Fabric Services System

Release 25.8

Software Installation Guide

3HE 21933 AAAA TQZZA
Edition: 1
August 2025

Nokia is committed to diversity and inclusion. We are continuously reviewing our customer documentation and consulting with standards bodies to ensure that terminology is inclusive and aligned with the industry. Our future customer documentation will be updated accordingly.

Table of contents

1 About this document

This document describes how to deploy the Fabric Services System software and the required software components such as the Kubernetes cluster and storage volumes.

This document is intended for network technicians, administrators, operators, service providers, and others who need to understand how to install and provision the Fabric Services System for deployment.

Note: This document covers the current release and may also contain some content that will be released in later maintenance loads. See the *Fabric Services System Release Notes* for information about features supported in each load.

1.1 What's new

This section lists the changes that were made in this release.

Table 1: What's new in Fabric Services System 25.8

Description	Location
The Fabric Services System base OS image has been upgraded to Rocky 9.6.	Preparing the Fabric Services System virtual machine nodes Creating the VM on bridged networks on KVM
Updated release-specific references	Creating the VM on VMware vSphere Software upgrade and rollback Preparing for software upgrade Performing a software upgrade

1.2 Precautionary and information messages

The following are information symbols used in the documentation.

DANGER: Danger warns that the described activity or situation may result in serious personal injury or death. An electric shock hazard could exist. Before you begin work on this equipment, be aware of hazards involving electrical circuitry, be familiar with networking environments, and implement accident prevention procedures.

WARNING: Warning indicates that the described activity or situation may, or will, cause equipment damage, serious performance problems, or loss of data.

Caution: Caution indicates that the described activity or situation may reduce your component or system performance.

Note: Note provides additional operational information.

Tip: Tip provides suggestions for use or best practices.

1.3 Conventions

Commands use the following conventions:

- **Bold** type indicates a command that the user must enter.
- Input and output examples are displayed in `Courier` text.
- An open right angle bracket indicates a progression of menu choices or simple command sequence (often selected from a user interface). Example: **start** > **connect to**
- Angle brackets (< >) indicate an item that is not used verbatim. For example, for the command **show ethernet <name>**, **name** should be replaced with the name of the interface.
- A vertical bar (|) indicates a mutually exclusive argument.
- Square brackets ([ ]) indicate optional elements.
- Braces ({ }) indicate a required choice. When braces are contained within square brackets, they indicate a required choice within an optional element.
- *Italic* type indicates a variable.

Examples use generic IP addresses. Replace these with the appropriate IP addresses used in your system.

2 Installation overview

This chapter describes the Fabric Services System components, the requirements for these components, and provides an overview of the installation process.

2.1 Supported deployment models

The Fabric Services System is deployed on one, three, or six Fabric Services System nodes. The Fabric Services System is deployed as an application on virtual machine servers. In a deployment with six nodes, the Fabric Services System can be deployed with the Digital Sandbox.

Related topics

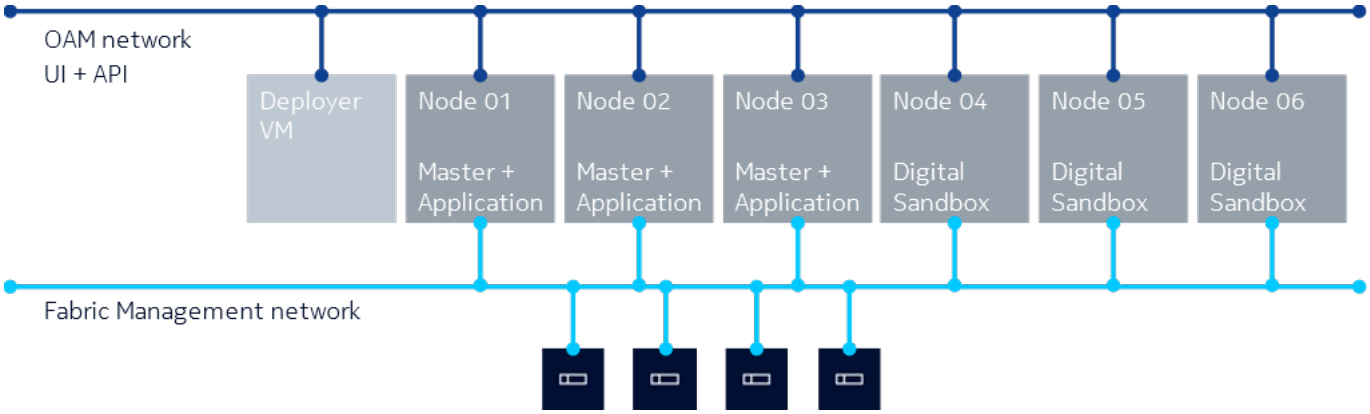
Fabric Services System node requirements

2.1.1 Networking for the Fabric Services System nodes

Nokia recommends that you use two different networks for the Fabric Services System nodes:

- Fabric Services System OAM interface
  This interface is used to access the UI and the API of the Fabric Services System. This is also the network through which the deployer VM reaches the node.
- Fabric management interface
  This interface is used to communicate with the management interfaces of the fabric (for example, SR Linux devices). This interface is where the Fabric Services System exposes its DHCP and ZTP services.

Figure 1: Deployment with six Fabric Services System nodes, with Digital Sandbox enabled



2.1.1.1 DHCP and ZTP

The Fabric Services System deployment can serve as a DHCP service for the fabric devices using one of the following methods:

- all the management interfaces of the fabric devices are connected to the same Layer 2 network as the DHCP interface of the Fabric Services System
- as the DHCP relay

2.1.2 Fabric Services System deployer

The Fabric Services System deployer is a virtual machine (VM) that is used to provision a Kubernetes environment and install Fabric Services System microservices on the Fabric Services System nodes.

It contains all the software packages, container images, and Helm charts that are used by the deployer script to install the entire Fabric Services System environment.

The Fabric Services System deployer VM:

- installs a Kubernetes cluster on the Fabric Services System nodes in your environment
- configures the nodes on the cluster for local storage
- labels the nodes to prepare them for Fabric Services System software deployment
- installs Fabric Services System microservices using Helm charts and container repositories hosted in the deployer
- provides technical support tools and backup and restore tools
- initializes the self-signed certificates used by the system used for the northbound services
- initializes the self-signed certificates use with managed nodes

Caution: To maintain the security of the Fabric Services System deployment, ensure that access to the deployer host and the deployer VM are restricted and password-protected.

Note: The deployer VM must remain powered on and must be reachable by the Fabric Services System cluster throughout its life cycle, as it serves as a container registry for all the Fabric Services System components. Additionally, the backup and restore tools and the technical support scripts are hosted on the deployer VM and must be executed from the deployer VM.

2.1.3 Fabric Services System nodes

The Fabric Services System nodes are the VMware vSphere-based or KVM-based VMs that host the Kubernetes environment on which the Fabric Services System application and Digital Sandbox are run.

The following deployment models are supported:

- an environment with one node, which hosts only the Fabric Services System application for very small scale deployments and does not support the use of Digital Sandbox
- an environment with three nodes, which hosts only the Fabric Services System application and does not support the use of Digital Sandbox

- an environment with a minimum of six nodes, which hosts both the Fabric Services System application and supports the use of Digital Sandbox

2.1.4 Requirements

This section details the specific requirements for the components and their connectivity. For geo-redundant systems, see “Configuring geo-redundancy” in the *Fabric Services System User Guide*.

2.1.4.1 Fabric Services System deployer requirements

The Fabric Services System deployer is used to install Kubernetes and the Fabric Services System application. It is delivered as one of the following images:

- QCOW2: the image that provides the deployer VM and which includes the operating system and the deployer software for the KVM hypervisor
- OVA: the VM image for the VMware vSphere 7.0 or 8.0 hypervisor

The minimal requirements for the Fabric Services System deployer virtual machines are:

- CPU: 1 virtual CPU (vCPU) on a VMware vSphere or KVM hypervisor with a modern x86-64 CPU that supports virtualization
  Recommended: 2 vCPU
- memory: 8 GB RAM
- storage: 50 GB of storage (used by the deployer QCOW2 image)
- networking:
  – 1 GbE network interface card (NIC)
  – the configured DNS servers must be reachable, functional, and able to resolve the hostnames used for the Fabric Services System nodes

2.1.4.2 Fabric Services System node requirements

The Fabric Services System nodes are deployed as virtual machine servers. The following are the requirements of the Fabric Services System nodes in KVM and VMware hypervisor:

- CPU: 32 vCPUs on a modern x86-64 CPU
- memory: 64 GB
- storage: at least 500 GB of available SSD-based storage
- networking:
  – at least one 10 Gbps NIC
  – the configured DNS servers must be reachable, functional, and able to resolve the hostnames used for the Fabric Services System nodes

You can run the Fabric Services System nodes as virtual machines using the following virtualization platforms:

- operating system: VMware vSphere 7.0 or 8.0 or RHEL/Rocky 8.9
- hypervisor: ESXi 7.0 or 8.0 or KVM

- resource reservation for CPU, memory, and disks must be set to 100% for the Fabric Services System node virtual machines

Related topics

Supported deployment models

2.1.4.3 Geo-redundancy

To bring up geo-redundant setup, ensure that the setup is similar in the active and backup sites and that network connectivity is stable in both sites. For additional requirements, see “Geo-redundancy configuration” in the *Fabric Services System User Guide*.

2.1.5 Installation process overview

The installation consists of the following high-level tasks:

1. Downloading the Fabric Services System deployer image
   This task describes how to access the Fabric Services System deployer image so it can be used in the next task.
2. Fabric Services System deployer VM creation
   This task describes how to create the VM in a VMware vSphere or KVM environment and how to configure the Fabric Services System deployer VM.
3. Preparing the Fabric Services System virtual machine nodes
   This task describes how to prepare and configure the nodes for deployment.
4. Editing the installation configuration file
   This task describes how to modify the variables and enter the expected data in the JSON-based configuration file, which is used to specify the configuration and deployment of the Fabric Services System environment.
5. Installing the Fabric Services System environment
   This task describes the deployment of the Fabric Services System environment using the `sample-input.json` file.

3 The Fabric Services System deployer VM

The procedures in this section describe how to deploy and configure the Fabric Services System deployer VM.

3.1 Downloading the Fabric Services System deployer image

Contact Nokia support for the location of the Fabric Services System deployer QCOW2 or OVA image.

Table 2: Deployer VM images

Deployment	Where to download the image
VMware vSphere	Download the OVA image to a host that can reach the VMware vCenter or ESXi host on which it will be deployed.
KVM	Download the QCOW2 image to the deployer host.

