointer to manage DNS endpoints
#dnsPointer: test-dns
```

Note: dataReplicationPointer is needed only if you have stateful applications that require data replication across peer clusters.

---

2 Run the following command on the bastion node:

```
oc apply -f
/infoscale-yamls-v9.1.2/DR/SampleDisasterRecoveryPlan.yaml
```

Note: Ensure that the Namespace you enter here is the same namespace specified in the data replication configuration resource.

---

3 Wait till the command run is successful and the following message appears.

```
disasterrecoveryplan.infoscale.veritas.com/
                  <Name of Disaster recovery plan> created
```

4 Run the following command on the bastion node:

```
oc get drplan
```

5 Review the output similar to the following:

```
NAME      PREFERREDCLUSTERLIST SPEC.PRIMARYCLUSTER
<Name of("ID of the cluster  "ID of cluster
Disaster you want "           where you want
recovery  to back up          to back up")
plan>

STATUS.PRIMARYCLUSTER DATAREPLICATION DNS
ID of the current     ID of the current
cluster               cluster
```

Upgrading InfoScale DR Manager operator

Prerequisites

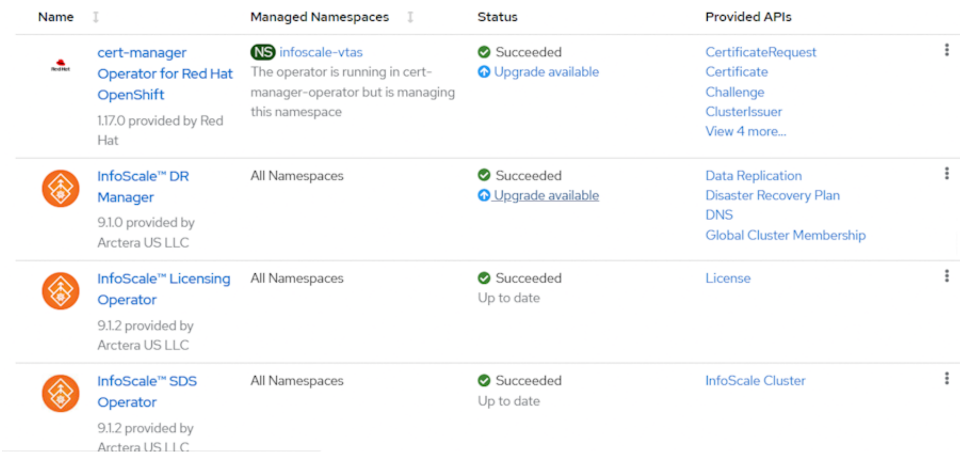
Make sure InfoScale SDS is upgraded before upgrading InfoScale DR Manager operator.

To upgrade InfoScale DR Manager operator

1. Connect to OpenShift console
   - For OpenShift version below 4.20 navigate to Operator → Installed Operators page.
   - For OpenShift version above 4.20 , navigate to the Ecosystem > Software Catalog page.

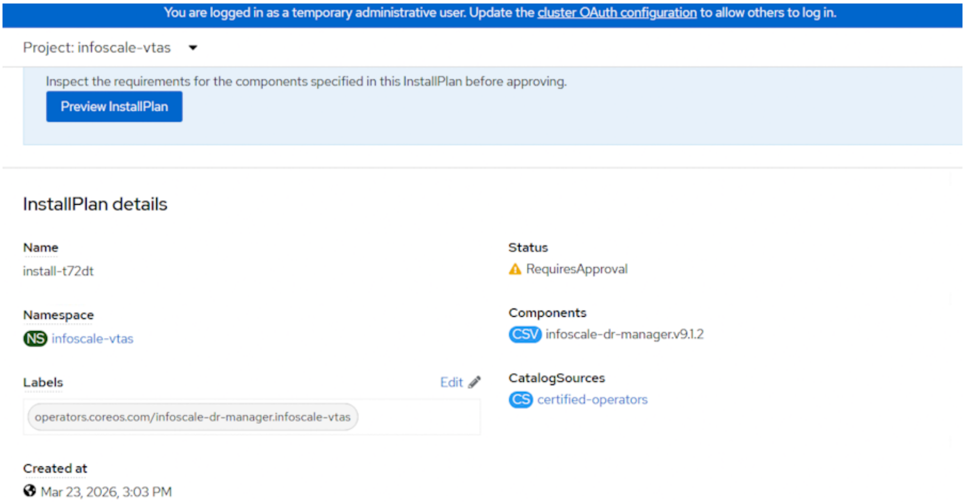
   You should see all InfoScale operators listed here.

2. ForInfoScale DR Manager, available upgrade is reported in Status.

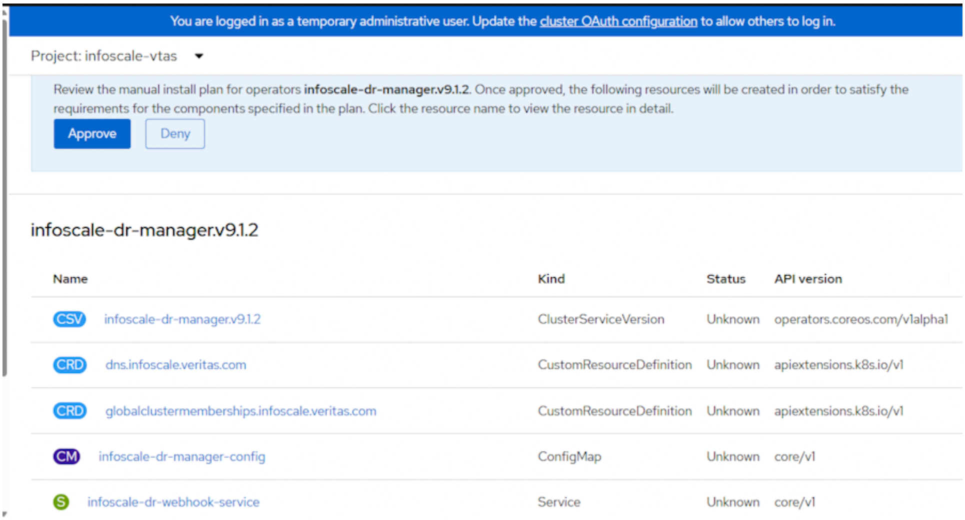


3. Click InfoScale DR Manager to upgrade it.

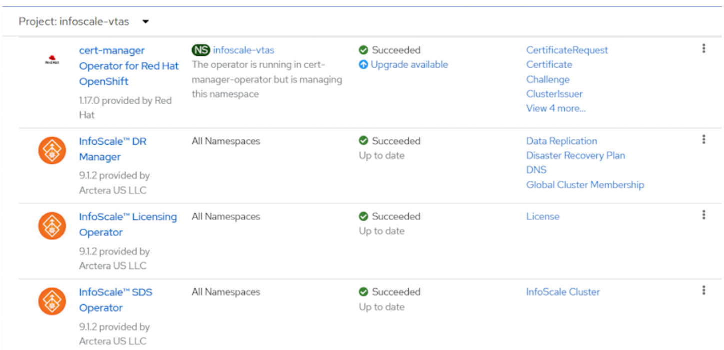
4 In Subscription, updates that need approval are displayed next to upgrade Status. Click Preview InstallPlan.



5 Review the details list. Click Approve to proceed with the upgrade.



6 Wait till it is upgraded and listed in Operators > Installed Operators.



Tech Preview: Disaster recovery in OpenShift virtualization environment

Note: OpenShift-V DR is strictly a tech preview and this should not be used in a production environment.

VIKE introduces Virtual Machine Disaster Recovery (VM-DR) as a Tech Preview capability for protecting virtualized workloads running on OpenShift Container Platform (OCP) 4.17 and later. VM-DR extends the existing DR Manager workflow by enabling the capture, migration, and restoration of VirtualMachine and VirtualMachinePool resources across clusters participating in a DisasterRecoveryPlan.

VM-DR extends InfoScale's disaster recovery workflow to support virtualized workloads on OpenShift by providing: Automated shutdown, capture, migration, restore, and restart operations No additional installation or configuration requirements Predictable migration, failover, and failback workflows Strong consistency validation across source and target clusters Preservation of persistent storage to support seamless failback VM-DR integrates directly with existing DR constructs—GlobalClusterMembership, DataReplication, and DisasterRecoveryPlan—and requires no changes to the established workflow.

Note: VM-DR has been validated for up to 5 to 7 VirtualMachines. Larger VM counts are not supported.

Capturing and replicating VM resources

VM-DR captures only the VM-specific and application-related objects required to recreate workloads on the DR site. Cluster-level, runtime-generated, or non-portable resources are intentionally excluded.

The following categories of resources are not replicated:

Excluded API resources

- events.events.k8s.io
- poddisruptionbudgets.policy
- virtualmachinerevisions.kubevirt.io
- virtualmachineinstances
- virtualmachineinstances.kubevirt.io
- virtualmachineinstancemigrations.kubevirt.io
- virtualmachineexports.export.kubevirt.io
- virtualmachinesnapshots.snapshot.kubevirt.io
- virtualmachinesnapshotcontents.snapshot.kubevirt.io
- virtualmachinerestores.snapshot.kubevirt.io
- virtualmachineclones.clone.kubevirt.io

- virtualmachineinstancepresets.kubevirt.io

Excluded CDI and storage-related resources

- cdiconfigs.cdi.kubevirt.io
- cdis.cdi.kubevirt.io
- storageprofiles.cdi.kubevirt.io
- dataimportcrons.cdi.kubevirt.io
- volumeclonesources.cdi.kubevirt.io
- volumeimportsources.cdi.kubevirt.io
- volumeuploadsources.cdi.kubevirt.io

Excluded operator, networking, and cluster-level resources

- networkaddonsconfigs.networkaddonsoperator.network.kubevirt.io
- hyperconvergeds.hco.kubevirt.io
- kubevirts.kubevirt.io
- virtualmachineclusterinstancetypes.instancetype.kubevirt.io
- virtualmachineclusterpreferences.instancetype.kubevirt.io
- virtualmachineinstancetypes.instancetype.kubevirt.io
- virtualmachinepreferences.instancetype.kubevirt.io

VirtualMachineInstanceReplicaSet is not supported.

Supported workload resources

- VirtualMachine
- VirtualMachinePool

Graceful workload shutdown

Before removal on the source cluster, VM-DR performs a graceful shutdown equivalent to executing `virtctl stop <vm>`.

Workloads should configure an adequate shutdown window using `spec.template.spec.terminationGracePeriodSeconds`.

Note: The maximum supported grace period is 3600 seconds (1 hour).

VMs that do not shut down within this period are terminated so that the DR process can continue.

Resource shutdown order

VM-DR removes resources in a controlled sequence to ensure consistency:

1. Remove VirtualMachinePool objects.
2. Stop and delete VirtualMachine objects.
3. Delete DataVolumes associated with the VM.
4. Remove remaining DR-captured namespace resources.

Prerequisites and compatibility requirements

Both clusters must provide matching virtualization runtimes, including:

- QEMU/KVM version
- Machine type (for example, pc-q35)
- OpenShift Virtualization components (KubeVirt, CDI, HCO)

A mismatch in these components can cause VirtualMachineInstance (VMI) synchronization failures during restore.

Example:

```
Warning SyncFailed virt-handler server error. command SyncVMI failed:
 "LibvirtError(Code=67, Domain=10, Message='unsupported configuration:
 Emulator /usr/libexec/qemu-kvm' does not support machine
type 'pc-q35-rhel9.6.0'')"
```

Ensuring version parity prevents restore failures and guarantees correct VMI reconstruction on the target cluster.

Required operators on the DR site

The following operators must exist and be in a running state:

- HyperConverged Operator (HCO)
- KubeVirt Operator
- CDI Operator
- Any required networking add-ons

Note: Cluster-level resources are not backed up. VM-DR backs up only namespace-scoped resources.

The system backs up only the following:

- CRDs
- SecurityContextConstraints
- ClusterRoles
- StorageClasses

All other cluster-level resources must already exist at the DR site. Administrators must ensure that VM-related cluster dependencies are manually created and properly configured on the target cluster, including:

- SR-IOV configuration
- PCI passthrough mappings
- Multus network definitions
- Host-level features used by the VM

Deployment behavior

VM-DR is supported by the standard InfoScale VIKE deployment.

No additional operators, controllers, or configuration steps are required.

Migration and failback considerations

To keep VM migration, failover, and failback smooth, the DR workflow leaves persistent storage on the source cluster even if VMs are removed during a DR event. This protects application data and allows fast, consistent recovery.

Persistent volume retention

Configure each VM's PersistentVolume (PV) with `spec: persistentVolumeReclaimPolicy: Retain`.

- This setting makes sure the PV is not deleted when its PVC or VM is removed during migration or failover.
- Keeping the PV lets the VM reattach to its original data during restore or failback. It reduces recovery time and prevents data loss.

Automatic policy correction

If the DR workflow sees that Retain is not set:

- It finds the VM's DataVolume (DV).
- It locates the related PVs and updates their reclaim policy to Retain.

This keeps failover and failback consistent and safe, even if the original PV was misconfigured.

Recommended storage practices

- For best results, create and link DVs and PVs to VirtualMachines using standard OpenShift Virtualization workflows.
- Manually created DVs or PVs may not be updated automatically by the DR workflow. They can miss reclaim policy fixes. Using integrated provisioning simplifies recovery and avoids storage inconsistencies during DR operations.

Chapter 11

Installing and configuring InfoScale DR Manager on Kubernetes

This chapter includes the following topics:

Introduction

This section informs you how to install Custom Resource (CR) files related to Disaster Recovery (DR).

Note: Multiple InfoScale clusters in different namespaces can be configured within a single Kubernetes cluster. However, DR configuration is supported for a single InfoScale cluster per Kubernetes cluster only.

Prerequisites

1. InfoScale must be installed and InfoScale pods must be configured on the clusters.

2. Load balancer must be installed and configured. If you choose to install Metallb as the load balancer, steps are listed in `Installing InfoScale DR`.
3. Be ready with the following information for every member cluster:
   - GlobalClusterMembership: Virtual IP address and a port for each member cluster in GCM. This information is captured in GCM custom resource.
   - Data Replication: Virtual IP address, port, netmask and network interface (NIC) required for each cluster to be specified in DataReplication CR. This information is exclusive for a DataReplication CR and is used to configure InfoScale Volume Replicator.
4. Ensure that stale Custom Resources (CR) and Custom Resource Definitions (CRD) related to DR do not exist on the clusters.
5. Optionally if you want to configure DNS resource, DNS key and DNS private key. To know how to obtain keys, see https://www.veritas.com/content/support/en_US/0doc/129694359-129694362-0/uxrt-731_id-SF1J0175244-129694362 .

Creating Persistent Volume for metadata backup

Ensure that you run the following steps on the primary and secondary clusters to create a PVC for backing up and restoring application metadata across clusters.

- Create a Storage Class by running the following `sc.yaml`.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: infoscale-dr-csi-sc
parameters:
  fstype: vxfs
  layout: mirror
allowVolumeExpansion: true
provisioner: org.veritas.infoscale
reclaimPolicy: Delete
```

- Create a Persistent Volume Claim (PVC) by running the following `pvc.yaml`.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: infoscale-dr-meta-bkp-pvc
```

```
    namespace: infoscale-vtas
  spec:
    accessModes:
    - ReadWriteOnce
    resources:
      requests:
        storage: 4Gi
    storageClassName: infoscale-dr-csi-sc
    volumeMode: Filesystem
```

External dependencies

- Load balancer service must be installed on all clusters that are a part of Disaster Recovery (DR) to allocate Virtual IP addresses for cluster-to-cluster communications. Virtual IP addresses ensure a resilient communication between the clusters. See Installing Metallb to install Metallb. Alternatively, you can install any other load balancer service. Refer to its documentation. If you are installing in a cloud environment, ensure that you use native load balancers.

Installing InfoScale DR Manager

This section informs you how to install and configure Disaster Recovery for your InfoScale cluster.

Note: When you download, unzip, and untar `infoscale-yamls-v9.1.2.tar.gz`, all files required for installation are available.

Complete the following steps to install the InfoScale DR Manager on the source and target DR cluster.

Creating application user role

1. Run the following command on the master node.

   ```
   kubectl apply -f
   infoscale-yamls-v9.1.2/DR/infoscale-dr-admin-role.yaml
   ```

2. Copy the following content to `infoscale-yamls-v9.1.2/DR/ClusterRoleBinding .yaml`.

   ```
   ---
   apiVersion: rbac.authorization.k8s.io/v1
   kind: ClusterRoleBinding
   metadata:
      name: infoscaledr-role-binding
   roleRef:
      apiGroup: rbac.authorization.k8s.io
      kind: ClusterRole
      name: infoscaledr-admin-role
   subjects:
   -  kind: User
      name: postgres-admin
      apiGroup: rbac.authorization.k8s.io
   ```

   This is an example for a ClusterRoleBinding role.

3. Run the following command on the master node.

   ```
   kubectl apply -f infoscale-yamls-v9.1.2/DR/ClusterRoleBinding.yaml
   ```

For DR controller installation and Global Cluster Membership (GCM) configuration, you must use the kubeadmin role only. To configure DR plan and data replication, you can use this role or the kubeadmin role.

1. Edit `/infoscale-yamls-v9.1.2/DR/Kubernetes/dro_deployment.yaml` to update the path with private repository path where InfoScale DR Manager image is saved and pulled for installation.

2. Run the following command on the master node of each cluster.

   ```
   kubectl apply -f
   /infoscale-yamls-v9.1.2/DR/Kubernetes/dro_deployment.yaml
   ```

3. Wait till the command execution is complete.

4 Run the following command on the master node to verify if the deployment is successful.