3.2 Prepare the Fabric Services System deployer hypervisor

Before you install the deployer VM, you must prepare the node on which you are installing the deployer VM. Virtualization must be enabled on the node and can be enabled in the BIOS or EFI. You must also install a VMware vSphere or KVM environment on the node before installing the deployer VM.

3.3 Fabric Services System deployer VM creation

After you have downloaded the OVA or QCOW2 image and prepared the deployer node, follow the installation steps to create the deployer VM.

The Fabric Services System nodes contained in the cluster (worker nodes) and the node hosting the deployer VM must communicate with each other. Both the worker nodes and the deployer VM must be able to initiate connections.

Use one of the following procedures to configure networking for the deployer VM using a bridged network:

- Creating the VM on a bridged network on KVM
- Creating the VM on VMware vSphere

3.3.1 Creating the VM on a bridged network on KVM

About this task

This section provides an example script used to create a VM in a KVM-based hypervisor. You can use this script or you can use your own procedure as long as the resulting VM meets the requirements for the Fabric Services System VM.

Procedure

Step 1. Create an `fssvm_create.sh` file, then copy the following contents into the file:

```
create_fssvm() {
    BRIDGE="breth0:1"
    VM=fss-deployer
    VMDIR=/var/lib/libvirt/images/$VM
    FSSIMAGE=<path to fss-installer qcow2 image>
    sudo mkdir -vp $VMDIR
    sudo cp $FSSIMAGE $VMDIR/$VM.qcow2
    sudo virsh pool-create-as --name $VM --type dir --target $VMDIR
    sudo virt-install --import --name $VM \
    --memory 8096 --vcpus 1 --cpu host \
    --disk $VMDIR/$VM.qcow2,format=qcow2,bus=virtio \
    --network bridge=$BRIDGE,model=virtio \
    --os-variant=centos7.0 \
    --noautoconsole --debug
}
VMDIR=.
create_fssvm
```

Step 2. In the script, modify the FSSIMAGE=<path to fss-installer qcow2 image> field to show the actual path to the Fabric Services System image on your system.

```
FSSIMAGE=./fss-deployer-x.y.qcow2
```

Step 3. Modify the permissions of the shell script file.

```
chmod 755 fssvm_create.sh
```

Step 4. Execute the shell script.

```
./fssvm_create.sh
```

3.3.2 Creating the VM on VMware vSphere

About this task

You can use one of the following methods to deploy the VM on VMware vSphere:

- the VMware vSphere vCenter or ESXi UI
  For instructions, see “Deploy an OVF or OVA Template” in the VMware vSphere documentation.
- the VMware Open Virtualization Format Tool CLI
  The following section provides an example of how to use the VMware OVF Tool CLI.

Procedure

Step 1. Download and install the latest version of the VMware OVF Tool from the VMware Developer website.

Step 2. Display details about the OVA image.

Example

Execute the **ovftool** command with just the OVA image name as the argument.

```
$ ovftool fss-deployer-25.8.1-414.ova
OVF version: 1.0
VirtualApp: false
Name: fss-deployer


Download Size: 17.40 GB


Deployment Sizes:
Flat disks: 40.00 GB
Sparse disks: 21.38 GB


Networks:
Name: OAM
Description: The Fabric Services System OAM (UI and API) network


Name: FABRIC
Description: The Fabric Services System Fabric Management network


Virtual Machines:
Name: fss-deployer
Operating System: centos7_64guest
Virtual Hardware:
Families: vmx-14
Number of CPUs: 2
Cores per socket: 1
Memory: 7.91 GB


Disks:
Index: 0
Instance ID: 4
Capacity: 40.00 GB
Disk Types: SCSI-lsilogic


NICs:
Adapter Type: VmxNet3
Connection: OAM


Adapter Type: VmxNet3
Connection: FABRIC


References:
File: fss-deployer-disk1.vmdk
```

Step 3. Deploy the OVA image using the OVF Tool.

For details about command line arguments, see the OVF Tool documentation from the VMware website.

Note: Ensure that you use thick provisioning for the disk and to connect all the interfaces to a network. The secondary interface can be disconnected and disabled after the deployment and before you power on.

Example

```
$ ovftool --acceptAllEulas -dm=thick -ds=VSAN -n=fss-deployer --net:"OAM=OAM-
network" --net:"FABRIC=Fabric-network" fss-deployer_24.5.1-414.ova vi://administrator
%40vsphere.local@vcenter.domain.tld/My-Datacenter/host/My-Cluster/Resources/My-
Resource-Group

Opening OVA source: fss-deployer_25.8.1-414.ova
The manifest validates
Enter login information for target vi://vcenter.domain.tld/
Username: administrator%40vsphere.local
Password: ***********
Opening VI target: vi://administrator%40vsphere.local@vcenter.domain.tld/My-
Datacenter/host/My-Cluster/Resources/My-Resource-Group
Deploying to VI: vi://administrator%40vsphere.local@vcenter.domain.tld/My-Datacenter/
host/My-Cluster/Resources/My-Resource-Group
Transfer Completed
```

3.4 Configuring the Fabric Services System deployer VM

Procedure

Step 1. From the VMware vSphere console or the KVM console, log in to the deployer VM.

Use the following credentials:

Username: `root`

Password: `N0ki@FSSb4se!`

Note: After the initial login, Nokia recommends that you change this default password to a stronger password to enhance the security of the deployer and the Fabric Services System environment.

Step 2. If your environment does not support or use cloud-init services, disable and stop these services.

```
# systemctl stop cloud-init cloud-init-local cloud-config cloud-final
```

```
# systemctl disable cloud-init cloud-init-local cloud-config cloud-final
```

Step 3. Enable SSH.

The base image is a hardened image, so SSH is disabled by default for the root user. To enable SSH, update the `/etc/ssh/sshd_config` file and change the following lines:

```
PasswordAuthentication no
PermitRootLogin no
```

to:

```
PasswordAuthentication yes
PermitRootLogin yes
```

Note: You can keep password authentication disabled to provide extra security. In this case, only key-based authentication works, and you must configure the appropriate public SSH keys for the root user to log in over SSH. In any case, this configuration is needed for the deployer VM to reach the nodes.

Step 4. Restart SSH.

```
# systemctl restart sshd
```

Step 5. Edit the `/etc/sysconfig/network-scripts/ifcfg-eth0` file to configure the correct static IP address, DNS servers, and gateway.

Example

The final content should look similar to the following, except with the IP address, DNS, and domain details specific to the target environment:

```
BOOTPROTO=static
DEVICE=eth0
ONBOOT=yes
TYPE=Ethernet
USERCTL=no
IPADDR=192.0.2.10
PREFIX=24
GATEWAY=192.0.2.1
DNS1=192.0.2.5
DNS2=192.0.2.6
DOMAIN=fss.nokia.local
MTU=9000
```

Step 6. Restart the network to apply the new configuration.

Execute the following command:

```
# systemctl restart NetworkManager.service
```

Step 7. Configure the appropriate NTP servers.

Edit the `/etc/chrony.conf` configuration file and replace all lines beginning with `server` with the correct server lines for the environment.

Step 8. Restart the chronyd service.

```
# systemctl restart chronyd
```

Step 9. Verify that time synchronization is functioning properly.

```
# chronyc tracking
```

Example

```
Reference ID    : 87E30FFE (192.0.2.5)
Stratum         : 4
```

```
Ref time (UTC)  : Wed Feb 16 01:20:36 2022
System time     : 0.000014215 seconds slow of NTP time
Last offset     : -0.000001614 seconds
RMS offset      : 0.000106133 seconds
Frequency       : 11.863 ppm slow
Residual freq   : -0.071 ppm
Skew            : 0.187 ppm
Root delay      : 0.063009784 seconds
Root dispersion : 0.018440660 seconds
Update interval : 64.5 seconds
Leap status     : Normal
```

Expected outcome

If the **Reference ID** field does not show any of the configured servers, but instead refers to something like 127.127.1.1, time synchronization is not functioning properly.

Step 10. Synchronize the RTC clock and the system clock.

Example

Ensure that the RTC and the system clock are synchronized after every reboot.

```
# hwclock --systohc
```

Then, verify that local time and the RTC time are synchronized.

```
# timedatectl
```

Step 11. Optional: Change the hostname.

```
# hostnamectl set-hostname new-hostname.domain.tld
```

Step 12. Reboot the Fabric Services System deployer VM to ensure that all services come up with the correct network configuration.

```
# reboot
```

4 Preparing the Fabric Services System virtual machine nodes

The procedures in this section describe how to create and configure Fabric Services System nodes in deployments that use virtual machine servers.

You must use the Fabric Services System base OS image. This image is specially designed for use with the Fabric Services System deployment and comes with the necessary software and components, pre-installed in a minimally-hardened Rocky 9.6 operating system.

Note: The base image OS upgrade to Rocky 9.6 introduces significant changes. The scope of the changes are beyond the scope of this document, you should familiarize yourself with manipulating any settings after creating the vm. The main differences could be in the configuration you will be changing and the tools available to make those configuration changes.

Complete the procedure for each individual Fabric Services System node, ensuring that each node is running on a separate hypervisor to minimize the risk of any impact if a hypervisor fails.

4.1 Downloading the Fabric Services System base OS image

Contact Nokia support for the location of the Fabric Services System base OS image. Download the OVA or QCOW2 image.

4.2 Networking considerations

Nokia recommends that you use two different networks for the Fabric Services System nodes.

Within the hypervisor, both networks should be available as bridged networks. Both these networks require support for jumbo frames (MTU set to 9000).

Ensure that the MTU is set to 9000 on all the interfaces on the hypervisor, Fabric Service System VM nodes, deployer, and the interconnecting devices.

Related topics

Networking for the Fabric Services System nodes

4.3 Create the Fabric Services System virtual machine

To deploy a Fabric Services System node as a virtual machine, complete the appropriate procedure for your deployment scenario:

- Creating the VM on bridged networks on KVM
- Creating the VM on bridged networks on VMware vSphere

4.3.1 Creating the VM on bridged networks on KVM

About this task

Complete the following steps to deploy a Fabric Services System node as a virtual machine on KVM. The OAM network is referred to as br0 and the fabric management network is referred to as br1.

Procedure

Step 1. Ensure that the **virt-install** tool is installed on the KVM hypervisor.

If you need to install the tool, use the following command:

```
# yum install virt-install
```

Note: With Rocky 9.6, the **--graphics** flag is no longer supported with the **virt-install** tool.

Step 2. Copy the base OS image to the appropriate location on the hypervisor where the virtual disks should be stored.

Example

```
# cp fss-baseos-rocky-0.0.0-alpha.1+20240430.191213-f50a660.qcow2 /path/to/fss-
compute-01.qcow2
```

Step 3. Resize the base OS image.