```
kubectl -n infoscale-vtas get pods
```

See the Status in the output similar to the following

```
NAME                          READY   STATUS     RESTARTS    AGE
infoscale-dr-manager-xxxx     1/1     Running    0           114m
```

Status must change from `ContainerCreating` to `Running`.

5 Run the following commands to configure ExternalIP for DR pod.

Note: Run these steps only if you want Metallb as the load balancer. If you choose any other load balancer, refer to its documentation for installation and configuration.

```
kubectl -n infoscale-vtas expose deployment infoscale-dr-manager
--name my-lb-service --type LoadBalancer --protocol TCP --port
14155 --target-port 14155
```

Here, DR controller uses port 14155 internally to communicate across peer clusters. After a successful installation and configuration, you can verify by running the following command

6 `kubectl get svc my-lb-service`

An output similar to the following indicates that installation and configuration is successful

```
NAME          TYPE         CLUSTER-IP   EXTERNAL-IP  PORT(S)         AGE
my-lb-service LoadBalancer <IP address> <IP address> 14155:14155/TCP 13h
```

Run this command on both the clusters and verify if installation and configuration is successful. Verify whether `EXTERNAL-IP` is accessible from one cluster to the other cluster.

Configuring Global Cluster Membership (GCM)

With Global Cluster Membership (GCM), you can define membership of clusters for disaster recovery. The GCM CR must be configured and applied on all clusters. When configured, the Global Cluster Membership forms a logical notion called 'Global Cluster' with all underlying clusters as 'Member Clusters'. Member clusters are

Kubernetes clusters providing disaster recovery capabilities to application components. To provide DR, these member clusters

1. Send heartbeats with each other periodically.
2. Exchange information like state, configuration, operation.
3. Perform/participate in operation like migration.

Complete the following steps:

1. Edit `/infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml` as under:

```
apiVersion: infoscale.veritas.com/v1
kind: GlobalClusterMembership
metadata:
  name: global-cluster-membership
spec:
  # Local cluster name in the global membership
  localClusterName: <Local cluster
                         where you want to apply this YAML>
  globalMemberClusters:
      # Cluster ID of each member of global cluster membership
    - clusterID: <A unique ID of the primary cluster>
      # Address Used For Communicating With Peer Cluster's DR Controller
      drControllerAddress: "<Load balancer IP address or haproxy
                                              of the local cluster>"
      # Port used for DR controller
      drControllerPort: "<Load balancer port number>"
    - clusterID: <A unique ID of the secondary cluster>
      drControllerAddress: "<Load balancer IP address or haproxy
                                               of the DR site>"
      drControllerPort: "<Load balancer port number>"
  # If heartbeat with peer cluster missed more than CounterMissTolerance
         times,then cluster will be moved to FAULTED state
  counterMissTolerance: 5
  globalClusterOperation: "none"
  # Application metadata backup sync frequency to DR site(s) in minutes
  metadataBackupInterval: 15
  # Refresh data replication status after specified minutes
  datarepRefreshStatusFrequency: 10
  # Include cluster-scoped Custom Resource Definitions (CRDs)
  #                           in disaster recovery plan backup
  backupClusterScopeCRD: true
```

```
  # Maximum metadata backup copies stored per DR plan
  maximumMetadataCopies: 5
```

Note: Do not enclose the parameter values in angle brackets(< >) . For example, if 8334 is the Load balancer port number; enter drControllerPort: "8334" for drControllerPort: "<Load balancer port number>". localClusterName and clusterID can have maximum 20 characters.

2. Run the following command on the master node of the source cluster.

   ```
   kubectl apply -f
   /infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml
   ```

3. Edit another instance of `/infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml` to add DR site as under

   ```
   apiVersion: infoscale.veritas.com/v1
   kind: GlobalClusterMembership
   metadata:
     name: global-cluster-membership
   spec:
     # Local cluster name in the global membership
   localClusterName: <DR site cluster name
                          where you want to apply this YAML>

     globalMemberClusters:
     # Cluster ID of each member of global cluster membership
      - clusterID: <A unique ID of the primary cluster>
     # Address Used For Communicating With Peer Cluster's DR Controller
       drControllerAddress: "<Load balancer IP address or haproxy
                                        of the local cluster>"

     # Port used for DR controller
        drControllerPort: "<Load balancer port number>"
         - clusterID: <A unique ID of the secondary cluster>
        drControllerAddress: "<Load balancer IP address or haproxy
        of the DR site>"
         drControllerPort: "<Load balancer port number>"
     # If heartbeat with peer cluster missed more than CounterMissTolerance
     # times, then cluster will be moved to FAULTED state
     counterMissTolerance: 5
   ```

```
globalClusterOperation: "none"
# Application metadata backup sync frequency to DR site(s) in minutes
metadataBackupInterval: 15
# Refresh data replication status after specified minutes
datarepRefreshStatusFrequency: 10
# Include cluster-scoped Custom Resource Definitions (CRDs) in
# disaster recovery plan backup
backupClusterScopeCRD: true
# Maximum metadata backup copies stored per DR plan
maximumMetadataCopies: 5
```

4. Copy this file to the DR site and Run the following command again on the master node of the DR site.

   ```
   kubectl apply -f
   /infoscale-yamls-v9.1.2/DR/SampleGlobalClusterMembership.yaml
   ```

5. Manually verify on all clusters whether the GLOBALCLUSTERSTATE is DISCOVER_WAIT by running the command on the master node of the cluster- `kubectl get gcm`.

   Various states are as follows:

State	Description
UNKNOWN	A transient default Global-Cluster state. After initial configuration/setup, cluster state must transition to DISCOVER_WAIT. Prolonged UNKNOWN state indicates errors in initial configuration/setup. Review DR Controller log for the ongoing activities.
DISCOVER_WAIT	Although local cluster has a copy of GCM and member cluster details, it is not certain whether local copy of GCM and member cluster is up-to-date. If GCM and member cluster details are identical on all peer clusters then all clusters automatically transition to RUNNING state. If the details are not identical, waits till you seed the cluster by updating GlobalClusterOperation to localbuild. When a member cluster transitions to RUNNING state, all peer clusters with identical membership transition to RUNNING state.
ADMIN_WAIT	If local membership definition does not match with peer cluster's membership definition, clusters transition to this state. Update membership on peer clusters and ensure that it is identical. Peer clusters then transition to RUNNING state.

State	Description
RUNNING	Cluster transitions to RUNNING state if you seed cluster membership by updating GlobalClusterOperation to localbuild. Cluster transitions to RUNNING state even when local copy of membership matches with peer clusters.
EXITING	You have initiated DR Controller stop.
EXITED	DR Controller stopped.

6. To verify whether the Global Cluster is successfully created, run the following command on the master node:

```
kubectl get gcm
```

7. Review the cluster names, GlobalClusterState, and PeerLinkState in the output similar to the following. GlobalClusterState must be Running and PeerLinkState must be Connected.

```
NAME                       CLUSTER NAME   CLUSTER STATE   PROTOCOL
global-cluster-membership Clus1          RUNNING         10


PEER LINK STATE
{"Clus1":"CONNECTED","Clus2":"CONNECTED"}
```

Here, `NAME` is the `Name of GlobalClusterMembership custom resource`, `CLUSTER NAME` is the `Local Cluster ID`, and `Clus1`, `Clus2` are the Cluster IDs that you defined for global membership.

Configuring Data Replication

Using Data Replication custom resource you can configure replication for persistent data (PVs and PVCs) associated with application components in a namespace. Custom resource created on a cluster is automatically synchronized on all peer clusters. Hence, this CR needs to be configured on the primary cluster only. After CR is configured, replication is set up. InfoScale Volume Replicator(VVR) is responsible for performing replication. You can check status of underlying replication and perform operations like stop, pause, resume, and migrate data replication.

If you are configuring data replication for an on-premise and cloud combination, ensure that you select the appropriate value for `cloudVendor` and for load balancer-based network traffic management, set `cvmaster:true`.

Note: If a node specified in the data replication custom resource is down, data replication configuration is unresponsive. After the node is back, data replication is configured.

Complete the following steps:

1. Edit `/infoscale-yamls-v9.1.2/DR/SampleDataReplication.yaml` on the primary cluster as under for replication of persistent data(PVs and PVCs) associated with application components in the specified namespace and labels.

```
apiVersion: infoscale.veritas.com/v1
kind: DataReplication
metadata:
  name: <Name for data replication>
  #namespace: <Enter the namespace>
spec:
  # In case of load balancer based n/w traffic management,
  # set lbEnabled to true.
  #  else the default value should be always kept false.
  lbEnabled : false
  # Virtual IP address to configure VVR
  localHostAddress: <Any free Virtual IP address
               to configure VVR for the primary cluster>
  # Corresponding netmask to configure VVR
  localNetMask: <Corresponding netmask to configure VVR>
  # Corresponding network interface to configure VVR
  # (If NIC name is identical for all nodes)
  localNIC: eth0
  # Corresponding network interface list (hostname and NIC name )
  # to configure VVR
  # (If NIC name is not identical for all nodes)
  #localNICList:
  #  - "host1=eth0"
  #  - "host2=eth0"
  #  - "host3=eth0"
  #  - "host4=eth1"

  # (optional) Cloud Vendor (e.g Azure/Aws) on Primary VVR site.
  cloudVendor: Local

  # (optional) Applicable for Cloud-Vendor based environments,
  # If "localHostAddress" value is an Overlay N/w IP, then specify
```

```
#  all applicable Route table resource ids.
#routeTableResourceIds:
#  - "rtb-fb97ac9d"
#  - "rtb-f416eb8d"
#  - "rtb-e48be49d"

# Namespace and optionally labels for which you want
#  to configure data replication
selector:
  namespace: mysql
  #labels:
  #  - "app=db,env=dev"

# Current primary cluster name - Name of the cluster
#    you want to back up
currentPrimary: <Current primary cluster name
                -Name of the cluster you want to back up>

# (optional) In case of takeover operation,
# specify force to true along with
# the updated currentPrimary value.
# In case of migrate operation,
# force should be specified as false and
#  only currentPrimary needs
# to be updated.
force: false

# Secondary cluster details
remoteClusterDetails:
    # ID of the Cluster to be used for a backup
  - clusterName: <ID of the cluster to be used for a backup>
    # In case of load balancer based n/w traffic management,
    #  set remoteLbEnabled to true.
    #  else the default value should be always kept false.
    remoteLbEnabled : false
    # Virtual IP address for VVR configuration of this cluster
    remoteHostAddress: <Any free Virtual IP address for
                       VVR configuration of the remote cluster>
    # Corresponding Netmask of this cluster
    remoteNetMask: <Corresponding netmask of the remote cluster>
    # Corresponding Network interface of this cluster
    remoteNIC: eth0
    # Corresponding Network interface list of this cluster
```

```
#remoteNICList:
#  - "host5=eth1"
#  - "host6=eth0"
#  - "host7=eth0"
#  - "host8=eth1"

# (optional) Cloud Vendor (e.g Azure/Aws) on remote VVR site.
remoteCloudVendor: Local

# (optional) Applicable for Cloud-Vendor based environments,
# If "remoteHostAddress" value is an Overlay N/w IP,
# then specify all applicable Route table resource ids.
#remoteRouteTableResourceIds:
#  - "rtb-fb97ac9d"
#  - "rtb-f416eb8d"
#  - "rtb-e48be49d"

# (optional) replication type can be sync or async.
# default value will be async if not specified.
replicationType: async

# (optional) replicationState can have values start, stop, pause
#     and resume.
# This field can be updated to start/stop/pause/resume replication.
# Default value will be set to start during initial configuration.
replicationState: start

# (optional) network transport protocol can be TCP or UDP.
# Default value will be set to TCP during initial configuration and
# can be later changed to UDP.
networkTransportProtocol: TCP

# (optional) By default, it will be set to N/A during
# initial configuration, which means the available bandwidth
# will be used.It can be later changed to set the
# maximum network bandwidth (in bits per second).
bandwidthLimit: N/A

# (optional) Supported values for latency protection are:
#  fail, disable and override.
# By default it will be set to disable during initial configuration
# and can be changed later.
latencyProtection: disable
```

```
# (optional) Supported values log (SRL) protection are:
# autodcm, dcm, fail, disable and override.
# By default it will be set to autodcm during initial configuration
# and can be changed later.
logProtection: autodcm
```

Note: Ensure that the current primary cluster name you enter here must be the same that you plan to specify in `DisasterRecoveryPlan.yaml`. For every Disaster Recovery Plan, you must create a separate Data Replication CR. Ensure that namespace and labels in Disaster Recovery Plan and its corresponding Data Replication CR are identical. If you want to exclude any PVCs within the same namespace from replication, label the PVCs as `` `org.veritas.infoscale/exclude-from-backup=true` ``.

---

2. Run the following command on the master node:

   ```
   kubectl apply -f
   /infoscale-yamls-v9.1.2/DR/SampleDataReplication.yaml
   ```

3. After these commands are executed, run the following commands on the master node:

   ```
   kubectl get datarep
   ```

4. Review the output similar to the following:

   ```
   NAME PROPOSED PRIMARY CURRENT PRIMARY NAMESPACE LABELS
   <Name of data replication resource> <proposed cluster ID>
               <current cluster ID> <namespace> <labels if any>
   ```

   ```
   kubectl get datarep -o wide
   ```

   Review the output similar to the following:

   ```
   NAME PROPOSED PRIMARY CURRENT PRIMARY NAMESPACE LABELS
   postgres-rep Clus1 Clus1 postgres <none>
    | replicating (connected) | behind by 0h 0m 0s

   REPLICATION   SUMMARY
   asynchronous | consistent,up-to-date
   ```

5. Wait for the initial synchronization of the application Persistent Volumes to complete on the DR site. Run the following command on the master node of the DR site.

   ```
   kubectl describe datarep <Data rep name for the application>
   ```

   Review the status in the output similar to the following. Data Status must be consistent up-to-date.

   ```
   Spec:
   ..
   ..
   Status:
   ..
   ..
      Primary Status:
      ..
      ..
      Secondary Status:
      ..
      Data Status:  consistent,up-to-date
   ```

Additional requirements for replication on Cloud

If any of your sites is on the Cloud, you must configure load balancer service. Ensure that you have specified the cloud service for `cloudVendor` or `remoteCloudVendor` while configuring data replication.

Load balancer can be used for both managed and non-managed clouds. lbEnabled and remoteLbEnabled are set in the datareplication yaml. Default value is false (set to true only in case where network traffic goes over load balancer).

In this configuration, load balancer Virtual IP address must be provided as `HostAddress` (local and/or remote) in the datareplication yaml. The prerequisite for this feature is that the load balancer network Kubernetes service must have the following selector in its spec- `cvmaster:true`. The sample file below is an example of load balancer service.

Note: The TCP/UDP ports that InfoScale Volume Replicator (VVR) uses must be open on all worker nodes of the cluster to enable communication between primary and secondary site. See *Choosing the network ports used by VVR* on the InfoScale support portal.

Note: Perform the following steps only if you want to use a Load balancer for data replication.

---

1 Copy the following content into a yaml file and save the file as `/infoscale-yamls-v9.1.2/DR/vvr-lb-svc.yaml`.

```
apiVersion: v1
kind: Service
metadata:
 annotations:
  service.beta.kubernetes.io/azure-load-balancer-internal: "true"

  name: vvr-lb-svc
  namespace: infoscale-vtas
spec:
  allocateLoadBalancerNodePorts: true
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - name: tcpportone
    port: 4145
    protocol: TCP
    targetPort: 4145
  - name: tcpporttwo
    port: 8199
    protocol: TCP
    targetPort: 8199
  - name: tcpportthree
    port: 8989
    protocol: TCP
    targetPort: 8989
  selector:
    cvmmaster: "true"
  sessionAffinity: None
  type: LoadBalancer
```

2 Run the following command on the master node.

```
kubectl apply -f /infoscale-yamls-v9.1.2/DR/vvr-lb-svc.yaml
```

3. Run the following command on the master node.

   ```
   kubectl get svc vvr-lb-svc  -n infoscale-vtas
   ```

   Load Balancer IP address is returned as the output.

4 Update the load Balancer IP address and copy the following content into a yaml file and save as `/infoscale-yamls-v9.1.2/DR/loadbalancer.yaml`. InfoScale Volume Replicator (VVR) requires both ports - TCP and UDP. Hence, a Load balancer service with mixed protocol support (TCP and UDP) is needed.

```
apiVersion: v1
kind: Service
metadata:
 annotations:
  service.beta.kubernetes.io/azure-load-balancer-internal:"true"
  service.beta.kubernetes.io/azure-load-balancer-internal-subnet:"worker"

  name: vvr-lb-svc
  namespace: infoscale-vtas
spec:
  loadBalancerIP: <Load Balancer IP address>
  allocateLoadBalancerNodePorts: true
  externalTrafficPolicy: Cluster
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - name: tcpportone
    port: 4145
    protocol: UDP
    targetPort: 4145
  - name: tcpporttwo
    port: 8199
    protocol: UDP
    targetPort: 8199
  - name: tcpportthree
    port: 8989
    protocol: UDP
    targetPort: 8989
  selector:
    cvmaster: "true"
  sessionAffinity: None
  type: LoadBalancer
```

5 Run the following command on the master node.

```
kubectl apply -f /infoscale-yamls-v9.1.2/DR/loadbalancer.yaml
```

Configuring DNS

Optionally, using DNS custom resource you can configure a DNS resource that updates the DNS server entries in the event of a failover or migration. The DNS CR must to be separately applied on all DR clusters. When configured, the DNS CR monitors the resource records for the hostname and IP address mappings on the DNS servers. When the Disaster Recovery Plan is configured, the DNS pointer can be provided in the Disaster Recovery Plan CR. Whenever, the DR plan is activated on any primary cluster, the configured DNS is also activated with the provided hostname and IP addresses. When the disaster recovery plan is migrated, the DNS entries from the primary site are removed and the DNS entries on the secondary site are updated.

DNS resource states can be as follows:

State	Description
INIT	Default state, not active.
OFFLINE	Corresponding DNS resource is offline. State on non-active cluster.
ONLINE	DNS entries are configured and DNS resource is online. State on the active primary cluster.
FAULTED	Underlying DNS resource is faulted

Applicable only to Secure Linux DNS

1 Following steps are the prerequisites for `SampleDNS.yaml`. DNS private key and DNS key must be added to `infoscale-dns-secret`.

- Run the following command on the master node:

  `cat dns.private | base64`

  Copy and remove all unnecessary spaces from the <dns private key> that is displayed.

- Run the following command on the master node:

  `cat dns.key | base64`

  Copy and remove all unnecessary spaces from the <dns key> that is displayed.

- Run the following command on the bastion node to generate the keys:

  `kubectl get secret -n infoscale-vtas |grep`

  `infoscale-sds-dns-secret`

  Use the output in the following command.

- Run the following command on the master node and add the keys:

```
kubectl edit secret <output for infoscale-sds-dns-secret> -n
infoscale-vtas
```

```
apiVersion: v1
data:
 dns.private: <dns private key>
 dns.key: <dns key>
kind: Secret
```

Note: You can add the data: section if it is not present in the file.

- Save and close the file.
- Run the following command to verify whether addition of keys is successful:
  `kubectl get secret infoscale-dns-secret -n infoscale-vtas -o json`
  Review the output as under:

```
{
    "apiVersion": "v1",
    "data": {
        "dns.key": "<dns key>",
        "dns.private": "<dns private key>"
    },
    "kind": "Secret",
.
.
```

- The private key files are created in `/etc/vx/dns-certs/` . You can run the following command on any of the InfoScale pods.
  `ls -l /etc/vx/dns-certs/dns.*`
  Review the output as under:

```
lrwxrwxrwx. 1 root root 18 Oct 18 05:10 /etc/vx/dns-certs/dns.key
                                                 -> ..data/dns.key
lrwxrwxrwx. 1 root root 18 Oct 18 05:10 /etc/vx/dns-certs/dns.private
```

```
-> ..data/dns.private
```

2 Edit `/infoscale-yamls-v9.1.2/DR/SampleDNS.yaml` as under:

Note: Ensure that this resource is applied in the same namespace as that of data replication configuration resource.

```
apiVersion: infoscale.veritas.com/v1
kind: DNS
metadata:
  name: <Name of DNS>
  #namespace: <Enter the namespace>
spec:
  # Domain name for the DNS
  domain: "<Add example.com here>"

  # (optional) Path for the file containing private TSIG key,
  # required for secure DNS updates.
  # Configure only for UNIX based DNS server
  #tsigKeyFile: "/etc/vx/dns-certs/dns.private"

  # (optional) The list of primary master name servers in the domain.
  #stealthMasters: ["1.2.3.4"]

  # (optional) An association of DNS resource record value
  # Specify the key values in map format
  #resRecord:
        #"HostName_1" : "100.200.30.41"
        #"HostName_2" : "100.200.30.42"
        #"HostName_3" : "100.200.30.43"
        #"www" : "HostName_1"
        #"abc" : "HostName_2"
        #"xyz" : "HostName_3"

  # (optional) Time to Live value,
  # in seconds for DNS entries in the zone
  # default value is 86400
  #ttl: 86400

  # (optional) Time in seconds after which DNS agent
  # attempts to refresh resrecords on DNS server
  #refreshInterval: 0
```

```
# (optional)  Set to "true" if the DNS server that
# you have configured is a Windows DNS server and
# only if it accepts secure dynamic updates
# default is false
#useGSSAPI: false

# (optional) Set to "true" if you want to clean up all
# the existing DNS records for the configured keys
# before adding new records default is false
#cleanRRKeys: false

# (optional) Set to "true" if DNS online should create PTR records
# for each RR of type A or AAAA
# default is false
#createPTR: false

# (optional) Set to "true" if if DNS offline should remove all records
# defined by ResRecord
# default is false
#offDelRR: false
```

Note: name, namespace, and domain are mandatory here. Update tsigKeyFile for secure DNS only.

3 Run the following command on the master node:

```
kubectl apply -f /infoscale-yamls-v9.1.2/DR/SampleDNS.yaml
```

4 To verify whether DNS resource is created successfully, run the following command on the master node:

```
kubectl -n infoscale-vtas get dns.infoscale.veritas.com/Name of
DNS
```

5 Review output similar to the following:

```
NAME         DOMAIN        STATE
Name of DNS example.com   INIT
```

Note: You must create a DNS resource with its attributes on each member cluster as DNS CR is not synchronized across peer clusters.

Configuring Disaster Recovery Plan

With a Disaster Recovery Plan (DR Plan) you can enable disaster recovery for a particular namespace. For a more granular control, you can selectively label components in the namespace and create a DR Plan with namespace and labels. A DR Plan cannot span multiple namespaces. DR Plan must be created only on the primary cluster. DR Plan is automatically created and synchronized on all peer clusters after its creation on the primary cluster. Migration and other operations on the namespace can be triggered by updating certain attributes.

To enable a non-kubeadmin user (any user other than infoscale-admin) access custom resources on a secondary cluster after applying Disaster Recovery Plan (DR Plan) custom resource on primary cluster, a role and role-binding must be created . After a DR Plan custom resource is created on the primary cluster, an application namespace is automatically created on the secondary cluster. The role and role-binding on the primary cluster can then be saved and applied on secondary cluster in the same namespace.

Note: Clusterwide (across namespaces) RBAC permissions are granted to the service account of `infoscale-dr-manager` pod as required. As a Storage Administrator, ensure that a non-privileged user does not have `kubectl exec` permissions on the infoscale-dr-manager pod.

---

1 Edit `/infoscale-yamls-v9.1.2/DR/SampleDisasterRecoveryPlan.yaml` as under to create DR plan for application components in a given namespace.

Note: Ensure that this resource is applied in the same namespace as that of data replication configuration resource.

```
apiVersion: infoscale.veritas.com/v1
kind: DisasterRecoveryPlan
metadata:
  name: <Name of the Disaster Recovery Plan>
  #namespace: <Enter the namespace>
spec:
  # Name of cluster that should be treated as primary for this DR plan
  primaryCluster: <ID of the cluster you want to back up>

  # (optional) Set force to True if peer cluster(s) is not reachable
  # and local cluster needs to perform takeover
  force: false

  # List of member cluster(s) where this DRPlan can failover.
  # Sequence of MemberCluster specified in this list denotes
  # relative preference of member cluster(s)
  # Must be subset of Global Cluster Membership
  preferredClusterList: ["<ID of the cluster you want to back up>",
                         "<ID of the cluster where you want to back up>"]

  # Kind of corrective action in case of disaster
  # default value will be "Manual" if not specified
  clusterFailOverPolicy: Manual

  # Specify namespace and optionally labels to decide what
  # all needs to be part of the disaster recovery plan
  selector:
    namespace: mysql
    #labels:
    #  - "app=db,env=dev"

  # (optional) Pointer to manage data replication
  #dataReplicationPointer: test-datareplication

  # (optional) Pointer to manage DNS endpoints
```

```
  #dnsPointer: test-dns
```

Note: dataReplicationPointer is needed only if you have stateful applications that require data replication across peer clusters.

---

2 Run the following command on the master node:

```
kubectl apply -f
/infoscale-yamls-v9.1.2/DR/SampleDisasterRecoveryPlan.yaml
```

3 Wait till the command run is successful and the following message appears:

```
disasterrecoveryplan.infoscale.veritas.com/
                  <Name of Disaster recovery plan> created
```

4 Run the following command on the master node:

```
kubectl get drplan
```

5 Review the output similar to the following:

```
NAME     PREFERREDCLUSTERLIST SPEC.PRIMARYCLUSTER
<Name of("ID of the cluster  "ID of cluster
Disaster you want "           where you want
recovery  to back up          to back up")
plan>