By default, the Fabric Services System base OS image comes with a small partition to reduce the download size of the image. To assign the appropriate size to the image, execute the following commands:

```
# qemu-img create -f qcow2 -o preallocation=metadata /path/to/fss-compute-
01.qcow2.temp 200G
# virt-resize --quiet --expand /dev/sda5 /path/to/fss-compute-01.qcow2 /path/to/fss-
compute-01.qcow2.temp
# mv /path/to/fss-compute-01.qcow2.temp /path/to/fss-compute-01.qcow2
```

Expected outcome

These commands create a temporary qcow2 file of 200GB and then resizes the root partition of fss-compute-01.qcow2 and writes it to the temporary file. Then, the `.temp` file is renamed to the actual qcow2 file used to create the compute VM.

Step 4. Optional: If the node is also going to be used as a storage node, create the necessary extra disk for the storage cluster to be formed.

Example

Create the virtual disk of 300GB using the following command:

```
# qemu-img create -f qcow2 /path/to/fss-node01-storage.qcow2 300G
```

Step 5. Create the virtual machine.

Example

The following command creates a node that also serves as a storage node. If a storage node is not needed, omit the second line that starts with `--disk`.

```
# virt-install --import --name fss-node01 \
 --memory 65536 --vcpus 32 --cpu host \
 --disk /path/to/fss-node01.qcow2,format=qcow2,bus=virtio \
 --disk /path/to/fss-node01-storage.qcow2,format=qcow2,bus=virtio \
 --network bridge=br0,model=virtio \
 --network bridge=br1,model=virtio \
 --os-variant=centos7.0 \
 --noautoconsole
```

Step 6. After the VM boots, use the **lsblk** command to verify the partition sizes.

Note:
Log in using the following credentials:

user: root

password: N0ki@FSSb4se!

Example

```
# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vda 252:0 0 200G 0 disk
├─vda1 252:1 0 99M 0 part /boot/efi
├─vda2 252:2 0 1000M 0 part /boot
├─vda3 252:3 0 4M 0 part
├─vda4 252:4 0 1M 0 part
└─vda5 252:5 0 198.9G 0 part /
vdb 252:16 0 300G 0 disk
```

Expected outcome

In the output, verify that the vda5 partition is resized to 198GB and the vdb partition is 300GB. You can use the vdb partition for the storage cluster.

What to do next

Configuring the Fabric Services System virtual machine

4.3.2 Creating the VM on bridged networks on VMware vSphere

About this task

You can use one of the following methods to deploy the VM on VMware vSphere:

- the VMware vSphere vCenter or ESXi UI
  For instructions, see *Deploy an OVF or OVA Template* in the VMware vSphere documentation.
- the VMware Open Virtualization Format Tool CLI
  The following section provides an example of how to use the VMware OVF Tool CLI.

Note: The VMware OVA for the base image comes with a 128GB disk during deployment. This disk is sufficient as a base disk for the OS partition. During the initial boot of the VM, cloud-init grows the root partition to fill the free space on that 128GB disk.

Procedure

Step 1. Download and install the latest version of the VMware OVF Tool from the VMware Developer website.

Step 2. Display details about the OVA image.

Example

Execute the **ovftool** command with just the OVA image name as the argument.

```
$ ovftool fss-baseos-25.8.1-414.ova
OVF version: 1.0
VirtualApp: false
Name: fss-baseos


Download Size: 2.32 GB


Deployment Sizes:
Flat disks: 128.00 GB
Sparse disks: 4.08 GB


Networks:
Name: OAM
Description: The Fabric Services System OAM (UI and API) network


Name: FABRIC
Description: The Fabric Services System Fabric Management network


Virtual Machines:
Name: fss-baseos
Operating System: centos7_64guest
Virtual Hardware:
Families: vmx-14
Number of CPUs: 16
Cores per socket: 1
Memory: 64.00 GB


Disks:
Index: 0
Instance ID: 4
Capacity: 128.00 GB
Disk Types: SCSI-lsilogic


NICs:
Adapter Type: VmxNet3
Connection: OAM


Adapter Type: VmxNet3
Connection: FABRIC
```

```
References:
File: fss-baseos-disk1.vmdk
```

Step 3. Deploy the OVA image using the OVF Tool.

For details about command line arguments, see the OVF Tool documentation from the VMware website.

Note: Ensure that you use thick provisioning for the disk and to connect all the interfaces to a network. The secondary interface can be disconnected and disabled after the deployment and before you power on.

Example

```
$ ovftool --acceptAllEulas -dm=thick -ds=VSAN -n=fss-node01 --net:"OAM=OAM-network"
 --net:"FABRIC=Fabric-network" fss-baseos_24.12.1-414.ova vi://administrator
%40vsphere.local@vcenter.domain.tld/My-Datacenter/host/My-Cluster/Resources/My-
Resource-Group

Opening OVA source: fss-base_25.8.1-414.ova
The manifest validates
Enter login information for target vi://vcenter.domain.tld/
Username: administrator%40vsphere.local
Password: ***********
Opening VI target: vi://administrator%40vsphere.local@vcenter.domain.tld/My-
Datacenter/host/My-Cluster/Resources/My-Resource-Group
Deploying to VI: vi://administrator%40vsphere.local@vcenter.domain.tld/My-Datacenter/
host/My-Cluster/Resources/My-Resource-Group
Transfer Completed
Completed successfully
```

Step 4. Optional: If you are using this node as a storage node, create the necessary extra disk for the storage cluster to be formed.

To create the extra disk, edit the VM in the VMware vCenter and add a new 300 GB disk.

Step 5. Enable 100% resource reservation for both CPU and memory for the VM.

You can configure the resource reservation for CPU and memory by editing the VM in vCenter.

What to do next

Configuring the Fabric Services System virtual machine

4.4 Configuring the Fabric Services System virtual machine

About this task

Complete the following steps to configure a Fabric Services System node.

Procedure

Step 1. From the VMware vSphere or KVM console, log in to the node VM.

Use the following credentials:

Username: `root`

Password: `N0ki@FSSb4se!`

Step 2. If your environment does not support or use the cloud-init services, disable and stop these services.

Example

```
# systemctl stop cloud-init cloud-init-local cloud-config cloud-final
# systemctl disable cloud-init cloud-init-local cloud-config cloud-final
```

Step 3. Edit the `/etc/sysconfig/network-scripts/ifcfg-eth0` file to configure the correct static IP address, DNS servers, and gateway for the OAM network.

Note: If you are deploying a dual-stack system, provide the IPv6 details in the `ifcfg-eth0` file. Additionally, ensure that the default gateway is configured for both IPv4 and IPv6 and that both gateways are functional before installing the Fabric Services System.

Example

The final content should look similar to the following example, except with the IP address, DNS, and domain details specific to the target environment:

```
BOOTPROTO=static
DEVICE=eth0
ONBOOT=yes
TYPE=Ethernet
USERCTL=no
IPADDR=192.0.2.10
PREFIX=24
GATEWAY=192.0.2.1
DNS1=192.0.2.5
DNS2=192.0.2.6
DOMAIN=fss.nokia.local
MTU=9000
```

Step 4. Edit the `/etc/sysconfig/network-scripts/ifcfg-eth1` file to configure the correct static IP address for the fabric management network.

Ensure that the MTU parameter is set to 9000 for all the interfaces.

Example

The final content should look similar to the following, except with the IP address, DNS, and domain details specific to the target environment:

```
BOOTPROTO=static
DEVICE=eth0
ONBOOT=yes
TYPE=Ethernet
USERCTL=no
IPADDR=192.0.2.10
PREFIX=24
MTU=9000
```

Step 5. Restart the network to apply the new configuration.

Example

```
# systemctl restart NetworkManager.service
```

Step 6. Configure the appropriate NTP servers.

Edit the `/etc/chrony.conf` configuration file and replace all lines that begin with `server` with the correct server lines for the environment.

Step 7. Restart the chronyd service.

```
# systemctl restart chronyd
```

Step 8. Verify that time synchronization is functioning properly.

```
# chronyc tracking
```

Expected outcome

If the **Reference ID** field is not set to any of the configured servers, but instead refers to something like 127.127.1.1, time synchronization is not functioning properly.

```
Reference ID    : 87E30FFE (192.0.2.5)
Stratum         : 4
Ref time (UTC)  : Wed Feb 16 01:20:36 2022
System time     : 0.000014215 seconds slow of NTP time
Last offset     : -0.000001614 seconds
RMS offset      : 0.000106133 seconds
Frequency       : 11.863 ppm slow
Residual freq   : -0.071 ppm
Skew            : 0.187 ppm
Root delay      : 0.063009784 seconds
Root dispersion : 0.018440660 seconds
Update interval : 64.5 seconds
Leap status     : Normal
```

Step 9. Synchronize the RTC clock and the system clock.

Ensure that the RTC and the system clock are synchronized after every reboot.

Example

```
# hwclock --systohc
```

Then, verify that local time and the RTC time are synchronized.

```
# timedatectl
```

Step 10. Change the hostname.

Example

```
# hostnamectl set-hostname fss-node01.domain.tld
```

Step 11. Set up key-based authentication from the deployer VM to Fabric Services System compute VMs.

Example

If password authentication has been enabled on the node for SSH, enter the following command from the deployer VM.

```
# ssh-copy-id root@<node IP/FQDN>
```

5 Fabric Services System installation

After the Fabric Services System deployer VM and the Fabric Services System nodes have been installed and configured, the environment is ready to be installed with the Fabric Services System application.

Using HTTPS for the UI and API

By default, HTTPS is enabled for the UI and API to enforce the use of TLS encryption (v1.2 or v1.3) for all communication to the Fabric Services System management interfaces. Enabling HTTPS guarantees that all information is secured against snooping or changes during transit.

During the installation, a set of default certificates are generated by the installer for use by the system in different functions. After the installation, a tool and process is available to replace these auto generated certificates with customer specific certificates. More information, see “Certificate Management” in the *Fabric Services System User Guide*.

Dual-stack networks

The system supports a dual-stack network, where each VM has an IPv4 and an IPv6 address; the system does not support a pure IPv6 network.

Note:

- The network must be ready for IPv6 and IPv4 IP addresses.
- Each node must be configured with an IPv4 and IPv6 default gateway and the gateways must be functional.
- The pods running in the VMs need to connect to SR Linux, which is in a different network.
- After creating the VMs, Nokia recommends that you verify the required connectivity over IPv4 and IPv6.

To enable support for a dual-stack network, set the **enable_dual_stack_networks** parameter to `True` in the `sample-input.json` file. You must also set an IP v6 address in the **ip6** parameter for each worker node and storage node.