STATUS.PRIMARYCLUSTER DATAREPLICATION DNS
ID of the current     ID of the current
cluster               cluster
```

Disaster Recovery scenarios

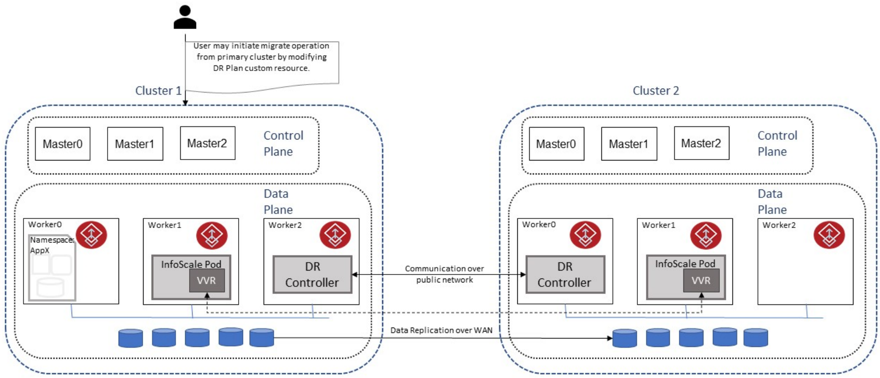
This chapter includes the following topics:

- Migration
- Takeover

Migration

You can initiate migration on a primary cluster when peer clusters are connected, configured for Disaster Recovery (DR), and application stack is online. Migration must be initiated from the primary cluster only – this is the source cluster. You can run the command `kubectl or oc edit/patch <Name of DR plan>` and update Spec:PrimaryCluster to change current primary cluster details. A different Spec:PrimaryCluster in specifications and status indicates that migration is ongoing.

Figure 12-1 Migration initiated. Namespace resides on Cluster 1.



Following entities are updated during migration:

1. Application metadata - When migration is initiated from source cluster, latest snapshot of managed application's metadata is taken and tagged for restoration. This latest snapshot is replicated across peer clusters (target cluster) for restoring. Thereafter, application goes offline on the source cluster. On the target cluster, this latest snapshot is used for restoring application stack.

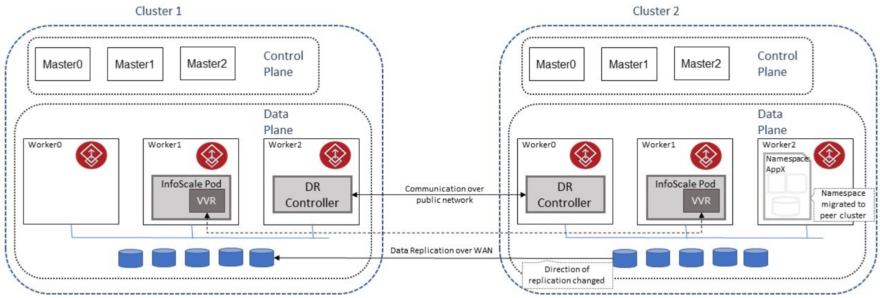
2. Application data - For stateful applications, you must have configured Data Replication CR and updated DisasterRecoveryPlan:Spec:DataReplicationPointer accordingly. Data Replication CR manages replication of application data from primary cluster to peer clusters (source to target). Currently, InfoScale Volume Replicator(VVR) is used for application data replication. When migration is initiated from the source cluster, the cluster roles are swapped. The proposed primary cluster assumes 'Primary' role whereas current primary cluster assumes 'Secondary' role.

3. DNS endpoints - The DNS custom resource updates and monitors the mapping for:

   - The host name to IP address (A, AAAA, or PTR record)
   - Alias to hostname or canonical name (CNAME)

   When migration is initiated, the DNS resource records are updated appropriately.

Figure 12-2 Migration complete. Namespace resides on Cluster 2.



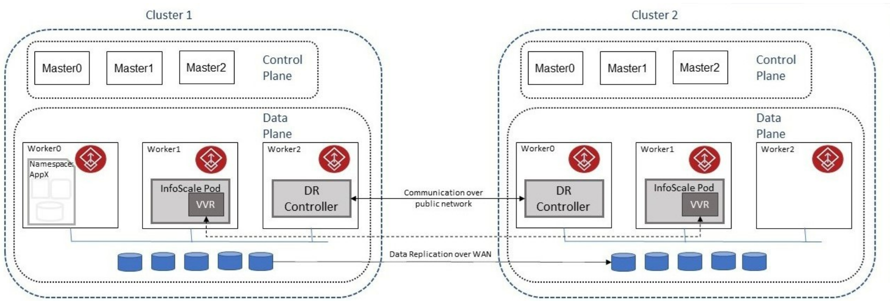
You can check intermediate transient states like BackupStatus, ScheduleStatus, RestoreStatus, and DataReplicationStatus attributes of Disaster Recovery Plan during migration. To check logs if migration is stuck, run `kubectl/oc logs -f --tail=100 deployments.apps/infoscale-dr-manager -n infoscale-vtas`. After migration is complete, these transient states are cleaned and Status:PrimaryCluster in Disaster Recovery Plan is updated to the new primary.

Takeover

You can initiate takeover on a secondary cluster when peer primary cluster is down or disconnected. This can be used in case of disaster when primary cluster abruptly goes down or is disconnected, and application stack is online on the primary cluster. To recover, you can initiate a takeover. Takeover must be initiated from the surviving target cluster(the secondary cluster) where you want to recover application instances. On the secondary cluster, you can run `kubectl or oc edit/patch <NameofDRplan>` and update `Spec:Force` and `Spec:PrimaryCluster` attributes. `Spec:PrimaryCluster` must be changed to current primary cluster details.

`Spec:Force` attribute is a part of the DisasterRecoveryPlan object. Whenever you initiate a takeover, `Spec:Force` must be set to true. After `Spec:Force` is set to true, do not edit DisasterRecoveryPlan object. As peer clusters are not connected, any changes in DisasterRecoveryPlan cannot be synchronized across all peer clusters. As soon as all clusters are connected, stale application instances on outdated primary cluster are removed. `Spec:Force` is also automatically reset to false when all clusters are connected, thus concluding takeover operations.

Figure 12-3 Before Takeover

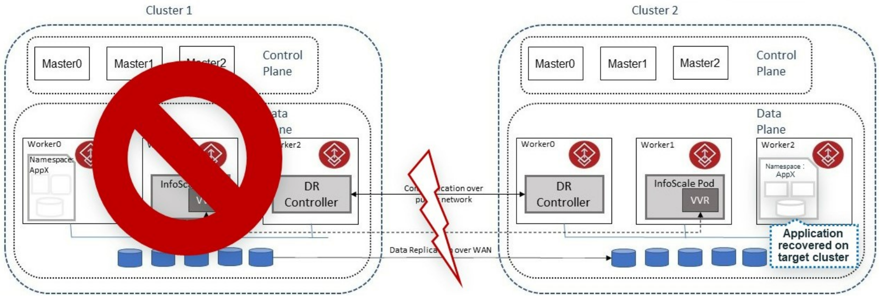


Following entities are updated during a takeover.

1. Application metadata: When a takeover is initiated on the target cluster, latest/recent snapshot of managed application's metadata from the primary cluster is tagged for restoration. This snapshot is used for restoring application stack.
2. Application data: For stateful applications, you must have configured DataReplication CR and updated `DisasterRecoveryPlan:Spec:DataReplicationPointer` accordingly. DataReplication CR manages replication of application data from the primary cluster to the peer clusters(source to target). When a takeover is initiated on the target cluster, the proposed primary cluster assumes 'Primary' role and is allowed read write on Volumes. When outdated primary cluster is connected, it joins as a 'Secondary' role.
3. DNS end points: The DNS custom resource updates and monitors the mapping for
   - The host name to IPaddress (A, AAAA, or PTR record)
   - Alias to hostname or canonical name(CNAME)

When a takeover is initiated, the DNS resource records are updated appropriately

Figure 12-4 After Takeover



You can check intermediate transient states like `BackupStatus`, `ScheduleStatus`, `RestoreStatus`, and `DataReplicationStatus` attributes of the Disaster Recovery Plan during takeover. To check logs if migration is stuck, run `kubectl/oc logs -f --tail=100 deployments.apps/infoscale-dr-manager -n infoscale-vtas`. After migration is complete, these transient states are cleaned and `Status:PrimaryCluster` in DisasterRecoveryPlan is updated to the new primary.

Chapter 13

Configuring InfoScale

This chapter includes the following topics:

Logging mechanism

InfoScale runs primarily as daemonsets on an OpenShift or a Kubernetes cluster. To access logs of a failed pod/container, the logs must persist beyond the lifecycle of the container. Containerized InfoScale generates logs as log files and container logs. Logs are helpful for debugging purposes. Log files generated by containerized InfoScale persist on the hostPath `/var/VRTS/log` of each host. You can access Container logs of running InfoScale pods/containers by using `oc logs` or `kubectl logs` command.

If DR Controller is configured, controller logs are also included in the Container logs. DR controller logs are independently generated on all peer clusters added in the Global Cluster Membership and hence, logs from all peer OpenShift or Kubernetes clusters must be collected.

To persist container logs beyond the lifecycle of the Container, following methods can be used.

EFK (ElasticSearch, Fluentd, Kibana) - EFK is the default logging stack on OpenShift. See the OpenShift documentation for configuring EFK.

Fluentd log collector - You can use Fluentd log collector to save the container logs at `/var/VRTS/log`. Fluentd runs as a daemonset on the OpenShift or Kubernetes cluster. Hence, it can collect logs generated at each node. On OpenShift or Kubernetes, Fluentd needs to run with privileged mode to access hostPath volumes.

Run the following command to enable this:

```
oc adm policy add-scc-to-user privileged -z fluentd
```

Create a file `fluentd-infoscale-spec.yaml`, and apply the following configuration by using oc command:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: infoscale-fluentd-config
  namespace: kube-system
data:
  fluent.conf: |
    <source>
      @type tail
      @id container-input
      read_from_head true
      format none
      path "/var/log/containers/infoscale*.log"
      pos_file "/var/log/infoscale.log.pos"
      tag infoscale.tail.*
    </source>
    <match infoscale.tail.**>
      @type file
      format json
      append true
      path /containerlogs/${tag[5]}
      <buffer tag>
        flush_interval 5s
      </buffer>
      <format>
        @type single_value
        message_key message
      </format>
    </match>

---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: fluentd
  namespace: kube-system
```

```
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: fluentd
  namespace: kube-system
rules:
- apiGroups:
  - ""
  resources:
  - pods
  - namespaces
  verbs:
  - get
  - list
  - watch

---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: fluentd
roleRef:
  kind: ClusterRole
  name: fluentd
  apiGroup: rbac.authorization.k8s.io
subjects:
- kind: ServiceAccount
  name: fluentd
  namespace: kube-system
---
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: fluentd
  namespace: kube-system
  labels:
    k8s-app: fluentd-logging
    kubernetes.io/cluster-service: "true"
spec:
  selector:
    matchLabels:
      name: fluentd
```

```
  template:
    metadata:
      labels:
        k8s-app: fluentd-logging
        kubernetes.io/cluster-service: "true"
        name: fluentd
    spec:
      serviceAccount: fluentd
      serviceAccountName: fluentd
      containers:
        - name: fluentd
#
# On Openshift, fluentd needs to run as privileged due to hostPath mounts
#
          securityContext:
            privileged: true

          image: fluent/fluentd-kubernetes-daemonset:v1-debian-elasticsearch
          env:
            - name: FLUENT_UID
              value: "0"
            - name: FLUENT_ELASTICSEARCH_SED_DISABLE
              value: "true"
          volumeMounts:
            - name: config-volume
              mountPath: /fluentd/etc/fluent.conf
              subPath: fluent.conf
            - name: container-logs
              mountPath: /var/log
            - name: hostpath-containerlogs
              mountPath: /containerlogs
#
# for docker containers (usual on vanilla kubernetes),enable below mountpoint
#            - name: varlibdockercontainers
#              mountPath: /var/lib/docker/containers
      volumes:
        - name: config-volume
          configMap:
            name: infoscale-fluentd-config
        - name: container-logs
          hostPath:
            path: /var/log
            type: Directory
```

```
        - name: hostpath-containerlogs
          hostPath:
            path: /var/VRTS/log/containers
            type: DirectoryOrCreate
#
# for docker containers (usual on vanilla kubernetes), enable below volume
#        - name: varlibdockercontainers
#          hostPath:
#            path: /var/lib/docker/containers
#            type: Directory
```

If you have configured DR, add the following to `fluentd-infoscale-spec.yaml`. You can add it at the end of this file.

```
<source>
@type tail
@id container-input
read_from_head true
format none
path "/var/log/containers/infoscale-dr-manager*.log"
pos_file "/var/log/dr-controller.log.pos"
tag infoscale.tail.*
</source>
```

This configuration enables Fluentd to log all Infoscale containers to the hostPath directory `/var/VRTS/log/`containers of each host.

Configuring InfoScale Oracle Data Manager (VRTSodm)

InfoScale Oracle Data Manager (VRTSodm) is offered as a part of InfoScale suite. With VRTSodm, Oracle Applications bypass caching and locks of the file system thus enabling a faster connection.

VRTSodm is enabled by the linking `libodm.so` with the Oracle Application. The I/O calls from Oracle Application are then routed through the ODM kernel module.

Following changes are needed to the Oracle database `yaml` file to enable it to run with InfoScale ODM:

1. Update the VxFS Data Volume (<vxfs pvc>) in the following code and add it to the `.yaml`.

Note: Oracle Container image requires the data volume to be mounted at `/opt/oracle/oradata`. This volume also needs to be writable by the 'oracle' (uid: 54321) user inside the container. VxFS data volume must be mounted at this path by using a PVC. To handle this permissions issue, the following `initContainer` can be used.

```
initContainers:
- name: fix-volume-permission
  image: ubuntu
  command:
  - sh
  - -c
  - mkdir -p /opt/oracle/oradata
    && chown -R 54321:54321 /opt/oracle/oradata
    && chmod 0700 /opt/oracle/oradata
  volumeMounts:
  - name: <vxfs pvc>
  mountPath: /opt/oracle/oradata
  readOnly: false
```

2. Add the following to your `.yaml` to disable DNFS.

```
args:
- sh
- -c
- cd /opt/oracle/product/19c/dbhome_1/rdbms/lib/ &&
 make -f ins_rdbms.mk dnfs_off && cd $WORKDIR &&
 $ORACLE_BASE/$RUN_FILE
```

3. Create a hostpath volume `devodm` in the `.yaml`, and mount at `/dev/odm`.

Note: On selinux-enabled systems (including OpenShift), the Oracle database container must be run as privileged.

---

4. Use the `libodm.so` that InfoScale provides. Run the following commands on the bastion/master nodes.

   - `oc/kubectl cp <infoscalepod>:/opt/VRTSodm/lib64/libodm.so.`
   - `oc/kubectl create configmap libodm --from-file libodm.so.`
   - Mount `libodm.so` inside the oracle container as under:

```
    - name: libodm-cmapvol
      mountPath: /opt/oracle/product/19c/dbhome_1/rdbms/lib/odm/libodm.so
      subPath: libodm.so

    volumes:
    - name: libodm-cmapvol
      configMap:
        name: libodm
        items:
        - key: libodm.so
          path: libodm.so
```

Run your .yaml on the bastion mode of the OpenShift cluster or the master node of the Kubernetes cluster.

Alternatively, copy the following content and create a new file `oracle-odm.yaml`.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: oracle-odm
  labels:
    app: oracledb
spec:
  replicas: 1
  selector:
    matchLabels:
      app: oracledb
  template:
    metadata:
      labels:
        app: oracledb
    spec:
      initContainers:
      - name: fix-volume-permission
        image: ubuntu
        command:
        - sh
        - -c
        - mkdir -p /opt/oracle/oradata && chown -R 54321:54321
          /opt/oracle/oradata && chmod 0700 /opt/oracle/oradata
        volumeMounts:
```

```
  - name: oracle-datavol
    mountPath: /opt/oracle/oradata
    readOnly: false
containers:
- name: oracle-app
  securityContext:
    privileged: true
  image:#replace this with the link for patched oracle container image
  imagePullPolicy: IfNotPresent
  # Modification to the args to disable dnfs before starting database
  args:
  - sh
  - -c
  - cd /opt/oracle/product/19c/dbhome_1/rdbms/lib/ && make -f
      ins_rdbms.mk dnfs_off && cd $WORKDIR && $ORACLE_BASE/$RUN_FILE
  resources:
    requests:
      memory: 8Gi
  env:
  - name: ORACLE_SID
    value: "orainst1"
  - name: ORACLE_PDB
    value: orapdb1
  - name: ORACLE_PWD
    value: oracle
  ports:
  - name: listener
    containerPort: 1521
    hostPort: 1521
  volumeMounts:
  - name: oracle-datavol
    mountPath: /opt/oracle/oradata
    readOnly: false
  - name: devodm
    mountPath: /dev/odm
  - name: libodm-cmapvol
   mountPath: /opt/oracle/product/19c/dbhome_1/rdbms/lib/odm/libodm.so
    subPath: libodm.so
volumes:
- name: oracle-datavol
  persistentVolumeClaim:
    claimName: oracle-data-pvc
- name: devodm
```

```
        hostPath:
          path: /dev/odm
          type: Directory
      - name: libodm-cmapvol
        configMap:
          name: libodm
          items:
          - key: libodm.so
            path: libodm.so
---
apiVersion: v1
kind: Service
metadata:
  name: ora-listener
  namespace: default
  labels:
    app: oracledb
spec:
  selector:
    app: oracledb
  type: NodePort
  ports:
  - name: ora-listener
    protocol: TCP
    port: 1521
    targetPort: 1521
```

Save the file.

Run the file on the bastion mode of the OpenShift cluster or the master node of the Kubernetes cluster to enable a faster connection.

Enabling user access and other pod-related logs in Container environment

OpenShift and Kubernetes clusters have an in-built logging mechanism. You can configure the `/etc/kubernetes/manifests/kube-apiserver.yaml` to include the following types of logs.

- Software upgrades and configuration file changes
- System (Virtual or Physical) boot and halt
- Process launches

- Non-normal process exits
- SELinux policy violations
- System login attempts
- Services starts and stops
- Container starts and exits

You can thus log all events related to InfoScale pods, secrets and config maps.

Note: After you configure these files, API server must be restarted. Hence, the server experiences a downtime. Ensure that you inform about the downtime to the user community. If the files are not correctly configured, the API server might not restart. The configuration must be performed by a competent Storage Administrator.

Add the following code to `/etc/kubernetes/manifests/kube-apiserver.yaml`to log all user login attempts to the InfoScale pods. The user name, time, and whether the attempt is successful or not is logged.