Geo-redundant installations

For a geo-redundant system, you need to deploy two Fabric Services System clusters, one for the intended active system and the other the standby, using different `sample-input.json` for the active and standby sites. The relevant geo-redundancy settings for the deployer on the active and standby sites are shown in Geo-redundant deployer configuration example. For related information, see Preparing the Fabric Services System clusters for geo-redundant configuration.

Load balancing using a VIP

The system supports cluster-based load balancing using a virtual IP (VIP) address for the OAM (UI, API, and other northbound services) and one for the node networks (for SR Linux ZTP and DHCP relay). If a Fabric Services System compute node that services an OAM or node network service IP becomes unavailable, the system can continue to service these networks through the VIP, as it automatically assigns the virtual IP to another compute node.

You can configure an IPv4 and IPv6 VIP for the OAM and node networks.

When cluster load balancing is configured, the MetalLB load balancer is installed on the K8s cluster. MetalLB provides load balancing service in L2 mode. MetalLB uses the configured IP address and assigns this IP address to each node in the cluster. MetalLB sets only one node to respond to ARP requests. If this active node fails, another node becomes active. Typically, the IP pool is within the CIDR of one of the interfaces of the nodes, so no extra routing configuration is needed in the network.

In the `sample-input.json` file, the parameters for configuring load-balancing are in **k8s** section, under **lbconfig**. First, set the **lbtype** parameter to `native`. The **oam** section is for configuring the VIP and interface on which to advertise the VIP for the North-bound interfaces (including the UI and REST API); the **node** section is for configuring the VIP and interface on which to advertise the VIP the node network (for SR Linux ZTP and DHCP relay). The VIP for the node network should be the same as the setting for **ztpaddress** and **ztpv6address** parameters in the `sample-input.json`.

Related topics

Editing the installation configuration file

5.1 Editing the installation configuration file

About this task

As part of the deployment, you must provide specific details about the configurable portions of the installation using a configuration file. The details you provide instruct the deployer how to proceed when setting up the Kubernetes deployment, the Fabric Services System software, and the Digital Sandbox.

Note: You can rename the input configuration file to a name specific to your deployment. In the examples that follow, the input configuration file is named `sample-input.json`.

Note:
Create a different `sample-input.json` for the active and standby sites. The relevant geo-redundancy settings for the deployer on the active and standby sites are shown in Geo-redundant deployer configuration example.

Procedure

Step 1. From the deployer, access the input configuration file.

```
[root@fss-deployer ~] vi sample-input.json
```

Step 2. Edit the configuration file.

Update the file with the following settings:

- IP addresses of the nodes to be used in your Fabric Services System deployment
- deployer node settings
- worker node settings
- storage node settings

Note: When you set the **devices** parameter, specify only the partition name (`"sdb" or "vdb"` in the example below). You do not need to specify the path.

- time synchronization
- replica count
- optional: Digital Sandbox installation characteristics
- optional: remote syslog settings
- optional: load balancer configuration

The deployer creates three Kubernetes master nodes. To specify which worker nodes are Kubernetes master nodes, tag the nodes with the `master` role in the configuration file.

The table below describes the fields in the `sample-input.json` file. Examples of `sample-input.json` files for IPv4 and dual-stack deployments follow.

Table 3: Field definitions

Heading	Configurable values
deployernode Specifies the IP address, gateway, and netmask configured on the network interface of the deployer VM. The deployer VM must be reachable by all of the Fabric Services System nodes, and the Fabric Services System nodes must be reachable by the deployer VM.	• **ipaddr**: the IP address of the deployer VM. • **gateway**: the gateway address of the interface on the deployer node. • **netmask**: the netmask of the interface on the deployer node. • **sitename**: specifies the name of the site in a geo-redunant configuration. If this field is not set, the deployer name is used by default. • **role**: specifies whether the role of the site is active, standby, or standalone in a geo-redunant configuration. If this field is not set, it is set to standalone by default. • **accessip**: specifies the IP address to use to access the deployer in a geo-redunant configuration. If this field is not set, the IP address configured for the **ipaddr** field is used.
rsyslog Specifies the remote syslog server settings.	• **host**: the IP address or FQDN of the remote syslog server. • **port**: the port that the rsyslog utility uses for network connectivity. • **proto**: the protocol used for syslog traffic, either TCP or UDP. **Note:** The system currently supports one remote syslog server.
digitalsandbox Specifies Digital Sandbox parameters.	**enabled**: when this flag is set to `true`, the Digital Sandbox component is installed. Ensure that at least one worker node is tagged with the `digitalsandbox` role. When set to `false`, the Digital Sandbox component is not installed.
fss Specifies Fabric Services System deployment options.	• **ztpaddress**: specifies an address associated with the node running Traefik. The node can be any of the Fabric Services System cluster nodes. The SR Linux nodes connect to this IP

Heading	Configurable values
	address during the boot process to get the software image and the configuration. This IP address must be reachable from the SR Linux management network. • **ztpv6address**: specifies the IPv6 address associated with the node for SR Linux to connect using IPv6. • **dhcpnode**: specifies a node on which the Fabric Services System DHCP pod is scheduled. • **dhcpinterface**: specifies the address that the DHCP server listens to for any DHCP requests coming from the DHCP relay agent. Optionally, you can connect SR Linux nodes via the relay agent to reach the Fabric Services System if they are not on the management network. If the network is not configured with a DHCP relay agent, do not set this parameter; remove it if it present in the input configuration file. • **dhcpv6interface**: specifies the IPv6 address of the DHCPv6 relay agent. If the network is not configured with a DHCP relay agent, do not set this parameter; remove it if it present in the input configuration file. • **truststoreFilename**: specifies the location of the truststore filename with the absolute path information. The JKS file must be generated to access the LDAP server from the Fabric Services System instance. The alternate names in the certificate should match the name and IP address configured for the federation provider (using the Fabric Services System UI or REST API). • **truststorePassword**: specifies the password used to access the truststore. • **kafkaconfig**: Configures the parameters that enable third-party tools to access Fabric Services System alarms. – **port**: the port number used by the client to connect to the Kafka service; specify a value between 30000 and 32767. – **groupprefix**: the user group prefix for the client to use to connect to Kafka service. – **user** and **password**: the credentials to use to authenticate. – **maxConnections**: the maximum number of clients that can connect to the Kafka service. The maximum allowed value is 10.
K8s Specifies whether the system supports dual-stack networks and configures cluster-based load balancing	• **enable_dual_stack_networks**: specifies whether dual-stack network is supported. Set to `True` to enable support for IPv6 networks.

Heading	Configurable values
	Note: The system supports only a dual-stack network where each VM has an IPv4 and IPv6 address; the system does not support a pure IPv6 network. • **lbconfig** section: configuration for the OAM and node load balancing. **lbtype**: set to `native` to configure lode-balancing. If set, either the **oam** or **node** sections, or both sections must be provided. – **oam** section: configures load-balancing for the north-bound interface. **interface**: specifies the interface on which the load balancer address is advertised. **ipv4**: specifies the IPv4 address for the load balancer to use. This value is mandatory. **ipv6**: specifies the IPv6 address for load balancer to use. – **node** section: configures load-balancing for SR Linux ZTP and DHCP relay. **interface**: specifies the interface on which the load balancer address is advertised. **ipv4**: specifies the IPv4 address for the load balancer to use. **ipv6**: specifies the IP address for the load balancer to use. This value must be the same as the value for the **ztpv6address** parameter.
workernodes Specifies the list of nodes intended to be part of the deployment, except for the deployer host. Worker nodes include storage nodes and Digital Sandbox nodes.	• **hostip**: the IP address of the specific worker node. • **ip6**: the IPv6 address of the worker node; required if the **enable_dual_stack_networks** parameter is set to `True`. • **hostname**: the hostname of the worker node. • **role**: the specified role of the worker node. For Digital Sandbox nodes, specify this value as `digitalsandbox`. For Kubernetes master nodes, specify this value as `master`.
replicacount Specifies the replica count for Gluster volumes, including the active volume.	The default value is `1`, indicating no replica (active volume only). A replica count higher than `1` creates the respective number of replica storage volumes. The value cannot be greater than the number of storage nodes.
storagenodes Specifies the list of nodes used to create a storage pool. The number of	• **hostip**: the IP address of the specific storage node. • **ip6**: the IPv6 address of the specific storage node; required if the **enable_dual_stack_networks** parameter is set to `True`.

Heading	Configurable values
storage nodes must match the value of `replicacount`, if configured. Nokia recommends that you configure a minimum of three storage nodes.	• **hostname**: the hostname of the storage node. • **devices**: separate block devices must be configured. Configure a raw partition as `xxx`. If an existing file system is present on the device, the setup cannot proceed.
singlenode Specifies whether the deployment consists of only a single node for extra small deployments.	The default value is `false`, indicating that the deployment is a standard three- or six-node deployment. If set to `true`, the deployment is set up on a single node and has no redundancy built in.

For examples, see the following `sample-input.json` files:

- IPv4 example
- Dual-stack example
- Geo-redundant deployer configuration example

What to do next

After you finish editing the input configuration file, you can install the Fabric Services System environment as described in Installing the Fabric Services System environment.