```
apiVersion: audit.k8s.io/v1 # This is required.

kind: Policy

rules

# Log pod/exec requests at RequestResponse level

- level: RequestResponse

namespaces: ["infoscale-vtas"]

resources:

- group: ""

resources: ["pods/exec"]

# Log everything else at Metadata level

- level: Metadata

omitStages:

- "RequestReceived"
```

Similarly, add the following code to log pods creation and deletion, config map changes, and secrets changes.

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

kind: Policy
```

```
omitStages:
- "RequestReceived"
rules:
#Log the Metadata of Pod changes in given namespace
- level: Metadata
resources:
- group: ""
resources: ["pods"]
verbs: ["create", "patch", "update", "delete"]
namespaces: [""] #Fill namespace
#Please use the appropriate namespace where
# the infoscale pods are deployed in the namespace tag
#For eg: namespaces["infoscale-vtas"]
#Log the Request body of configmap changes in given namespace
- level: Request
resources:
- group: ""
resources: ["configmaps"]
namespaces: [""] #Fill namespace
#Log the Request of secrets changes in given namespace
- level: Request
resources:
- group: ""
resources: ["secrets"]
namespaces: [""] #Fill namespace
```

Login attempts to an OpenShift cluster get recorded in `oauth-openshift- pod` logs. The log level must be 'debug'. You can run `oc edit authentications.operator.openshift.io` to change the log level to 'debug'.

On an OpenShift cluster, pods creation and deletion gets logged in `journalctl`. Run `journalctl no-pager` on all the worker nodes for information about pods creation and deletion.

To enable extended logging for all core InfoScale components like VxVM, VxFS, and VCS, set the `EO_COMPLIANCE` to enabled.

Note: You must enable `EO_COMPLIANCE` for your InfoScale deployment first. As a prerequisite, ensure that DNS is correctly configured or `/etc/hosts` is used to define IP address, full-qualified domain name(FQDN), and host name for the cluster node. Edit sds-operator deployment. Run the following command `oc/kubectl edit deployment -n infoscale-vtas <deployment_name>` . Ensure that you update `EO_COMPLIANCE` here to enabled as under.

```
name: EO_COMPLIANCE

value: enabled
```

After you edit and save the sds-operator deployment, the InfoScale sds operator restarts automatically. You have to manually restart infoscale-sds pods. Ensure that you restart one pod at a time. After a restarted pod is in a 'Ready' state, restart the next pod.

If you want to enable `EO_COMPLIANCE` on an OpenShift cluster by using OLM, run `oc edit subscription infoscale-sds-operator -n infoscale-vtas` and add the following to spec:

```
config:

env:

name: EO_COMPLIANCE

value: enabled
```

Chapter 14

Administering InfoScale on Containers

This chapter includes the following topics:

Adding nodes to an existing cluster

Complete the following steps to add nodes to an existing InfoScale cluster:

1. Ensure that you add the worker nodes to the Kubernetes cluster.
2. Run the following command on the master node to check whether the newly added node is Ready.

   `kubectl get nodes`

   Review output similar to the following

   ```
   NAME           STATUS  ROLES         AGE  VERSION
   worker-node-1  Ready   control-plane,
                                 master 222d v1.21.0
   worker-node-2  Ready   worker        222d v1.21.0
   worker-node-3  Ready   worker        222d v1.21.0
   ```

3. To add new nodes to an existing cluster, the cluster must be in a running state. Run the following command on the master node to verify.

   `kubectl get infoscalecluster -A`

   See the State in the output similar to the following -

   ```
   NAMESPACE   NAME                VERSION   CLUSTERID   STATE   AGE
   .
   .
   <Namespace> <Name of the
                InfoScale cluster> 9.1.2  <Cluster ID> Running 25h
   .
   .
   ```

4 Edit clusterInfo section of the sample `/infoscale-yamls-v9.1.2/kubernetes/cr.yaml` to add information about the new nodes.

In this example, worker-node-1 and worker-node-2 exist. worker-node-3 is being added.

Note: If you specify IP addresses, the number of IP addresses for the new nodes must be same as the number of IP addresses for the existing nodes.

```
metadata:
  name: < Assign a name to this cluster >
  namespace: < The namespace where you want to
                      create this cluster>



 spec:
  clusterID: <Optional- Enter an ID for this cluster.
               The ID can be any number between 1 and 65535 >
  isSharedStorage: true
  - nodeName: <Name of the first node>
    ip:
    -  <Optional - First IP address of the first node >
    -  <Optional - Second IP address of the first node>
    excludeDevice:
    - <Optional - Device path of the disk on the node that you want
                           to exclude from Infoscale disk group.>
  - nodeName: <Name of the second node>
    ip:
   -  <Optional - First IP address of the second node >
   -  <Optional - Second IP address of the second node>
    excludeDevice:
    - <Optional - Device path of the disk on the node that you want
                           to exclude from Infoscale disk group.>
  - nodeName: <Name of the third node>
    ip:
   -  <Optional - First IP address of the third node >
   -  <Optional - Second IP address of the third node>
    excludeDevice:
    - <Optional - Device path of the disk on the node that you want
                           to exclude from Infoscale disk group.>
.
```

```
.
.

 YOU CAN ADD UP TO 16 NODES.

  enableScsi3pr:<Enter True to enable SCSI3 persistent reservation>

  fencingDevice: ["<Hardware path to the first fencing device>",
                  "<Hardware path to the second fencing device>",
                  "<Hardware path to the third fencing device>", ]
  encrypted: false
  sameEnckey: false

  customImageRegistry: <Custom registry name /
                 <IP address of the custom registry>:<port number> >
```

5 Run the following command on the master node to initiate add node workflow.

```
kubectl apply -f /infoscale-yamls-v9.1.2/kubernetes/cr.yaml
```

6 You can run the following commands on the master node when node addition is in progress.

a. `kubectl get infoscalecluster -A`

See the State in the output as under. ProcessingAddNode indicates node is getting added.

```
NAMESPACE   NAME            VERSION   CLUSTERID STATE             AGE
.
.
<Namespace> <Name of the              <Cluster
          InfoScale cluster> 9.1.2  ID>       ProcessingAddNode 25h
.
.
```

b. `kubectl describe infoscalecluster -n <Namespace>`

Output similar to following indicates the cluster status during add node. The cluster is Degraded when node addition is in progress.

```
Cluster Name:  infoscalecluster-dev
  Cluster Nodes:
    Exclude Device:
      <Excluded device path 1>
      <Excluded device path 2>
    Node Name:  worker-node-1
    Role:       Joined,Master
    Node Name:  worker-node-2
    Role:       Joined,Slave
    Node Name:  worker-node-3
    Role:         Out of Cluster
   Cluster State:  Degraded
   enableScsi3pr:  false
   Images:
    Csi:
      Csi External Attacher Container:      csi-attacher:v3.1.0
```

7 Run the following command on the master node to verify if pods are created successfully. It may take some time for the pods to be created.

```
kubectl get pods -n infoscale-vtas
```

Output similar to the following indicates a successful creation.

```
NAME                                 READY STATUS  RESTARTS AGE
infoscale-csi-controller-35359-0     5/5   Running 0        12d
infoscale-csi-node-35359-7rjv9       2/2   Running 0        3d20h
infoscale-csi-node-35359-dlrxh       2/2   Running 0        4d21h
infoscale-csi-node-35359-dmxwq       2/2   Running 0        12d
infoscale-csi-node-35359-j9x7v       2/2   Running 0        12d
infoscale-csi-node-35359-w6wf2       2/2   Running 0        3d20h
infoscale-fencing-controller-
          35359-6cc6cd7b4d-l7jtc     1/1   Running 0        3d21h
infoscale-fencing-enabler-
                     35359-9gkb4     1/1   Running 0        12d
infoscale-fencing-enabler-35359-
                           gwn7w     1/1   Running 0        3d20h
infoscale-fencing-enabler-35359-
                           jrf2l     1/1   Running 0        12d
infoscale-fencing-enabler-35359-
                           qhzdt     1/1   Running 1        3d20h
infoscale-fencing-enabler-35359-
                           zqdvj     1/1   Running 1        4d21h
infoscale-sds-35359-
          ed05b7abb28053ad-7svqz     1/1   Running 0        13d
infoscale-sds-35359-
          ed05b7abb28053ad-c272q     1/1   Running 0        13d
infoscale-sds-35359-
          ed05b7abb28053ad-g4rbj     1/1   Running 0        4d21h
infoscale-sds-35359-
          ed05b7abb28053ad-hgf6h     1/1   Running 0        3d20h
infoscale-sds-35359-
          ed05b7abb28053ad-wk5ph     1/1   Running 0        3d20h
infoscale-sds-operator-
                 7fb7cd57c-rskms     1/1   Running 0        3d20h
infoscale-licensing-operator-
                756c854fdb-xvdnr     1/1   Running 0        13d
```

8 Run the following command on the master node to verify if the cluster is 'Running'

```
kubectl get infoscalecluster -A
```

See the State in the output similar to the following:

```
NAMESPACE   NAME                VERSION   CLUSTERID   STATE   AGE
.
.
<Namespace> <Name of the
             InfoScale cluster> 9.1.2  <Cluster ID> Running 25h
.
.
```

9 Run the following command on the master node to verify whether the cluster is 'Healthy'.

```
kubectl describe infoscalecluster <Cluster Name> -n <Namespace>
```

Check the Cluster State in the output similar to the following:

```
Status:
  Cluster Name:  <Cluster Name>
  Cluster Nodes:
    Node Name:    worker-node-1
    Role:         Joined,Master
    Node Name:    worker-node-2
    Role:         Joined,Slave
    Node Name:    worker-node-3
    Role:         Joined,Slave
  Cluster State:  Healthy
```

Removing nodes from an existing cluster

The remove node operation is supported only for shared storage.

It is not supported in the following cases:

- When other nodes (excluding those targeted for removal) in the cluster are in the 'out of cluster' state.
- During an upgrade.
- In the FSS environment.

Note: When performing this operation on OpenShift, replace `kubectl` with `oc` in the following procedure and run the commands on the bastion node instead of master node.

Complete the following steps to remove nodes from an existing InfoScale cluster:

1. Ensure that all the nodes (excluding the nodes that are being removed) must be in a Ready state. Run the following command on the master node to check whether the nodes are ready.

   ```
   kubectl get nodes
   ```

2. Ensure that nodes not being removed are in a Joined state. Run the following command on the master node and then review the output.

   ```
   kubectl describe <Name of the InfosScale Cluster> -n <Namespace>
   ```

   For example, if worker-node-3 is planned for removal, verify that worker-node-1 and worker-node-2 are in the Joined state by checking their Role status:

   ```
   Cluster Name: <Name of the InfosScale Cluster>
      Cluster Nodes:
        Node Name:    worker-node-1
        Role:         Joined,Master
        Node Name:    worker-node-2
        Role:         Joined,Slave
        Node Name:    worker-node-3
        Role:         Joined,Slave
   ```

3. The InfoScale cluster must be in a running state. Run the following command on the master node to verify.

   ```
   kubectl get infoscalecluster -n <Namespace>
   ```

   See the State in the output similar to the following:

```
NAME           VERSION   CLUSTERID      STATE     DISKGROUPS   STATUS    AGE
<Name of       9.1.2   <Cluster ID>   Running   <DG Name>    <Status>  3d13h
the InfoScale
Cluster>
```

4. Edit the clusterInfo section of the sample `/infoscale-yamls-v9.1.2/kubernetes/cr.yaml` to remove information about the nodes that are planned for removal.

5 Run the following command on the master node to initiate the remove node operation.

```
kubectl apply -f /infoscale-yamls-v9.1.2/kubernetes/cr.yaml
```

6 You can run the following command on the master node to check the InfoScale cluster state.

```
kubectl get infoscalecluster -n <Namespace>
```

See the state in the output as displayed below. ProcessingRemoveNode indicates the node is getting removed.

```
NAME       VERSION   CLUSTERID     STATE                 DISKGROUPS  STATUS   AGE
<Name      9.1.2  <Cluster ID>  ProcessingRemoveNode  <DG Name>   Healthy  3d13h
of the
InfosScale
Cluster>
```

7 Run the following command to verify if pods are being removed. It may take some time for the pods to get deleted.

```
kubectl get pods -n <Namespace> -o wide
```

8 Run the following command on the master node to verify if the cluster is 'Running' & 'Healthy’.

```
kubectl get infoscalecluster -n <Namespace>
```

Verify the output:

```
NAME                  VERSION   CLUSTERID     STATE     DISKGROUPS  STATUS    AGE
<Name of the
InfoScale cluster>  9.1.2   <Cluster ID>  Running   <DG Name>   Healthy   3d13h
```

Adding storage to an InfoScale cluster

After the existing storage is consumed, you can present new disks to the InfoScale cluster and increase storage capacity. However, these disks are not automatically discovered or added to the InfoScale cluster.

Prerequisites:

- InfoScale cluster must be 'Running'.
- Additional storage must be presented to the node and detected by the operating system.

Run the following command:

```
kubectl/oc annotate infoscalecluster <infoscalecluster name> -n
<infoScalenamespace> infoscale.veritas.com/storage-scale-up='enabled'
```

To monitor progress, run the following command:

```
kubectl/oc describe infoscalecluster <infoscale cluster name> -n
<infoscale namescape>
```

Review output similar to the following:

```
  Type    Reason                         Age
  ----    ------                         ----


  Normal  Storage Scale Up operation:  12m (x2 over 26m)

  Normal  Storage Scale Up operation:  3m34s (x2 over 18m)


 From               Message
 ----               -------
 infoscalecluster   Started
 infoscalecluster   Ended
```

Note: If the output of the above `oc describe` command has warning like "Adding new disks in dg failed" then refer to See "Troubleshooting when adding storage to InfoScale cluster" on page 291.

Managing licenses

While installing InfoScale in an OpenShift or Kubernetes environment, you apply licenses. You select the license edition and optionally enter the IP address of the VIOM server in your environment. With a single license of Enterprise edition on an OpenShift or Kubernetes cluster, you can form multiple InfoScale clusters.

The license server IP address can be updated to `license_cr.yaml` subsequently, if not entered while installing. Edit `<path>/license_cr.yaml` and enter IP address of the VIOM server for `licenseServer`. After updating the yaml, run `kubectl/oc apply -f <path>/license_cr.yaml` to apply the license with the updated IP address of the VIOM server.

After the IP address of the VIOM server is specified and the license is applied, you can check the status. Run `kubectl describe license <path>/license_cr.yaml`.

Review the `License Server Status` message with the following table. You can initiate appropriate action on the basis of the message.

Table 14-1 License status message with the meaning

Message..	Means..
Active	Connected successfully.
License Server is not registered with InfoScale	VIOM Server is not registered with InfoScale.
Subscription License has expired	Trial period of the license is over.
Subscription License not available on License Server	License is not available on the VIOM server to which the connection is successful.
Subscription License overused. Kindly check the usage limit.	Exceeded limits of the license edition.
Error getting License Server status	VIOM Server is successfully connected, but does not return any message to the licensing operator pod.
Connection Error : Not Connected with License Server	Licensing operator pod is unable to connect to the VIOM server on the specified IP address.

Monitoring InfoScale

With InfoScale installed in OpenShift or Kubernetes environments, monitoring tools like Prometheus, Grafana, and AlertManager are enabled for monitoring of your InfoScale installation. These tools are the monitoring mechanism that OpenShift and Kubernetes provide. Prometheus collects massive data at regular intervals and presents the data as 'Metrics'. You can use this data for monitoring of cluster components and analyze performance of your InfoScale installation.

Note: For a deported diskgroup, metrics is not visible.

On an OpenShift cluster, after you install InfoScale and configure InfoScale clusters, Prometheus monitoring is automatically installed.