5.1.1 IPv4 example

Example

The following is an example of a `sample-input.json` configuration file for an IPv4 network.

```
{
  "deployernode": {
    "ipaddr": "192.0.2.200",
    "gateway": "192.0.2.1",
    "netmask": "255.255.254.0"
    "sitename": "SiteA",
        "role": "active",
        "accessip": "10.254.107.74"
  },
  "digitalsandbox": {
    "enabled": true,
    "volumenode": "fss-node04"
  },
  "fss": {
     "dhcpnode": "fss-node01",
     "dhcpinterface": "128.66.0.201/24",
     "ztpaddress": "192.168.2.3",
     "kafkaconfig": {
         "port": "31000",
         "groupprefix": "fsskafka",
         "user": "fssalarms",
         "password": "fssalarms",
         "maxConnections": 2
     }
  },
  "rsyslog": {
        "host": "192.0.2.161",
```

```
      "port": 51400,
      "proto": "udp"
   },
"k8s": {
    "enable_dual_stack_networks": false
    "lbconfig": {
      "lbtype": "native",
       "oam": {
         "interface": "eth0"
         "ipv4": ["192.168.5.3"]
       }
        "node": {
        "interface": "eth1"
        "ipv4": ["192.168.2.3"]
       }
     }
 },
"replicacount": 2,
"workernodes": [
  {
    "hostip": "192.0.2.201",
    "hostname": "fss-node01",
    "role": "master"
  },
  {
    "hostip": "192.0.2.202",
    "hostname": "fss-node02",
    "role": "master"
  },
  {
    "hostip": "192.0.2.203",
    "hostname": "fss-node03",
    "role": "master"
  },
  {
    "hostip": "192.0.2.204",
    "hostname": "fss-node04",
    "role": "digitalsandbox"
  },
  {
    "hostip": "192.0.2.205",
    "hostname": "fss-node05",
    "role": "digitalsandbox"
  },
  {
    "hostip": "192.0.2.206",
   "hostname": "fss-node06",
    "role": "digitalsandbox"
  }
],
"storagenodes": [
  {
    "hostip": "192.0.2.204",
    "hostname": "fss-node04",
    "devices": [
      "sdb1"
    ]
  },
  {
    "hostip": "192.0.2.205",
    "hostname": "fss-node05",
    "devices": [
      "sdb1"
    ]
```

```
    },
    {
      "hostip": "192.0.2.206",
      "hostname": "fss-node06",
      "devices": [
        "sdb1"
      ]
    }
  ]
}
```

5.1.2 Dual-stack example

Example

The following `sample-input.json` configuration file is an example for a dual stack, IPv4 and IPv6 networks.

```
{
  "deployernode": {
    "ipaddr": "192.0.2.200",
    "gateway": "192.0.2.1",
    "netmask": "255.255.254.0"
  },
  "digitalsandbox": {
    "enabled": true,
    "volumenode": "fss-node04"
  },
  "fss": {
     "dhcpnode": "fss-node01",
     "dhcpinterface": "128.66.0.201/24",
     "dhcpv6interface": "2001:db8:f128:0::201/64",
     "ztpaddress": "128.66.0.201",
     "ztpv6address": "2001:db8:f128:0::201",
     "truststoreFilename": "/root/fss.truststore.jks",
     "truststorePassword": "fss123"
     "kafkaconfig": {
         "port": "31000",
         "groupprefix": "fsskafka",
         "user": "fssalarms",
         "password": "fssalarms",
         "maxConnections": 2
  },
  "k8s": {
      "enable_dual_stack_networks": true
      "lbconfig": {
        "lbtype": "native",
        "oam": {
          "interface": "eth0",
          "ipv4": ["192.0.2.100"],
          "ipv6": ["2001:db8:f685:0::100"]
        }
         "node": {
         "interface": "eth1",
         "ipv4": ["128.66.0.201"],
         "ipv6": ["2001:db8:f128:0::201"]
        }
      }
  },
  "replicacount": 2,
  "rsyslog": {
```

```
      "host": "192.0.2.161",
      "port": 514,
      "proto": "udp"
  },
  "workernodes": [
    {
      "hostip": "192.0.2.201",
      "ip6": "2001:db8:f685:0::201",
      "hostname": "fss-node01",
      "role": "master"
    },
    {
      "hostip": "192.0.2.202",
      "ip6": "2001:db8:f685:0::202",
      "hostname": "fss-node02",
      "role": "master"
    },
    {
      "hostip": "192.0.2.203",
      "ip6": "2001:db8:f685:0::203",
      "hostname": "fss-node03",
      "role": "master"
    },
    {
      "hostip": "192.0.2.204",
      "ip6": "2001:db8:f685:0::204",
      "hostname": "fss-node04",
      "role": "digitalsandbox"
    },
    {
      "hostip": "192.0.2.205",
      "ip6": "2001:db8:f685:0::205",
      "hostname": "fss-node05",
      "role": "digitalsandbox"
    },
    {
      "hostip": "192.0.2.206",
      "ip6": "2001:db8:f685:0::206",
      "hostname": "fss-node06",
      "role": "digitalsandbox"
    }
  ],
  "storagenodes": [
    {
      "hostip": "192.0.2.204",
      "ip6": "2001:db8:f685:0::204",
      "hostname": "fss-node04",
      "devices": [
        "sdb1"
      ]
    },
    {
      "hostip": "192.0.2.205",
      "ip6": "2001:db8:f685:0::205",
      "hostname": "fss-node05",
      "devices": [
        "sdb1"
      ]
    },
    {
      "hostip": "192.0.2.206",
      "ip6": "2001:db8:f685:0::206",
      "hostname": "fss-node06",
     "devices": [
```

```
        "sdb1"
      ]
    }
  ]
}
```

5.1.3 Geo-redundant deployer configuration example

For a geo-redundant system, create two `sample-input.json` files: one for the active and one for the standby. The following examples show the deployer section for the active and standby sites.

Example

In the `sample-input.json` file for the intended active site "primary", the `deployernode` section should be similar to the following example:

```
"deployernode": {
 "ipaddr": "192.0.2.200",
 "gateway": "192.0.2.1",
 "netmask": "255.255.254.0"
 "sitename": "primary",
 "role": "active",
 "accessip": "192.0.2.200"
},
```

In the `sample-input.json` file for the intended standby site "secondary", the `deployernode` section should be similar to the following example:

```
"deployernode": {
 "ipaddr": "192.0.20.200",
 "gateway": "192.0.20.1",
 "netmask": "255.255.254.0"
 "sitename": "secondary",
 "role": "standby",
 "accessip": "192.0.20.200"
},
```

5.2 Installing the Fabric Services System environment

Procedure

Step 1. Initiate the setup.

Example

```
[root@fss-deployer ~]$ fss-install.sh configure sample-input.json
```

The CLI prompt indicates when the configuration is complete.

Step 2. Start the installation of Kubernetes, the Fabric Services System software, and the Digital Sandbox.

Example

```
[root@fss-deployer ~]$ fss-install.sh
```

The installation time varies depending on the capacity of your system.

Step 3. After the installation script is completed, verify the installation by logging in to the Fabric Services System user interface using one of the node IP addresses.

Log in using the following default username and password:

Username: `admin`

Password: `NokiaFss1!`

Note: After the initial login, Nokia recommends that you change this default admin password to a stronger password to secure the platform properly.

What to do next

For geo-redundant systems, continue with Preparing the Fabric Services System clusters for geo-redundant configuration.

5.3 Preparing the Fabric Services System clusters for geo-redundant configuration

About this task

As described in Editing the installation configuration file, you need to deploy two Fabric Services System clusters, one for the intended active system and the other the standby.

Procedure

Step 1. Ensure that the clusters in the systems meet the following requirements:

- Use the same deployer and the compute image versions on both clusters.
- The number of nodes on the active cluster should match the number of nodes on the standby cluster; the node configurations should also match.
- The active and standby clusters should be installed with the site-specific network configurations.
- Connectivity should be established between the active and standby sites.
- The active and standby sites should have the correct network connectivity to the SR Linux network.

Step 2. Configure geo-redundancy for the active and standby clusters.

For instructions, see “Geo-redundancy” in the *Fabric Services System User Guide*.

5.4 How to troubleshoot a failed installation

If the Fabric Services System installation fails for any reason, you can use a script that is bundled with the system to generate information about the installation status. For assistance with troubleshooting, contact your Nokia support team.

The technical support script is included with the Fabric Services System.

For more information about the script and how to run it, see "Capturing troubleshooting data" in the *Fabric Services System User Guide*.

6 Software re-installation on an existing Kubernetes cluster

The Fabric Services System deployer provides a fast method to reinstall the system software without reinstalling the Kubernetes deployment. Use this re-installation procedure only if the Kubernetes deployment is functional and the hardware topology has not changed after the initial installation.

Note: The re-installation of the Fabric Services System application can only be done on a Kubernetes cluster that was previously installed using the same Fabric Services System deployer.

When you execute this procedure, all Fabric Services System software and data is removed except for the Kubernetes cluster. This procedure installs the same version of the Fabric Services System software with the same images used in the previous installation.

Complete the following tasks:

1. Uninstalling Fabric Services System software only – removes all Fabric Services System software and intent data while the Kubernetes cluster remains running
2. Reinstalling Fabric Services System software only – reinstalls the Fabric Services System software on the existing Kubernetes cluster

6.1 Uninstalling Fabric Services System software only

About this task

This procedure removes all the Fabric Services System software and intent data in the cluster. Only the Kubernetes cluster remains running.

Procedure

To uninstall Fabric Services System software and the related data, log in to the deployer VM and run following command:

```
[root@fss-deployer ~]$ /root/bin/fss-app-uninstall.sh
```

What to do next

After the software has been uninstalled, you can reinstall the same version of Fabric Services System software without reinstalling the Kubernetes deployment.

Related topics

Reinstalling Fabric Services System software only

6.2 Reinstalling Fabric Services System software only

Prerequisites

This procedure reinstalls Fabric Services System software on the existing Kubernetes cluster. You must first uninstall the Fabric Services System software.

- To reinstall the Fabric Services System software, log in to the deployer VM and enter the following command:

```
[root@fss-deployer ~]$ /root/bin/fss-app-install.sh
```

Related topics

Uninstalling Fabric Services System software only

7 Uninstalling software

This section describes how to uninstall a Fabric Services System deployment. Topics include:

- Uninstalling a Fabric Services System deployment: uninstalls a previous Fabric Services System installation
- Deleting the deployer VM: deletes the deployer VM from the node on which it is hosted

Note: If you want to bring up a new Fabric Services System cluster, Nokia recommends that you delete and recreate the cluster.
To quickly uninstall the Fabric Services system cluster and the deployer VM, delete the VMs where the Fabric Services System cluster and deployer are running or use the uninstall scripts as described in the procedures that follow.

For geo-redundant systems, complete the procedures on both clusters.

7.1 Uninstalling a Fabric Services System deployment

About this task

After completing the initial installation, you can uninstall a Fabric Services System deployment.

Procedure

Step 1. Uninstall Fabric Services System services, Kubernetes, Digital Sandbox, and storage volumes.

Example

From the deployer VM, run the following command:

```
[root@fss-deployer ~]$ /root/bin/fss-uninstall.sh all
```

Step 2. Reboot the storage nodes.

As part of the Fabric Services System uninstall procedure, the file system cleanup requires the storage nodes to be rebooted so the partition or disk can be used for Fabric Services System installation or other purposes.

7.2 Deleting the deployer VM

Prerequisites

To completely remove a Fabric Services System deployment, after uninstalling the Fabric Services System software, Digital Sandbox, and Kubernetes components, you must delete the deployer VM from the node on which it is hosted.
To delete the deployer VM:

Procedure

Step 1. Log in to the deployer host.

Step 2. Create an `fssvm_delete.sh` file, then copy the following contents into the file:

```
#!/bin/bash

delete_fssvm() {
  D=/var/lib/libvirt/images
  VM=$1

  sudo virsh shutdown $VM
  sudo virsh undefine $VM
  sudo  virsh pool-destroy $VM
  sudo  rm -ri $D/$VM
}
```

Step 3. Modify the permissions of the shell script file.

```
chmod 755 fssvm_delete.sh
```

Step 4. Execute the shell script.

```
./fssvm_delete.sh <name of the VM>
```

Expected outcome

The deployer VM is deleted from the deployer node.

8 Software upgrade and rollback

The Fabric Services System supports software upgrades from one release to a newer one in a KVM or VMware vSphere environment running in a supported configuration (with one deployer VM and one, three, or six Fabric Services System virtual machine nodes).

The procedures in this section describe how to prepare for the software upgrade and execute the software upgrade. The rollback procedure describes how to roll back after an upgrade, if needed.

The supported upgrade path is from Release 24.12.*x* to Release 25.8.*y*.

Note: In the software upgrade and rollback procedures, Release 24.12.*x* is referred to as the *old* release and Release 25.1.*y* is the *new* release.

8.1 Preparing for software upgrade

About this task

Use this procedure to prepare for the software upgrade from the *old* release to the *new* release.

Procedure

Step 1. Display the system information and status.

Log in to the old deployer VM using SSH.

a. Verify that the Helm charts are running the current release.

Example

```
[root@fss-deployer ~]# export KUBECONFIG=/var/lib/fss/config.fss
[root@fss-deployer ~]# helm list
NAME NAMESPACE REVISION UPDATED                                  STATUS  CHART
                        APP VERSION
prod default  1       2024-08-20 06:08:37.917207827 +0000 UTC deployed fss-FSS_24_8_
B1-charts-v24.8.2-XX
```

b. Verify that the pods are all up and running.

Example

```
[root@fss-deployer ~]# kubectl get pods
NAME                                                READY   STATUS    RESTARTS      AGE
prod-cp-kafka-0                                     2/2     Running   2 (32m ago)   35m
prod-cp-kafka-1                                     2/2     Running   1 (32m ago)   33m
prod-cp-kafka-2                                     2/2     Running   0             32m
prod-cp-zookeeper-0                                 2/2     Running   0             35m
prod-cp-zookeeper-1                                 2/2     Running   0             32m
prod-cp-zookeeper-2                                 2/2     Running   0             32m
prod-ds-apiserver-75ffbd8dd8-bbgt2                  1/1     Running   0             35m
prod-ds-cli-67bc6c89c7-dfdbn                        1/1     Running   0             35m
prod-ds-docker-registry-589fb466d5-tdltk            1/1     Running   0             35m
prod-ds-imgsvc-deploy-66dc945bf6-9k24k              1/1     Running   0             35m
prod-fss-alarmmgr-7b6ff65f97-mj5x8                  1/1     Running   0             35m
prod-fss-auth-57648dff7c-vv8j2                      1/1     Running   0             35m
```

```
prod-fss-catalog-75b6b87c69-rqg87                    1/1     Running   0             35m
prod-fss-cfggen-6df55b9fd9-hnj99                     1/1     Running   0             35m
prod-fss-cfgsync-6ccdd5474f-ppgfj                    1/1     Running   0             35m
prod-fss-connect-686589bb65-vx8wj                    1/1     Running   0             35m
prod-fss-da-58b4fbbd5-42d52                          1/1     Running   0             35m
prod-fss-da-58b4fbbd5-674tl                          1/1     Running   0             35m
prod-fss-da-58b4fbbd5-7vzw8                          1/1     Running   0             35m
prod-fss-da-58b4fbbd5-fskd8                          1/1     Running   0             35m
prod-fss-da-58b4fbbd5-lj5wz                          1/1     Running   0             35m
prod-fss-da-58b4fbbd5-plx6s                          1/1     Running   0             35m
prod-fss-da-58b4fbbd5-tg67x                          1/1     Running   0             35m
prod-fss-da-58b4fbbd5-w8gg7                          1/1     Running   0             35m
prod-fss-deviationmgr-acl-66c754c6cc-h7c25           1/1     Running   0             35m
prod-fss-deviationmgr-bfd-65c5ff9457-45sgz           1/1     Running   0             35m
prod-fss-deviationmgr-interface-58fc8bb77d-xfdlp     1/1     Running   0             35m
prod-fss-deviationmgr-netinst-b8cbccb9d-9gtl6        1/1     Running   0             35m
prod-fss-deviationmgr-platform-7c9f6bf67b-x96mn      1/1     Running   0             35m
prod-fss-deviationmgr-qos-6b97dcc884-8w5dv           1/1     Running   0             35m
prod-fss-deviationmgr-routingpolicy-67df5489cc-w6tpc 1/1     Running   0             35m
prod-fss-deviationmgr-system-67b85d8974-bgd8l        1/1     Running   0             35m
prod-fss-dhcp-7b77b55746-27mh7                       1/1     Running   0             35m
prod-fss-digitalsandbox-7c84dbfb85-wgngf             1/1     Running   0             35m
prod-fss-filemgr-6d6cfcc68b-4wz4j                    1/1     Running   0             35m
prod-fss-imagemgr-c8d7cdfb5-hsgnq                    1/1     Running   0             35m
prod-fss-intentmgr-858587457f-ppk6g                  1/1     Running   0             35m
prod-fss-inventory-69bdbf979-wp852                   1/1     Running   0             35m
prod-fss-labelmgr-56cf479f99-ljbkm                   1/1     Running   0             35m
prod-fss-maintmgr-569c964d89-4j7l7                   1/1     Running   0             35m
prod-fss-mgmtstack-7dcddf4479-hlpd8                  1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-28xj4                     1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-gpvd4                     1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-hslvh                     1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-nsbgd                     1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-qtplh                     1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-x7rgg                     1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-xd964                     1/1     Running   0             35m
prod-fss-oper-da-fdb879d95-xdczn                     1/1     Running   0             35m
prod-fss-oper-topomgr-598c95fbc4-cm5hl               1/1     Running   0             35m
prod-fss-protocolmgr-5d67f4548b-9vxs7                1/1     Running   0             35m
prod-fss-topomgr-5d8dd6597f-n76xq                    1/1     Running   0             35m
prod-fss-transaction-b678ff785-czgcz                 1/1     Running   0             35m
prod-fss-version-758dfd6c49-7wxvg                    1/1     Running   0             35m
prod-fss-web-59c677689-44dtr                         1/1     Running   0             35m
prod-fss-workloadmgr-7469c5b46b-whgdw                1/1     Running   0             35m
prod-fss-ztp-646cd8df76-9gmt4                        1/1     Running   0             35m
prod-keycloak-0                                      1/1     Running   0             35m
prod-mongodb-arbiter-0                               1/1     Running   0             35m
prod-mongodb-primary-0                               1/1     Running   0             35m
prod-mongodb-secondary-0                             1/1     Running   0             35m
prod-neo4j-core-0                                    1/1     Running   0             35m
prod-postgresql-0                                    1/1     Running   0             35m
prod-sftpserver-b68849d55-gsv7m                      1/1     Running   0             35m
```

Step 2. Back up the necessary files from the deployer VM.

Copy the following files to a directory that is outside of the deployer VM:

- SSH files

  `/root/.ssh/id_rsa`

  `/root/.ssh/id_rsa.pub`

  `/root/.ssh/known_hosts`

- the input JSON configuration and user values YAML file (if any) that were used to install the Fabric Services System, for example:

  `/root/customer-input.json`

  `/root/user_values.yaml`
- optional files; for example, SRL image tar files that were added to the old deployer VM

Step 3. Back up the Fabric Services System.

For instructions, see “Backup and restore” in the *Fabric Services System User Guide*.

Step 4. Shut down the old deployer VM.

Note: Ensure that the old deployer VM is not configured to automatically restart.

Step 5. Create a new deployer VM using the image for the new release.

For instructions, see Fabric Services System deployer VM creation. The new deployer should have the same network configuration and the same IP address of old deployer VM.

Step 6. Copy the files that you saved in Step 2 to the new deployer VM in the same locations.

What to do next

Performing a software upgrade

8.2 Performing a software upgrade

About this task

Use the **fss-upgrade.sh** script to perform the software upgrade steps. The command syntax is as follows:

```
[root@fss-deployer ~]# /root/bin/fss-upgrade.sh
Usage: /root/bin/fss-upgrade.sh configure <config-file> - Reads config file and configures
 accordingly.
 ./bin/fss-upgrade.sh discover - Discovers cluster.
 ./bin/fss-upgrade.sh upgrade - Upgrades FSS application.
 ./bin/fss-upgrade.sh status - Prints status of cluster and FSS Chart information.
 ./bin/fss-upgrade.sh clean - Cleans temporary files.
 ./bin/fss-upgrade.sh [help | -h] - Prints usage.
```

This script updates the entire Fabric Services System application using the new Helm charts.

Using the new deployer VM, upgrade the Fabric Services System application in place on the existing Fabric Services System cluster.

Prerequisites

Ensure that you have completed the steps described in Preparing for software upgrade. Execute the following steps while logged in to the new deployer VM that you created in Preparing for software upgrade.

Procedure

Step 1. Update the configuration.

Example

In the following example, the input configuration file is `customer-input.json`.

```
[root@fss-deployer ~]# /root/bin/fss-upgrade.sh configure customer-input.json
    Timesync service is running on 192.0.2.74  Time difference is 0 seconds
    Timesync service is running on 192.0.2.75  Time difference is 0 seconds
    Timesync service is running on 192.0.2.76  Time difference is 0 seconds
    Timesync service is running on 192.0.2.77  Time difference is 0 seconds
    Timesync service is running on 192.0.2.78  Time difference is 0 seconds
    Timesync service is running on 192.0.2.79  Time difference is 0 seconds
  Maximum time difference between nodes 0 seconds
  Successfully configured. Please run /root/bin/fss-upgrade.sh discover
```

Step 2. Discover the cluster.

Example

```
[root@fss-deployer ~]# /root/bin/fss-upgrade.sh discover
Trying to discover fss-node01 192.0.2.74
Trying to discover fss-node02 192.0.2.75
Trying to discover fss-node03 192.0.2.76
NAME STATUS ROLES AGE VERSION
fss-node01 Ready control-plane,master 14h v1.23.1
fss-node02 Ready control-plane,master 14h v1.23.1
fss-node03 Ready control-plane,master 14h v1.23.1
fss-node04 Ready <none> 14h v1.23.1
fss-node05 Ready <none> 14h v1.23.1
fss-node06 Ready <none> 14h v1.23.1
FSS release discovered fss-FSS_24_5_B1-charts-v24.5.2-57
```

Step 3. Upgrade the Fabric Services System application.