On a Kubernetes cluster, following are the additional steps to install Prometheus

1. Check if helm is available on the Kubernetes cluster.
2. If it is available, go to step 4.

3 Run the following steps:

```
url -fsSL -o get_helm.sh
https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
```

```
chmod 700 get_helm.sh
```

```
./get_helm.sh
```

4
```
helm repo add prometheus-community
https://prometheus-community.github.io/helm-charts
```

```
helm repo update
```

```
helm install <RELEASE_NAME>
prometheus-community/kube-prometheus-stack
```

On an OpenShift cluster, following is the additional step

- Create/Edit `cluster-monitoring-config` in the `openshift-monitoring` namespace as under

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: cluster-monitoring-config
  namespace: openshift-monitoring
data:
  config.yaml: |
    enableUserWorkload: true
```

Run the following commands to verify:

whether pods are created `oc/kubectl get pods -A |grep infoscale`

whether Prometheus monitoring pods are successfully installed.

On an OpenShift cluster - `oc get pods -n {openshift-monitoring} | grep prometheus`

On a Kubernetes cluster - `kubectl get pods -n {namespace} | grep prometheus`

Complete the following steps to enable Prometheus.

1. Run the following command to know the REST server name:

   ```
   oc/kubectl get svc -n {Namespace where InfoScale is installed}
   |grep rest | awk '{print $1}'
   ```

   Note `infoscale-sds-rest-<numerical value>` from the command output.

2. Optionally, to collect metrics with TLS verification; a client certificate is required. Copy the following content into `prom-secret.yaml` and apply the file.

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: infoscale-prom-cert
  namespace: infoscale-vtas
spec:
  commonName: infoscale-prom
  usages:
  - client auth
  duration: 2880h0m0s
  issuerRef:
    kind: ClusterIssuer
    name: infoscale-cert-issuer
  renewBefore: 720h0m0s
  secretName: infoscale-prom-tls
```

3. Copy the following content into `podmonitor.yaml` and apply the file. Ensure that you update `serverName` .

```
apiVersion: monitoring.coreos.com/v1
kind: PodMonitor
metadata:
  name: infoscale-metrics
  namespace: infoscale-vtas
  labels:
    release: kube-prometheus
spec:
  selector:
    matchLabels:
      cvmmaster: "true"
  podMetricsEndpoints:
  - path: /infoscale/api/2.0/metrics
    port: rest-endpoint
    interval: 2m
    scrapeTimeout: 30s
    scheme: https
    tlsConfig:
      ca:
```

```
  secret:
    key: ca.crt
    name: infoscale-prom-tls
cert:
  secret:
    key: tls.crt
    name: infoscale-prom-tls
keySecret:
  key: tls.key
  name: infoscale-prom-tls
serverName: infoscale-sds-rest-<Cluster ID>
```

Configuring Alerts for monitoring InfoScale

You can configure various alerts for your InfoScale. To configure, you need to update and apply `alertmanager.yaml`.

The following table lists the alerts you can configure:

Table 14-2 Alert expressions with the messages

Alert with its expression Message
Volume/PVC size threshold `vxfs_filesystem_used_percentage {job="infoscale-vtas/infoscale-metrics",namespace="infoscale-vtas"}> <Enter the threshold value in percentage>` Volume size exceeds <threshold value> %.
Diskgroup-related alert `vxvm_diskgroup_free_bytes {job="infoscale-vtas/infoscale-metrics",namespace="infoscale-vtas"} < <Enter the value in bytes>` The diskgroup has <value> free bytes.
NodeNotReady `kube_node_status_condition {job="kube-state-metrics",condition="Ready",status="true"} == 0` Node is not ready.

Table 14-2 Alert expressions with the messages *(continued)*

Alert with its expression Message
DiskFailure `vxvm_disk_status {job="infoscale-vtas/infoscale-metrics",namespace="infoscale-vtas"} == 1` Disk failed.
DR-related alerts `vxvm_rvg_replication_status {job="infoscale-vtas/infoscale-metrics",namespace="infoscale-vtas"} == 0.0` Replication failed.

The expressions for the alerts with their messages are entered in the alert manager configuration yaml file.

Configuring alert manager on an OpenShift cluster

1. Be ready with the expression and its message for the alert you want to configure. See Table 14-2 on page 274.
2. Copy the following and save the file as alertmanager.yaml.

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  name: dr-alert
  namespace: <Namespace you want to monitor>
spec:
  groups:
  - name: dralert
    rules:
    - alert: <Name of the Alert>
      annotations:
        message: '<Alert Message>'
      expr: |
        <Alert Expression>
      for: 5m
      labels:
        severity: critical
        prometheus: openshift-monitoring/k8s
```

Note: Ensure that you update the alert message and expression.

3. Run the following command on the bastion node:

```
oc apply -f alertmanager.yaml
```

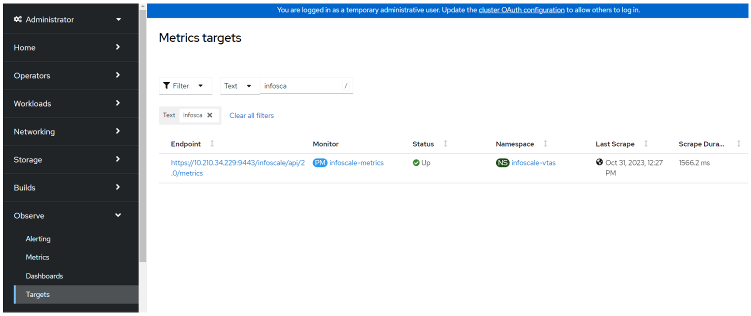
4. You can update and apply `alertmanager.yaml` for other alerts.

You can use the OpenShift monitoring features to configure alerts. See the following OpenShift help topics - Configuring the monitoring stack, Enabling monitoring for user-defined projects, and Enabling alert routing for user-defined projects.

Using OpenShift console to configure alerts

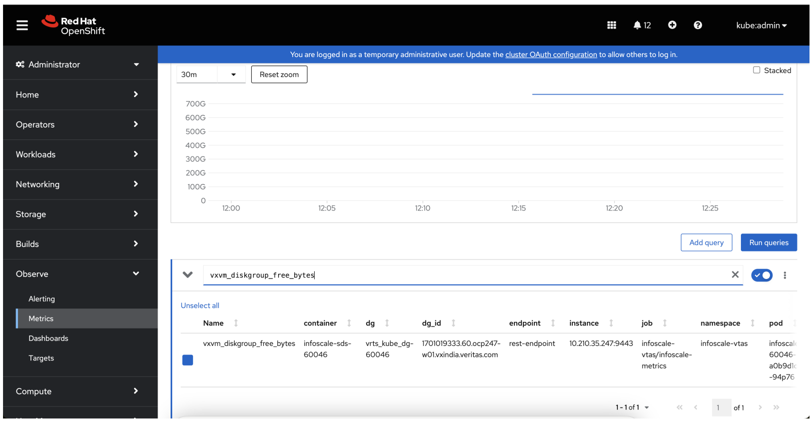
1. Access the OpenShift console. Select Operators > Installed Operators in the left frame.
2. Click InfoScale Cluster. Ensure that the cluster is in a 'Running' state.
3. Select Observe > Targets in the left frame.

4 Enter InfoScale as the search string as under. InfoScale deployment is displayed.



5 Select Observe > Metrics in the left frame.

6 In the screen that opens, you can add a new query or run saved queries.



7 Click Add query to configure a new query.

8 Assign a name to the query and select the Container.

9 You can then select diskgroup, job, namespace, or pods. Click the parameter and select the value for which you want to configure an alert.

10 Save the query. The saved query now gets listed. You can select a query in the main frame and click Run queries.

11 You can also See Table 14-2 on page 274. and enter a query to search.

After you save the alert, the alert is listed in Observe > Alerts.

Using OpenShift console to silence or stop configured alerts

1. Select Observe > Alerts in the left frame.
2. Select the Alert you want to silence.
3. Click the Actions menu (three vertical dots) at the end of the page.

   Alert details are listed.
4. Click Silence to stop the alert.

Configuring alert manager on a Kubernetes cluster

1. Be ready with the expression and its message for the alert you want to configure. See Table 14-2 on page 274.

2. Copy the following and save the file as alertmanager.yaml.

```
apiVersion: monitoring.coreos.com/v1
kind: PrometheusRule
metadata:
  annotations:
    meta.helm.sh/release-name: prometheus
    meta.helm.sh/release-namespace: default
  generation: 1
  labels:
    app: kube-prometheus-stack
    app.kubernetes.io/instance: prometheus
    app.kubernetes.io/managed-by: Helm
    app.kubernetes.io/part-of: kube-prometheus-stack
    app.kubernetes.io/version: 45.31.1
    chart: kube-prometheus-stack-45.31.1
    heritage: Helm
    release: prometheus
  name: prometheus-kube-prometheus-alertmanager-infoscale1.rules
  namespace: infoscale-vtas
spec:
  groups:
    - name: dralert.rules
      rules:
        - alert: <Alert>
          annotations:
            description: <Message>

            summary: <Summary>
          expr: |-
            <Expression>
          labels:
            severity: critical
```

Note: Ensure that you update the alert message and expression.

3 Run the following command on the master node:

```
kubectl apply -f alertmanager.yaml
```

4 You can update and apply `alertmanager.yaml` for other alerts.

Draining InfoScale nodes

You might need to drain InfoScale nodes for maintenance reasons (upgrading your software or hardware), poweroff and restart, or any other maintenance reasons. Pods running or scheduled on those InfoScale nodes are taken to other nodes of InfoScale cluster.

In such a case, InfoScale does not recommend draining more than one node at a time; due to constraints of infoscale-toolset pods.

Using InfoScale toolset

After you install InfoScale, InfoScale toolset yaml files are automatically downloaded. You can use these files to know the storage consumption or Volume rekey operations. The files are downloaded at `templates/`.

To know storage consumption

1 Edit `templates/storage/infoscale-storage-usage.yaml`. You can specify the namespace and cluster. To know the storage consumption of all namespaces and clusters, do not specify anything in `infoscale-storage-usage.yaml`.

2 Run the following command to apply `infoscale-storage-usage.yaml`.

```
kubectl/oc apply -f
/templates/storage/infoscale-storage-usage.yaml
```

3 Output similar to the following indicates success:

```
certificate.cert-manager.io/infoscale-tools-cert created
job.batch/infoscale-storage-usage created
```

4 Run the following command to list storage consumption:

```
kubectl/oc get po -n infoscale-vtas
--sort-by=.metadata.creationTimestamp -o
'jsonpath={.items[-1].metadata.name}' | xargs kubectl/oc logs -n
infoscale-vtas
```

Review output similar to the following:

```
NAMESPACE       CLUSTER         DISKGROUP
<namespace>     <cluster name> <diskgroup name>


STATE           VERSION    FREESIZE
imported shared <version> <Available disk space in GB>


 Disk information of Diskgroup-ID: <ID of the diskgroup>
 DISKS      DISKSIZE   MEDIATYPE
 <disk id> <disk size> <disk type>
.
.
.


.
.
.
ALL DISKGROUPS ARE LISTED
```

To enforce rekey operation on encrypted volumes

1. Run the following command to prepare PVC for enforced rekey operation:

   ```
   kubectl/oc annotate pvc <PVC name>
   infoscale.veritas.com/rekey="true" --overwrite
   ```

   The following command indicates that the PVC is ready for enforced rekey operation:

   ```
   persistentvolumeclaim/<PVC name> annotated
   ```

2. Run the following command to enforce rekey:

   ```
   kubectl/oc apply -f /templates/rekey/infoscale-rekey-ondemand.yaml
   ```

To schedule rekey operation on encrypted volumes

1. Edit `templates/rekey/infoscale-rekey-schedule.yaml` to add schedule for rekey operation.
2. Run the following command to apply the schedule:

   ```
   kubectl/oc apply -f /templates/rekey/infoscale-rekey-schedule.yaml
   ```

   Following output indicates a successful application of the schedule:

   ```
   certificate.cert-manager.io/infoscale-tools-cert created
   cronjob.batch/infoscale-rekey-schedule created
   ```

To perform a diskgroup rekey operation

◆ Run the following command:

   ```
   kubectl/oc apply -f /templates/rekey/infoscale-rekey-schedule.yaml
   ```

Note: After Disaster Recovery (DR) configuration, if rekey is performed on a Volume; the parameter `vxvm.attr.encwvek` is not replicated from the primary to the secondary cluster. Although `vxvm.attr.encwvek` is different on the secondary cluster, Disaster Recovery (DR) operation takes place successfully.

Changing the peerinact value in a cluster

The LLT peerinact parameter determines the amount of time InfoScale waits before declaring a node as faulted. Increasing the peerinact value allows for more tolerance for network outages, especially in environments that have known higher network latencies.

The default peerinact value may not be suitable for all environments. You may have to adjust this parameter as per your specific network requirements to ensure an optimal performance. Set peerinact to a value greater than the longest expected network delay in your environment.

The peerinact value is set in hundredths of a second or centiseconds. The default value is 1600 or 16 seconds. The recommended value is 3200 or 32 seconds.

To modify the peerinact value

1. Sign in to a driver node as an admin user and overwrite the peerinact annotation value using the following command:

   ```
   # oc annotate infoscalecluster -n infoscale-vtas
   infoscalecluster-qa "infoscale.veritas.com/peerinact"="<value>"
   --overwrite
   ```

   Here, <*value*> is the desired peerinact value in seconds or minutes.

   For example, to set the peerinact value to 32 seconds, run the following command:

   ```
   # oc annotate infoscalecluster -n infoscale-vtas
   infoscalecluster-qa "infoscale.veritas.com/peerinact"="32s"
   --overwrite
   ```

   Note: Specify the peerinact value in seconds or minutes. The InfoScale Operator converts it into centiseconds and updates the llttab configuration file.

2. It may take some time for the changes to take effect in a running cluster. You can verify the changes using the following commands:

   ```
   # oc exec -ti -n infoscale-vtas
   infoscale-sds-21532-731e5d2fe3e92f-v67zp – lltconfig -T query |
   grep peerinact
   ```

   The output may resemble the following:

   ```
   peerinact = 6000
   udppeerinact = 700
   ```

   ```
   # oc exec -ti -n infoscale-vtas
   infoscale-sds-21532-731e5d2fe3e92f-v67zp – cat /etc/llttab | grep
   peerin
   ```

   The output may resemble the following:

   ```
   set-timer peerinact:6000
   ```

Note: For fresh InfoScale clusters, you can add this annotation with the desired value in the yaml before applying the yaml file. In case of OpenShift, from the web console, switch to the yaml view for adding or modifying this parameter value.

PV rebuild

Overview

The PV Rebuild feature enables users to restore Persistent Volumes (PVs) and Volume Snapshot Contents (VSCs) when reinstalling Kubernetes (K8s) or OpenShift (OCP) instead of performing an upgrade. This feature ensures data retention during an InfoScale cluster rebuild and is also useful in cases where the K8s/OCP setup is accidentally destroyed.

Enabling and disabling PV rebuild

By default, the pv-rebuild feature is enabled.

Check the PV rebuild status

To verify whether PV rebuild is enabled, run the following command:

```
# oc describe infoscalecluster -n infoscale-vtas
<infoscale_cluster_name> "infoscale.veritas.com/pv-rebuild"
```

Disable the PV rebuild feature

To disable the PV rebuild feature, use the following command:

```
# oc annotate infoscalecluster -n infoscale-vtas
<infoscale_cluster_name> "infoscale.veritas.com/pv-rebuild"="disabled"
-overwrite
```

Note: Ensure that the cluster ID and cluster name remain the same during the InfoScale cluster rebuild.

Restoring PVs and VSCs after cluster rebuild

1. Restore PVs and PVCs
   - After the InfoScale cluster rebuild, PVs will be in an available state.
   - To bind PVs to Persistent Volume Claims (PVCs), reapply the same YAML files used for PVC creation.
   - Once applied, the PVCs and PVs will transition to the Bound state.
2. Restore VSCs to Volume Snapshots (VS)
   - Modify the existing Volume Snapshot (VS) YAML file with the following change:

     ```
     volumeSnapshotContentName: <vsc_name>
     ```
   - Ensure that the VS and VSC names match correctly.

- Retrieve the names of all Volume Snapshot (VS) and Volume Snapshot Contents (VSCs) using the following command:

  ```
  # oc get vsc -A
  ```

Troubleshooting

This chapter includes the following topics:

Adding a sort data collector utility

A data collector utility is available on the InfoScale support portal. To know more about the utility, see https://sort.veritas.com/data_collectors/overview.

Complete the following steps

1. Download https://sort.veritas.com/data/vos/collectors/sort_linux_x64.sh to collect evidences of the pods in `infoscale-vtas`.
2. Run the script on the master node of the Kubernetes cluster or on the bastion node of the OpenShift cluster.

Collecting logs by using SORT Data Collector

For troubleshooting purposes, logs are most necessary, By using SORT Data Collector (`sortdc`), you can collect node-level and cluster-level logs and InfoScale-specific logs at the node and cluster level as well.

Collecting logs on OpenShift

1. Run `sortdc` on the bastion node. With a single instance running on a bastion node, all node-level and cluster-level logs are collected.
2. Run the following command on the bastion node to export cluster's `admin.conf` to `KUBECONFIG`:

   ```
   export KUBECONFIG=/etc/kubernetes/admin.conf
   ```

3. Run the `sortdc` tool. OpenShift node-level and cluster-level logs are saved at the location you specify while running the tool.

Collecting logs on Kubernetes

1. Run `sortdc` on the master node. With a single instance running on a master node, all node-level and cluster-level logs are collected.
2. Run the following command on the master node to export cluster's `admin.conf` to `KUBECONFIG`:

   ```
   export KUBECONFIG=/etc/kubernetes/admin.conf
   ```

3. Run the `sortdc` tool. Kubernetes node-level and cluster-level logs are saved at the location you specify while running the tool.

Run the following commands on every InfoScale SDS pod in the InfoScale cluster for InfoScale-specific node-level and cluster-level logs.

1. Login to the InfoScale SDS pod.
   - On an OpenShift cluster, run the following command on the bastion node:

     ```
     oc exec -it <InfoScale SDS PodName> -n <InfoScale namespace>
     -- /bin/bash
     ```

   - On a Kubernetes cluster, run the following command on the master node:

```
kubectl exec -it <InfoScaleDriverContainerPodName> -n
<InfoScale namespace> -- /bin/bash
```

2 Run the following commands to create and enable the swap file:

```
dd if=/dev/zero of=/var/log/swapfile bs=1024 count=1000000;
mkswap /var/log/swapfile;
swapon /var/log/swapfile;
```

3 Run `/opt/VRTSspt/VRTSexplorer/VRTSexplorer`.

Specify log location as `/var/log/` to collect explorer files at the host-mounted path.

4 Run the following commands to disable and remove the swapfile.

```
swapoff /var/log/swapfile;
rm -f /var/log/swapfile;
```

Note: Ensure that you run these commands on all InfoScale SDS pods.

Approving certificate signing requests (csr) for OpenShift

Sometimes entry into the shell of an OpenShift container returns an error message like

```
Error from server: error dialing backend: x509: certificate is valid...
```

This error message is due to pending certificate signing requests (csr) approval or expired certificate signing requests (csr). Complete the following steps to approve certificate signing requests (csr). You can also refer to OpenShift documentation to know about csr.

1 Ensure that entry into the shell returns an error on all pods of the node.

2 Run the following command to check for pending approvals.

```
oc get csr
```

3 Run the following command to approve pending csrs.

```
oc get csr -o go-template='{{range .items}}{{if not
.status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm
certificate approve
```

4 Try entry into the shell to ensure that it is unblocked.

Cert Renewal related

1. If Prometheus is configured and if after the external CA certificate is renewed, the pod monitor fails to come up. Delete the prometheus cert secret followed by deletion of all pods in the monitored namespace. Prometheus pods then come up.
2. If the following error message is logged:

   ```
   x509: certificate has expired or is not yet valid: current time
   <timestamp1> is after <timestamp2>
   ```

   - If you are using an External CA certificate and the certificate has expired. See 'Renewing with an External CA certificate' and renew the certificate.
   - If you are using a self-signed CA, delete the infoscale-CA secret and wait for 10 minutes. Delete the secrets of all the client certificates.
3. If one of the following error messages is logged:

   ```
   error:veritas:tls parseCertsCertificate
                   expired :/etc/vx/<component name>/certs/tls.crt
   ```

   or

   ```
    tls : bad certificate
   ```

   - If you are using an External CA certificate and the certificate has expired. See 'Renewing with an External CA certificate' and renew the certificate. If the certificate has not expired, delete all secrets of the client certificates.
   - If you are using a self-signed CA and the certificate has expired, delete the infoscale-CA secret and wait for 10 minutes. Delete the secrets of all the client certificates. If the certificate has not expired, delete the secrets of all the client certificates.

PVC deletions after PV rebuilds

After performing a Persistent Volume (PV) rebuild, if you delete the Persistent Volume Claims (PVCs) after the PVCs and PVs are in a bound state, it can lead to storage management issues.

To prevent a mismatch between PVC and PV mappings, you should re-deploy the InfoScale cluster before creating new PVCs. This ensures a clean environment and avoids conflicts.

Troubleshooting when adding storage to InfoScale cluster

Complete the following steps:

1. Check the sds container pods:

   ```
   oc get pods -n infoscale-vtas | grep -i sds | grep -v operator
   ```

2. Log in into one of the sds-container pods:

   ```
   oc exec -it <sds-container-name> -n infoscale-vtas -- bash
   ```

3. Check disk status:

   ```
   #/opt/VRTSvcs/bin/hacli -cmd 'vxdisk list'
   ```

   All the new disks need to be in online invalid state. If new disks are not visible, then run "vxdisk scandisks". For shared storage, this operation needs to be done only from one of the <sds-pod-container> pod. But for locally attached storage, this needs be done from all the nodes for which additional storage is presented and the disk is not visible in “vxdisk list”.

4. Re-try the storage scale up operation.

Known Issues

The following issues are observed during testing in an internal test environment. The issues have remained unresolved and you might encounter these issues. The issue is described and if a workaround exists to resolve the issue, it is mentioned.

Note: Workaround is a temporary solution to the issue. InfoScale is working towards fixing the issue.

Table 15-1 Issue description and workaround

Description	Workaround
Deployment fails because of issues related to kernel module load/unload. (VIKE-4603)	Reboot the worker node on which the SDS pod fails to come up.
Licensing Operator does not upgrade as part of the install plan during OLM upgrade from version 8.0.310 to 8.0.320 due to different channels for operators. (VIKE-4570)	Upgrade the Licensing Operator separately.
Deleting InfoScale pods by using `oc delete pods` or `kubectl delete pods` command might result in Configuration failure. (4045599)	To undeploy InfoScale, delete by using InfoScale CR procedure.
On a VMware Virtual Machine, deployment of InfoScale on OpenShift or Kubernetes fails if Disk UUID is not enabled. (4046388)	Enable Disk UUID before deployment. See Enabling disk UUID on virtual machines.
For space-optimized snapshots, if writes on volume are more than size of cache object size, a write error is observed. This leads to snapshot volume getting marked as INVALID after detaching the mirror. (4043239)	Use space-optimized snapshots only in cases where expected rate of data change is much smaller than actual data volume size.
When a file system is 100% full, and PVC resize is attempted, allocating space for the metadata or the config files that are required for file system resize might fail, causing PVC resize failure. (4045020)	Contact InfoScale support for system recovery.
Deployment of pods with PVC which are restored from a snapshot or are cloned from another PVC and is initiated in ReadOnlyMany(ROM) access mode fails. Deployment goes into CreateContainerError state. (4040975)	Set the following deployment parameters to True - Pod.spec.volumes.persistentVolumeClaim.readOnly and Pod.spec.containers.volumeMounts[x].readOnly.
Disk initialization performed by using the `vxdisksetup` command fails with the following error message - `VxVM vxdisksetup ERROR V-5-2-1120 node002_vmdk0_0: Disk is tagged as imported to a shared disk group. Can not proceed.`. (4045033)	Ensure that the disk does not belong to any other diskgroup. If it does not belong to any other diskgroup, the disk might have some stale metadata. Run `vxdiskunsetup` on the disk and try disk initialization again.
A message 'File missing or empty: /boot/grub/menu.lst' is displayed even after a successful disk initialization. (4039351)	Ignore the message.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
When majority of the nodes in a cluster go in a 'NotReady' state, fast failover (kube-fencing) panics nodes. InfoScale fencing panics rest of the nodes. Even after the cluster is back with majority of the nodes in a 'Ready' state, unfinished kube-fencing jobs continue to panic nodes. (4044408)	Manually delete the kube-fencing jobs till the cluster is up.
With a heavy workload, node goes in a 'NotReady' state and InfoScale pods are getting killed. (4044963)	OpenShift Container Platform (OCP) runs extra system pods which consume memory. With heavy workloads, pods are killed to clear memory. Try the following - ■ Place a resource cap on less important OCP system pods like Prometheus (OCP Monitoring service). See OpenShift documentation. ■ Set pod eviction thresholds and set Kube-reserved and System-reserved resources. Pods are evicted when resources available for the node fall below the limits specified. See OpenShift documentation. ■ Provision higher physical memory for the node.
When a back enclosure is disabled and enabled in a cvm-slave node, disk fails to attach back to the disk group. (4046928)	Login to the InfoScale SDS pod and run `/etc/vx/bin/vxreattach`.
After faulting a slave node, one or more volumes do not get mounted or existing volumes get unmounted inside application pod. (4044533)	Reschedule/restart the pod to mount the volume.
If a worker node is powered off or rebooted, the node goes into emergency shell and enters NotReady state, thus becoming inaccessible. (4053892)	Reinstall or reconfigure the control plane on the worker node. See OpenShift documentation.
After creating PVC in RWX mode, data written on an application pod running on one node is not accessible from an application pod scheduled on another node. (4046460)	See https://access.redhat.com/solutions/6153272 for the recommended solution.
In container form factor if public/private NICs to be used for LLT are bonded and the underlying bonded NICs have been configured on the same switch,then the worker nodes on which InfoScale is configured might panic randomly with a message - `kernel BUG at mm/slub.c:305!` . (4048786)	If NIC bonding is required for the LLT links, ensure that the underlying NICs are configured on different switches to avoid the kernel node panic, even though the crash has no functional impact. If private links are connected to the same switch, the bond mode must be Active-Backup

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
Disks from a node of the InfoScale cluster do not get added to the disk group - `vrts_kube_dg`. The following error message `V-5-1-18986 sal_map_devices: da_online failed with error 142 for SAL disk <disk_name>` is logged in `syslog` on the master node. On running `kubectl describe infoscalecluster -n infoscale-vtas`, Output indicates disk addition failure. (4055278)	Add these disks to the disk group manually from the InfoScale SDS pod.
During InfoScale deployment, InfoScale configuration is not complete on one of the nodes and the node remains in a 'Not ready' state (4047598)	Remove `cr.yaml` and deploy InfoScale again by using `cr.yaml`.
If InfoScale is undeployed on all nodes while retaining the disk groups, re-creating InfoScale cluster on some nodes and adding nodes to the cluster fails. (4047205)	Undeploy and re-create InfoScale cluster on identical number of nodes. To undeploy InfoScale on all nodes and re-create InfoScale clusters on some nodes, contact InfoScale support.
After restoring space-optimized snapshot to new PVC, mount on restored PVC may fail if the source snapshot volume is detached (4012858)	Try one of the following - ■ Use CSI space-optimized snapshot functionality for read-intensive applications. ■ Use full-instant snapshot or CSI clone functionality for write-intensive or read-write-update applications. ■ Manually set the values of the configurable parameters like cachesize to an appropriate value based on the application workload while creating CSI volumesnapshotclass object
CSI controller pod remains in 'Terminating' state in case of graceful node shutdown or power-off (4011482)	Try one of the following - ■ If a node must be kept shut down for certain period, to ensure availability, use the following command to drain the node before shutting it down: `kubectl drain <node_name> --force --ignore-daemonsets--delete-local-data` ■ f you intend to delete the node from the Kubernetes cluster, delete the node object. In such case, you need not drain the node manually.
CSI node pods does not get rescheduled on other worker nodes when its parent node is drained (4011384)	None

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
While restoring a snapshot, PVC goes into pending state after rebooting all nodes except the master node in the cluster (4014525, 4048825)	Delete the snapshot volume by using the `vxedit` command. Kubernetes automatically reattempts to create a volume snapshot again.
In case of storage failure, application IOs to the mountpoint inside container fails and pod goes into CreateContainerConfigError or Error state (4011219, 4014758, 4015259)	Manually restart the application pod after the storage failure is resolved.
Current application on the primary must not be deleted until it is clear that DR is possible. In some cases, DR fails and application gets deleted on the primary.(4047475)	Ensure that the peer clusters are connected and Data Replication is in a healthy state.
If all worker nodes go down at the same time, InfoScale availability configuration is lost (4050355)	After recovery, InfoScale configuration is re-created. It might take up to 20 minutes.
If applications on a cluster with Load Balancer configuration are migrated, Load balancer service appears in 'Pending' state if the target cluster's Load balancer IP addresses are different. (4051429)	If you are using Load Balancer service, use DNS custom resources to manage DNS endpoint mapping.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
Delete datarep operation goes in an unresponsive state and force delete in CR fails. (4050857) Delete datarep operation fails if secondary site is not available. (VIKE-3885)	Complete the following steps to clean up 1. Check DR controller logs to check which cleanup part is failing. 2. Delete DataReplication Custom Resource (CR) on all clusters by using kubectl or oc edit datarep <name> command and removing finalizer string `infoscale.veritas.com.datareplication/finalizer`. 3. Login to InfoScale cluster pod infoscale-sds-* on all clusters 4. Complete the following steps for InfoScale Volume Replicator (VVR) objects cleanup ■ Stop replication for relevant RVG ■ Delete secondary ■ Delete primary ■ Delete corresponding SRL volume 5. Complete the following steps for InfoScale Cluster Server (VCS) objects cleanup ■ Change cluster operation to RW ■ Offline VIPgroup and RVGShared service groups corresponding to Datareplication CR (service group names are shown in CR status) ■ Delete resources available in these service. groups ■ Delete service groups' dependencies if any ■ Delete VIPgroup and RVGShared service groups ■ Change cluster operation to RO See InfoScale Volume Replicator and InfoScale Cluster Server documentation for details.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
In case DR migrate fails to complete and running the command kubectl describe datareplication.infoscale.veritas.com/<datarep_name> returns a message vradmin migrate command failed. (4053632)	Complete the following steps 1 Run the command to know the RVG name kubectl get datareplications.infoscale.veritas.com <application Data Replication name> 2 Login to one of the Infoscale SDS pods. 3 Run the following command vxprint -g vrts_kube_dg-<Infoscale cluster ID> <RVG name > 4 Review the output. If tutil is set to 'CONVERTING' , run the following command vxedit -g vrts_kube_dg-<Infoscale cluster ID> -f set tutil0="" <RVG name> tutil is cleared and DR migration completes
Kube-fencing is not functional when REST service and InfoScale operator pod is not running. (4054545)	None
In VVR environment, vradmin migrate might fail with the following error message VxVM VVR vxrvg ERROR V-5-1-1617 giving up: utility fields must be cleared by executing: vxedit -f set tutil0="" <rvg> (4057713)	Run the following command - vxedit -f set tutil0="" <rvg> to clear tutil on the RVG and retry the 'vradmin migrate' operation.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround

During cluster configuration or while adding new nodes on OpenShift or Kubernetes, node join might fail if the disks in the cluster have old disk group records from previous deployments. Output similar to the following indicates old disk group records.
 	
`vxvm:vxconfigd[238047]: V-5-1-11092 cleanup_client: (Disk in use by another cluster) 223`
`esxd05vm06 kernel: VxVM vxio V-5-0-164 Failed to join cluster infoscale_22670, aborting :`
 	
(4057178)
	Reset the cluster ID on the disks of joiner node, in order to allow the node to join.
	1 Check cluster ID on existing/already-joined nodes in cluster: Run
	`vxdisk list node000_vmdk0_9 | grep -i cluster`
	Cluster ID is returned in the output.
	2 If the node join fails, verify if cluster ID is different on the joiner node using same command: Run
	`vxdisk list node001_vmdk0_5 | grep -i cluster`
	A different Cluster ID is returned in the output.
	3 Change the cluster ID to match it with existing nodes in cluster.
	`/etc/vx/diag.d/vxprivutil set /dev/vx/dmp/node001_vmdk0_5 hostid=<Cluster ID returned in the first command>`
	Ensure that disks belong to same disk group. Consult InfoScale Technical support.

Stale InfoScale kernel modules might be left over after undeploying InfoScale cluster on OpenShift or Kubernetes.(4042642)	Before deploying InfoScale on OpenShift or Kubernetes, check if any stale InfoScale kernel modules (`vxio/vxdmp/veki/vxspec/vxfs/odm/glm/gms`) are loaded. If stale modules from old deployments are still loaded, reboot all worker nodes and then proceed with the InfoScale deployment.
LLT tools is unable to detect duplicate InfoScale cluster id on OpenShift.(4057800)	Re-deploy InfoScale CR to avoid duplicate cluster id match.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
RLINK detach is observed while replication autosync is in progress between the primary and secondary clusters. (VIKE-1290)	Run `kubectl describe datarep datareplicationName` to check datareplication status. If `ReplicationStatus` is stopped, Edit `datareplication` and set attributes `force: true` and `replicationState: stop` in remoteClusterDetails. Again, edit `datareplication` and set attributes `force: false` and `replicationState: start` in remoteClusterDetails.
On a Kubernetes cluster, tcp traffic reaches the node but does not get forwarded to the pod. (VIKE-1108)	Disable tx and rx offload on all nodes for calico tunnel device.
Migration or takeover fails if the namespace being backed up pre-exists on the target cluster with different SCC (Security context constraints)- related annotations like uid-range, supplemental-groups, and seLinuxOptions.(VIKE-1277)	None
Takeover operation fails if it is initiated when 'Disaster Recovery plan' is configured before 'Data replication configuration' is complete on the primary cluster. (VIKE-1294)	Wait for the following error in Disaster Recovery Plan CR status to go away before attempting takeover - metadataSyncStatus: Metadata backup transfer failed for all peer clusters or Wait for data replication CR status to be consistent up-to-date before applying Disaster Recovery plan CR as mentioned in 'Configuring Data Replication'.

On OCP 4.9, `nfd-worker` pods are not getting created. (VIKE-909)	Run `oc edit scc nfd-worker` on the bastion node, change
	`users:`
	`- system:serviceaccount::nfd-worker`
	to
	`users:`
	`- system:serviceaccount:`
	`<Name of the namespace>:nfd-worker`
	Wait for some time for the nfd-worker pods to be in the 'Running' State.

Table 15-1 Issue description and workaround *(continued)*

<table>
<tr><th>Description</th><th>Workaround</th></tr>
<tr><td>If multiple Data Replication plans for Disaster Recovery are created in parallel, the initial VVR synchronization is slow due to the locking that needs to be acquired in VVR for each of the Data Replication plan. (VIKE-1505)</td><td>To create multiple Data Replication plans, create the subsequent plans after the current Data Replication plan's initial synchronization is complete and status is consistent-up-to-date.</td></tr>
<tr><td>When UEFI secure boot is enabled, only the signed kernel modules that are authenticated by using a key on the system key ring can be successfully loaded. Hence InfoScale kernel modules might fail to load with UEFI secure boot. (VIKE-1578)</td><td>Disable UEFI secure boot. See the relevant documentation for details.</td></tr>
<tr><td>In the following conditions, addition of disks to a disk group in an already configured InfoScale cluster fails with the error message
<code>VxVM vxdg ERROR V-5-1-559</code>
<code>Disk node000_vmdk0_1: Name is already used</code>
1 Existing disk group is imported during cluster creation
2 Additional storage is presented to the cluster
3 Sequence of the node name in the CR is changed.
(VIKE-2598)</td><td>None</td></tr>
<tr><td>If thin-provisioned LUNs are configured, a mismatch in the total size and free size of the Disk Group might be observed. (VIKE-2497)</td><td>None</td></tr>
<tr><td>After DR takeover is completed and the old primary is up, VVR failback synchronization of the data is initiated from the new primary site to the new secondary site. This data synchronization might be slow or the system might be unresponsive resulting in the VVR failback to be incomplete. (VIKE-2634)</td><td>1 Pause all data replications on the primary site by setting the <code>replicationState</code> to <code>pause</code> in the data replication custom resources. Perform a sequential VVR failback synchronization operation then by setting the <code>replicationState</code> to <code>resume</code> on only one of the data replication custom resources, that is the active data replication.
2 After the VVR failback synchronization is complete on the only active data replication, resume VVR failback synchronization on the next paused data replication. Complete VVR failback synchronization on all the configured data replications.</td></tr>
</table>

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
Sometimes after a cluster fault, InfoScale nodes join the cluster back, infoscale-sds pods go in running state but the InfoScale disk group remains in a deported state.(VIKE-2666)	Login to any of the InfoScale-sds pods. ■ Run `/opt/VRTSvcs/bin/hagrp -clear DISK_GROUP` to clear the fault on `DISK_GROUP` and `/opt/VRTSvcs/bin/hagrp -online DISK_GROUP -any` to get `DISK_GROUP` online. ■ If the service group fails to be online, run `vxdg -s import <diskgroup_name>` to manually import the disk group. ■ If the disk group import fails with an error message `quorum lost`, reboot all InfoScale nodes in the cluster.
After InfoScale is deployed, cluster phase does not get updated and state is 'Degraded', even when it is Healthy. (VIKE-2586)	Restart InfoScale operator by deleting the pod.
If stale Volumes are present on the system, undeploying InfoScale is unresponsive with SDS pods in a terminating state (VIKE-2635)	Reboot all worker nodes. You can reboot sequentially, if applications in the cluster might be impacted due to a simultaneous reboot of all nodes.
Sometimes a primary cluster fault event might occur after a PVC resize operation on the primary cluster, followed by a takeover operation carried out from secondary cluster. In that case, when the old primary joins back the cluster membership, some Volumes on the old primary are in a `NEEDSYNC` state. The `NEEDSYNC` state is indicated by running `vxprint` on the old primary infoscale-sds pod. (VIKE-2917)	Run `vxrecover -s` from an infoscale-sds pod. The state changes to `Active`.
During upgrade, sometimes InfoScale sds pod remains in a terminating state. (VIKE-3003)	Reboot the node where the pod is stuck in a terminating state.
Error observed in Persistent Volume Claim addition in an application namespace if one of the worker nodes is down with the following error message in data replication status. `VxVM vxassist ERROR V-5-1-2127 Plex srl_4x0g2eewpu3b6m30ay71-02 should be enabled` (VIKE-4193)	Data replication status returns to normal state after the worker node restarts.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
When all the worker nodes are down, kubectl / oc get shows the incorrect InfoScale cluster status.(VIKE-4149)	Whenever any worker pod is 'Ready', InfoScale status is updated.
When an old primary cluster rejoins after a takeover operation, Volume number mismatch data replication config error is observed as under - Output of kubectl/oc describe datarep <datarep name> Replication Config Errors: <IP address>: volume-number mismatch, Primary-Primary configuration (VIKE-4151)	Run the following steps - 1 On the new primary, stop replication by editing the datarep CR. 2 On old primary(secondary), run the following commands on any infoscale-sds-* pod ■ vxvol -g <dg_name> -f dis <vol_name> ■ vxrvg -g <dg_name> makesecondary <rvg_name> ■ vxrlink -g <dg_name> -f start <rlink_name> (where rlink_name can be found by - vxprint | grep rlk_NewPrimaryVIP_rvgName) 3 On the new primary, start replication by editing the datarep CR.
Data replication is stuck in an inconsistent state while logging to DCM (VIKE-4369)	Run kubectl/oc get datarep -o wide. If the output indicates inconsistent | logging to DCM (needs dcm resynchronization) for a data replication plan, edit the corresponding datarep cr to stop replication. Wait for the replication to stop. Restart replication by editing the same datarep cr. The repstatus subsequently is replicating (connected).
While adding or removing disks to Storage Arrays without AVID support, applications might not come up after a node reboot. (VIKE-4139)	Contact InfoScale support.
After a node reboot, the toolset pod does not come online if the Volumes are in a 'Sync' state. (VIKE-4434)	Run the following command to change the state of the Volume from 'Sync' to 'Active '. vxvol -g <disk group name> -f resync <name of the Volume>
In a DR setup after a node reboot, if the InfoScale SDS pod on that node is not in a 'Ready' state and the cluster CR status is 'Degraded' on either primary or secondary, applications running on that node might face issues in performing IO operations. (VIKE-4428)	Reboot that node again and check the IO operations. If IO operations issues persist, reboot all nodes in the cluster together.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
PVC creation on a cluster with cri-o container engine fails with the following error message - `VxVM vxassist ERROR V-5-1-1401 Your license does not include support for mirroring. Cannot create volume with requested attributes` (VIKE-4367)	Run `crictl images` on the worker nodes of the cluster to verify if InfoScale images are present. If NOT present, see https://access.redhat.com/solutions/5350721 without the 'node reboot' section for the resolution.
If cluster nodes reboot or get powered off in a cascaded manner, application pods might not failover to a running node. (VIKE-4375)	This issue might be due to Storage connectivity. If Storage is connected and this issue is observed - 1 Identify the node where application pod is scheduled after failover. 2 Login to InfoScale SDS pod running on that node and run `vxprint -uH | grep LDISABLED` to identify the Volume which has a LDISABLED attribute. 3 Run `vxvol -g <DG_NAME> -f enable <VOL_NAME>` to remove that attribute.
Deleting application pod operation may get stuck, when the file system provisioned through Persistent Volume Claims (PVC) consumed by the application pod is in a disabled state. (VIKE-4514)	Login to the node where the application pod is scheduled and manually unmount all the mount points that has file system disabled.
While upgrading InfoScale Stack from 8.0.300 to 8.0.310, the SDS driver pod is stuck or does not boot for longer duration than expected. (VIKE-4523)	Check cluster start logs using the below command: `oc -n infoscale-vtas exec -it infoscale-sds-62453-a8d04dbf0095f7bc-wmgtx -- tail -f /var/VRTSvss/vss.log` If you see error messages or abnormal number of re-attempts for starting cluster, then reboot the corresponding worker node where SDS pods resides.
While upgrading OCP/orchestrator platform, the process may not progress and there is a mismatch between the version of worker node operating system and that of the selector. (VIKE-4522)	Ensure that the NFD label sync on the nodes is complete. Restart the InfoScale Operator pod.

Table 15-1 Issue description and workaround *(continued)*

Description	Workaround
Application pod is stuck in `init/ContainerCreating` state. Check error in CSI node log of the node where the application pod is scheduled using the command: `oc logs <csi-node-pod> -n infoscale-vtas -c infoscale-csi.` Following error is displayed. "Run: Error occurred while calling method" method=/csi.v1.Node/NodePublishVolume error="rpc error: code = Internal desc = Error occurred node staging is not done"" (VIKE 4532)	Delete the application pod. When the new pod is scheduled, it calls NodeStage and NodePublish.
The diskinfo field is missing from the InfoScaleCluster status. (VIKE-5119)	To obtain the disk group details, including diskinfo, use infoscale-storage-usage.yaml as specified in the "Using InfoScale toolset" section. See "Using InfoScale toolset" on page 280.
OLM certificate is expired. (VIKE-5415)	If you come across any instances of expired OLM certificates, restart the InfoScale operator.

InfoScale workloads on OpenShift can fail to read/write to shared RWX volumes because OpenShift assigns unique SELinux MCS labels to each pod. When replicas mount the same volume with different labels, SELinux denies access. This issue is well-documented in Red Hat and IBM storage guidance, where mismatched SELinux contexts are a common cause of RWX mount failures. (VIKE-5347)	Set a fixed SELinux label in the pod or container securityContext.seLinuxOptions so all replicas use the same SELinux context. This prevents MCS mismatches and restores proper RWX access.
	
	`securityContext:`
	`    seLinuxOptions:`
	`        level: "s0:c447,c946"`
	
	Refer to:
	
	SELinux options
	
	Security context constraints
	
	Assign SELinux labels to a Container
	
	Considerations for mounting read-write many (RWX) volumes

Table 15-1 Issue description and workaround *(continued)*

<table>
<tr><th>Description</th><th>Workaround</th></tr>
<tr><td>Clone volumes in the DiskGroup require serialized splitting before DataRep can move forward. This process depends on storage performance and may take several minutes. Allow the included script to complete the split operations before applying DataRep. (VIKE-5455)</td><td>This script performs continuous background monitoring of the DG and triggers clone splitting when sync is complete.

Download infoscale-tools-v9.1.2.tar.gz from Download center.
If you are planning to migrate storage, relocating PVCs, or performing DR setup run the below script from infoscale-tools-v9.1.2 after applying DataRep.
<code>nohup ./volume-clone-split.sh >
/var/log/clone-split.log 2>&1 &</code>
Wait for script to finish; splitting clones must be fully removed before proceeding, check logs.
<code>tail -f /var/log/clone-split.log</code>
Below are sample logs indicates script is processing correctly:
<code>[INFO] | No clone snapshots
found in DG=vrts_kube_dg-35322
[INFO] | Clone snapshots exist but sync
NOT complete yet
[INFO] | Splitting clone volume:
clone_xxxxx
[INFO] | Split completed for:
clone_xxxxx</code>
Once the split has completed , user can proceed with applying DataRep.
</td></tr>
<tr><td>When upgrading from InfoScale 8.0.400 to 9.1.2, the preflight check may fail if it detects volumes in a DETACHED state.</td><td>In the SDS container pod, check if any volume is in a DETACHED state by running:<code>vxprint -ht | grep -i detach</code>

If a volume is detached, manually start it using the following command:<code>vxvol -g <diskgroup_name> start <vol_name></code></td></tr>
</table>

Limitations

Following are the functional limitations due to external factors.

Table 15-2 Limitation description and workaround

Description	Workaround
Users can create a maximum of 32 snapshots for a volume. (VIKE-4605)	None
When DataReplication for DR is configured for a Galera 3 pod cluster, VVR performs a full synchronization instead of a smartsync (4051639)	None. Wait for initial full synchronization to complete before triggering the migration.
Remove node is not supported for FSS storage.	To remove node, contact InfoScale Technical support.

InfoScale modules fail to load on Kubernetes if SELinux is enabled on the worker nodes. (VIKE-1028)	Perform the following steps
	1 Run the following commands on the worker nodes to enable the tunables.
	`setsebool -P container_manage_cgroup on`
	`setsebool -P domain_kernel_load_modules 1`
	2 Set `selinux-enabled: true` in `/etc/docker/daemon.json`.
	3 Run the following command on the worker nodes to reboot.
	`systemctl daemon-reload && systemctl restart docker`

Replacing disks or replacing coordination points without unconfiguring fencing is not supported. (VIKE-2563)	Reconfigure the InfoScale cluster with the new coordination points.
During migration of an application or before takeover of an application, if PVCs and PVs are added to its namespace, the PVCs and PVs might not successfully restore on the secondary site.	None
Upgrades are not supported on two-node FSS configurations using majority-based fencing. If the node with nid 0 is selected for the upgrade, the remaining node is fenced out, causing the Disk Group (DG) to enter a DISABLED state and the upgrade process to hang. (VIKE-5806)	None