Example

```
[root@fss-deployer ~]# /root/bin/fss-upgrade.sh upgrade
NAME                  STATUS   ROLES                  AGE   VERSION
fss-node01            Ready    control-plane,master   75m   v1.23.1
fss-node02            Ready    control-plane,master   74m   v1.23.1
fss-node03            Ready    control-plane,master   74m   v1.23.1
fss-node04            Ready    <none>                 73m   v1.23.1
fss-node05            Ready    <none>                 73m   v1.23.1
fss-node06            Ready    <none>                 73m   v1.23.1
FSS will be upgraded from fss-FSS_24_5_B1-charts-v24.5.2-57 to fss-FSS_24_8_B1-charts-
v24.8.1-XX : Are you sure [YyNn]? y

Upgrade in progress...
Upgrading fss-logs
fss-logs release discovered: fluent-bit-0.20.9 ; Deployer packages fss-logs release:
 fluent-bit 0.39.0
Upgrading cert-manager
cert-manager release discovered: cert-manager-v1.12.0 ; Deployer packages cert-manager
 release: cert-manager v1.13.3
cert-manager upgrade started
Upgrading metallb
metallb release discovered: metallb-0.13.7 ; Deployer packages metallb release:
 metallb 0.13.7
metallb upgrade not required
Upgrading traefik and ingress routes
traefik release discovered: traefik-21.0.0 ; Deployer packages traefik release:
 traefik 26.0.0
Upgrading kafka and kafkaop if required
```

```
kafkaop release discovered: strimzi-kafka-operator-0.31.0 ; Deployer packages kafkaop
 release: strimzi-kafka-operator 0.37.0
NEW KAFKA operator, upgrade chart and update kafka
Waiting for fss services to stop
KAFKA will be deleted and reinstalled.
Upgrading / Installing new FSS Chart
Release "prod" has been upgraded. Happy Helming!
NAME: prod
LAST DEPLOYED: Tue  Aug 20 06:42:09 2024
NAMESPACE: default
STATUS: deployed
REVISION: 3
NOTES:
KAFKA being reinstalled.
Checking for FSS pods
Waiting for default fss-catalog 1 Running
Waiting for FSS pods to comeup
Waiting for default fss-version 1 Running
Waiting for FSS pods to comeup
All FSS pods are running
Checking for FSS digitalsandbox pods
FSS digital sandbox pods are running
Checking for digitalsandbox pods
Digital sandbox pods are running

FSS is started, it can take upto 10 mins to be ready for use
GUI can be accessed at https://192.0.2.74
```

Step 4. Verify that the upgrade is successful.

a. Check the status of the compute nodes.

Example

The STATUS field should show Ready.

```
[root@fss-deployer ~]# /root/bin/fss-upgrade.sh status
NAME             STATUS   ROLES                  AGE   VERSION
fss-node01       Ready    control-plane,master   27m   v1.23.1
fss-node02       Ready    control-plane,master   27m   v1.23.1
fss-node03       Ready    control-plane,master   27m   v1.23.1
fss-node04       Ready    <none>                 26m   v1.23.1
fss-node05       Ready    <none>                 26m   v1.23.1
fss-node06       Ready    <none>                 26m   v1.23.1
Installed FSS release: fss-FSS_24_8_B1-charts-v24.8.1-XXX ; Available FSS release:
 fss-FSS_24.8_B1-charts-v24.8.1-XXX
Checking for FSS pods
All FSS pods are running
Checking for FSS digitalsandbox pods
FSS digital sandbox pods are running
Checking for digitalsandbox pods
Digital sandbox pods are running

FSS is started, it can take upto 10 mins to be ready for use
GUI can be accessed at https://10.254.107.75
```

b. Display the Helm chart version.

Example

```
[root@fss-deployer ~]# export KUBECONFIG=/var/lib/fss/config.fss
[root@fss-deployer ~]# helm list
NAME NAMESPACE REVISION UPDATED STATUS CHART APP VERSION
fss-logs default 2 2024-08-20 06:35:35.469592459 +0000 UTC deployed fluent-bit-0.39.0 2.1.10
```

```
kafka default 1 2024-08-20 06:42:22.087340711 +0000 UTC deployed fss-strimzi-kafka-0.1.11 3.5.1
prod default 3 2024-08-20 06:42:09.940429218 +0000 UTC deployed fss-FSS_24_8_B1-charts-v24.8.1-XX
 v24.8.1
```

c. Check the status of system services and processes.

Expected outcome

```
[root@fss-deployer ~]# kubectl get pods -A
NAMESPACE NAME READY STATUS RESTARTS AGE
cert-manager cert-manager-7bff5cd474-hs5sf 1/1 Running 0 19m
cert-manager cert-manager-cainjector-c79788c78-r7tsg 1/1 Running 0 19m
cert-manager cert-manager-webhook-65f55dfd68-jv92l 1/1 Running 0 19m
default fss-logs-fluent-bit-485hn 1/1 Running 0 19m
default fss-logs-fluent-bit-8mw8p 1/1 Running 0 19m
default fss-logs-fluent-bit-8zvmb 1/1 Running 0 18m
default fss-logs-fluent-bit-hkxql 1/1 Running 0 18m
default fss-logs-fluent-bit-jp87k 1/1 Running 0 17m
default fss-logs-fluent-bit-mctvg 1/1 Running 0 18m
default prod-ds-apiserver-7db4cb486c-jjd72 1/1 Running 0 12m
default prod-ds-cli-6f9d474fd5-7vl2b 1/1 Running 0 12m
default prod-ds-docker-registry-5b467bbf67-w4cps 1/1 Running 0 11h
default prod-ds-imgsvc-deploy-88b6f5d74-c7kxz 1/1 Running 0 12m
default prod-fss-alarmmgr-75fbfcff7-jhvh7 1/1 Running 0 12m
default prod-fss-alertmgr-webhook-2spqg 1/1 Running 0 12m
default prod-fss-alertmgr-webhook-7m8xc 1/1 Running 0 11m
default prod-fss-alertmgr-webhook-cbckz 1/1 Running 0 12m
default prod-fss-alertmgr-webhook-gcxsp 1/1 Running 0 12m
default prod-fss-alertmgr-webhook-l7j6d 1/1 Running 0 12m
default prod-fss-alertmgr-webhook-r76d5 1/1 Running 0 12m
default prod-fss-auth-587bcdc997-sfg9w 1/1 Running 0 12m
default prod-fss-catalog-779bc9b857-t6jdf 1/1 Running 0 12m
default prod-fss-cfggen-5d74d7444c-7st2r 1/1 Running 0 12m
default prod-fss-cfgsync-7dc4fdf6bc-psxx7 1/1 Running 0 11m
default prod-fss-connect-6cf889b4fc-c2bg8 1/1 Running 0 12m
default prod-fss-da-0 1/1 Running 0 12m
default prod-fss-da-1 1/1 Running 0 12m
default prod-fss-da-2 1/1 Running 0 12m
default prod-fss-da-3 1/1 Running 0 12m
default prod-fss-da-4 1/1 Running 0 12m
default prod-fss-da-5 1/1 Running 0 12m
default prod-fss-da-6 1/1 Running 0 12m
default prod-fss-da-7 1/1 Running 0 12m
default prod-fss-deviationmgr-acl-c4f9fd588-nh2d4 1/1 Running 0 11m
default prod-fss-deviationmgr-bfd-778dd6f456-fxgn2 1/1 Running 0 11m
default prod-fss-deviationmgr-interface-8db5b55f5-cqgqz 1/1 Running 0 11m
default prod-fss-deviationmgr-netinst-56fb5b7887-rgjzg 1/1 Running 0 11m
default prod-fss-deviationmgr-platform-7f6d4f47d9-464jd 1/1 Running 0 11m
default prod-fss-deviationmgr-qos-75bd65f489-4n5nw 1/1 Running 0 12m
default prod-fss-deviationmgr-routingpolicy-5d74cf65cb-bhb5r 1/1 Running 0 11m
default prod-fss-deviationmgr-system-8f8887c67-glrzg 1/1 Running 0 11m
default prod-fss-dhcp-5d66876465-z8xf4 1/1 Running 0 12m
default prod-fss-dhcp6-65779bf496-d69qz 1/1 Running 0 12m
default prod-fss-digitalsandbox-68576766d5-tvvt5 1/1 Running 0 11m
default prod-fss-filemgr-74965868d9-mbfl5 1/1 Running 0 11m
default prod-fss-imagemgr-55445fbbb4-pdbm2 1/1 Running 0 11m
default prod-fss-intentmgr-565fc64855-hn8kh 1/1 Running 0 11m
default prod-fss-inventory-6bf79df9d5-z2jcj 1/1 Running 0 11m
default prod-fss-labelmgr-998d95bd-sln88 1/1 Running 0 11m
default prod-fss-maintmgr-5f4b4ff745-ndhd2 1/1 Running 0 11m
default prod-fss-mgmtstack-f959df45d-2nsj8 1/1 Running 0 11m
default prod-fss-oper-da-0 1/1 Running 0 12m
default prod-fss-oper-da-1 1/1 Running 0 12m
default prod-fss-oper-da-2 1/1 Running 0 12m
```

```
default prod-fss-oper-da-3 1/1 Running 0 12m
default prod-fss-oper-da-4 1/1 Running 0 12m
default prod-fss-oper-da-5 1/1 Running 0 12m
default prod-fss-oper-da-6 1/1 Running 0 12m
default prod-fss-oper-da-7 1/1 Running 0 12m
default prod-fss-oper-topomgr-56cbcb7586-xp8fx 1/1 Running 0 11m
default prod-fss-protocolmgr-75f866f5d-tjtdv 1/1 Running 0 11m
default prod-fss-syncmgr-f5784f8df-q9f2l 1/1 Running 0 12m
default prod-fss-topomgr-75d4656f45-wlnn2 1/1 Running 0 11m
default prod-fss-transaction-55ccfbd46f-9nvwv 1/1 Running 0 11m
default prod-fss-version-7dd5c7877-wrq5z 1/1 Running 0 6m53s
default prod-fss-web-6674c4d54c-4hgvb 1/1 Running 0 11m
default prod-fss-workloadmgr-7bc56d8556-dx4nw 1/1 Running 0 11m
default prod-fss-ztp-6687c57b75-695zd 1/1 Running 0 11m
default prod-keycloak-0 1/1 Running 0 12m
default prod-mongodb-arbiter-0 1/1 Running 0 12m
default prod-mongodb-primary-0 1/1 Running 0 12m
default prod-mongodb-secondary-0 1/1 Running 0 12m
default prod-neo4j-core-0 1/1 Running 0 11h
default prod-postgresql-0 1/1 Running 0 12m
default prod-sftpserver-77cd8696d5-hxwwg 1/1 Running 0 11h
digital-sandbox-system prod-netsegmd-2fdk7 1/1 Running 0 12m
digital-sandbox-system prod-netsegmd-cj7xz 1/1 Running 0 12m
digital-sandbox-system prod-netsegmd-f64wm 1/1 Running 0 11m
digital-sandbox-system prod-netsegmd-gdc9t 1/1 Running 0 9m1s
digital-sandbox-system prod-netsegmd-pl7mn 1/1 Running 0 12m
digital-sandbox-system prod-netsegmd-wwwqs 1/1 Running 0 12m
digital-sandbox-system srlfabrics-controller-manager-5dd4fc9555-br5wl 2/2 Running 0 12m
kafka kafka-fss-entity-operator-8fc655656-lnkvp 3/3 Running 0 8m31s
kafka kafka-fss-kafka-0 1/1 Running 0 10m
kafka kafka-fss-kafka-1 1/1 Running 0 10m
kafka kafka-fss-kafka-2 1/1 Running 0 10m
kafka kafka-fss-zookeeper-0 1/1 Running 0 12m
kafka kafka-fss-zookeeper-1 1/1 Running 0 12m
kafka kafka-fss-zookeeper-2 1/1 Running 0 12m
kafka strimzi-cluster-operator-5dbd754b4d-57dhs 1/1 Running 0 19m
kube-system coredns-7947bbfb58-nk9xn 1/1 Running 0 2d14h
kube-system coredns-7947bbfb58-sbdlb 1/1 Running 0 2d14h
kube-system dns-autoscaler-74d7b6c6c-cw4cw 1/1 Running 0 2d14h
kube-system kube-apiserver-blrsrlfsphw01-compute1 1/1 Running 1 2d14h
kube-system kube-apiserver-blrsrlfsphw01-compute2 1/1 Running 1 2d14h
kube-system kube-apiserver-blrsrlfsphw01-compute3 1/1 Running 1 2d14h
kube-system kube-controller-manager-blrsrlfsphw01-compute1 1/1 Running 1 2d14h
kube-system kube-controller-manager-blrsrlfsphw01-compute2 1/1 Running 1 2d14h
kube-system kube-controller-manager-blrsrlfsphw01-compute3 1/1 Running 1 2d14h
kube-system kube-flannel-b6s2c 1/1 Running 0 2d14h
kube-system kube-flannel-bktxx 1/1 Running 0 2d14h
kube-system kube-flannel-d7n6h 1/1 Running 0 2d14h
kube-system kube-flannel-gbvdn 1/1 Running 0 2d14h
kube-system kube-flannel-hmb56 1/1 Running 0 2d14h
kube-system kube-flannel-wcsmg 1/1 Running 0 2d14h
kube-system kube-proxy-6czd6 1/1 Running 0 2d14h
kube-system kube-proxy-g8zgl 1/1 Running 0 2d14h
kube-system kube-proxy-l7zzt 1/1 Running 0 2d14h
kube-system kube-proxy-m6h9g 1/1 Running 0 2d14h
kube-system kube-proxy-ntt57 1/1 Running 0 2d14h
kube-system kube-proxy-xjd68 1/1 Running 0 2d14h
kube-system kube-scheduler-blrsrlfsphw01-compute1 1/1 Running 1 2d14h
kube-system kube-scheduler-blrsrlfsphw01-compute2 1/1 Running 1 2d14h
kube-system kube-scheduler-blrsrlfsphw01-compute3 1/1 Running 1 2d14h
kube-system nginx-proxy-blrsrlfsphw01-compute4 1/1 Running 0 2d14h
kube-system nginx-proxy-blrsrlfsphw01-compute5 1/1 Running 0 2d14h
kube-system nginx-proxy-blrsrlfsphw01-compute6 1/1 Running 0 2d14h
kube-system snapshot-controller-55dbff48d-jxp4f 1/1 Running 0 2d14h
```

```
kube-system snapshot-controller-55dbff48d-zshwb 1/1 Running 0 2d14h
kube-system traefik-2zqqf 1/1 Running 0 19m
kube-system traefik-dkmbv 1/1 Running 0 19m
kube-system traefik-jhsxw 1/1 Running 0 19m
kube-system traefik-mdwcx 1/1 Running 0 19m
kube-system traefik-smgvx 1/1 Running 0 19m
kube-system traefik-tcjgd 1/1 Running 0 19m
metallb-system metallb-controller-b99bbcfff-vnz96 1/1 Running 0 2d14h
metallb-system metallb-speaker-drgcj 1/1 Running 0 2d14h
metallb-system metallb-speaker-fw2ng 1/1 Running 0 2d14h
metallb-system metallb-speaker-ghsv7 1/1 Running 0 2d14h
metallb-system metallb-speaker-nj5hj 1/1 Running 0 2d14h
metallb-system metallb-speaker-q4bbs 1/1 Running 0 2d14h
metallb-system metallb-speaker-zk7mg 1/1 Running 0 2d14h
rook-ceph csi-cephfsplugin-5jrj6 3/3 Running 0 2d14h
rook-ceph csi-cephfsplugin-6mdf9 3/3 Running 0 2d14h
rook-ceph csi-cephfsplugin-p2g7j 3/3 Running 0 2d14h
rook-ceph csi-cephfsplugin-provisioner-59dd67b6c-8gwzv 6/6 Running 0 2d14h
rook-ceph csi-cephfsplugin-provisioner-59dd67b6c-9j5h6 6/6 Running 0 2d14h
rook-ceph csi-cephfsplugin-ps8wz 3/3 Running 0 2d14h
rook-ceph csi-cephfsplugin-rbbr5 3/3 Running 0 2d14h
rook-ceph csi-cephfsplugin-smpxx 3/3 Running 0 2d14h
rook-ceph rook-ceph-mds-ceph-filesystem-a-55cfd884fc-2m2ds 1/1 Running 0 2d14h
rook-ceph rook-ceph-mds-ceph-filesystem-b-7568464675-ng266 1/1 Running 0 2d14h
rook-ceph rook-ceph-mgr-a-5888bc4d88-qpvbr 2/2 Running 0 2d14h
rook-ceph rook-ceph-mgr-b-58d9f87688-km8dl 2/2 Running 0 2d14h
rook-ceph rook-ceph-mon-a-5d45554b5c-5dbgh 1/1 Running 0 2d14h
rook-ceph rook-ceph-mon-b-b6d489fc8-ssfkh 1/1 Running 0 2d14h
rook-ceph rook-ceph-mon-c-5869bf7b69-k7lr8 1/1 Running 0 2d14h
rook-ceph rook-ceph-operator-77cf896f-tw6zd 1/1 Running 0 2d14h
rook-ceph rook-ceph-osd-0-6dbfd94778-5ttjz 1/1 Running 0 2d14h
rook-ceph rook-ceph-osd-1-6d7679c466-76x2c 1/1 Running 0 2d14h
rook-ceph rook-ceph-osd-2-78949bfc94-zbt9n 1/1 Running 0 2d14h
rook-ceph rook-ceph-osd-prepare-blrsrlfsphw01-compute1-86f4f 0/1 Completed 0 8h
rook-ceph rook-ceph-osd-prepare-blrsrlfsphw01-compute2-47jb6 0/1 Completed 0 8h
rook-ceph rook-ceph-osd-prepare-blrsrlfsphw01-compute3-58xmq 0/1 Completed 0 8h
rook-ceph rook-ceph-tools-c859b5678-2qhzs 1/1 Running 0 2d14h
```

Step 5. Restore previously added software catalog images.

The upgrade procedure replaces the software catalog. Changes that you made to the software catalog are lost after the upgrade. If you previously added new SR Linux images to the software catalog, you need to modify the software catalog and then upload the SR Linux images. For instructions, see “Adding a network operating system version to the software catalog” in the *Fabric Services System User Guide*. You must complete this step before you can access the Fabric Services System to make any change.

What to do next

Upgrading the Fabric Services System cluster

8.3 Upgrading the Fabric Services System cluster

About this task

After you complete the procedure Performing a software upgrade, the Fabric Services System software is upgraded to the new release.

Procedure

Step 1. Back up the Fabric Services System.

For instructions, see “Backup and restore” in the *Fabric Services System User Guide*.

Step 2. Make a copy of the `customer-input.json` file, certificates, SR Linux images and other user-created files from the old deployer VM to a safe location.

Step 3. Delete all the compute VMs.

Step 4. Recreate the compute VM with the Release 25.8 base image.

The number of compute VMs, hostname and IP address details should be same as the previous Fabric Services System cluster.

Step 5. Copy the `customer-input.json` file, certificates, SR Linux images and other user-created files to the new deployer VM.

Step 6. Reset the old cluster configurations.

Log in to the deployer VM running the new release and execute the **/root/bin/fss-install.sh reset** command.

Step 7. Set up key-based authentication from the deployer VM to all the compute VMs.

Example

If password authentication is enabled on the node for SSH, enter the following command from the deployer VM:

```
# ssh-copy-id root@<node IP/FQDN>
```

Step 8. Restore the backup that you created in Step 1.

For instructions, see “Backup and restore” in the *Fabric Services System User Guide*.

What to do next

To configure the LDAP settings or remote syslog capabilities, complete the following procedures in the *Fabric Services System User Guide*:

- “Configuring LDAP after software upgrade”
- “Configuring a remote syslog server for user audit logs”

To bring-up the system in geo-redundant configuration, see Deploying a geo-redundant system.

8.4 Deploying a geo-redundant system

About this task

If you want to bring up geo-redundant system after the software upgrade, complete the following steps. The newly upgraded system is considered the active system.

Procedure

Step 1. Ensure that the upgraded system is up and running.

Check the Fabric Services System pod status and the fabric status.

Step 2. Install another cluster, the standby cluster.

Follow the instructions described in this document, starting with Installation overview.

Note: The standby cluster must of the same size and the same software version as the active system.

Step 3. Verify the network connectivity between the active and standby clusters and the connectivity with the SR Linux network.

Step 4. Follow the instructions in “Geo-redundancy” in the *Fabric Services System User Guide*.

8.5 Performing a software rollback

About this task

If a software upgrade is unsuccessful, use the following steps to roll back from the new release to the old release.

Procedure

Step 1. Shut down and delete the deployer VM and Fabric Services System virtual machine nodes for the upgraded deployment.

Step 2. Deploy and configure the deployer VM with the old release.

For instructions, see the procedures in The Fabric Services System deployer VM.

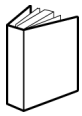
Step 3. Deploy new base virtual machine nodes.

For instructions, see Preparing the Fabric Services System virtual machine nodes.

Step 4. Restore the saved backup of the old release.

For instructions, see “Backup and restore” in the *Fabric Services System User Guide*.

Customer document and product support



Customer documentation

Customer documentation welcome page

Technical support

Product support portal

Documentation feedback

Customer documentation feedback