Cisco Optical Network Controller 24.3.x Installation Guide

First Published: 2024-02-14

Last Modified: 2024-04-05

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

CHAPTER 1

Install Cisco Optical Network Controller Using VMware vSphere

Installation Requirements

The following list contains the pre-requisites of Cisco Optical Network Controller 24.3.1 installation.

- Before installing Cisco Optical Network Controller 24.3.1, you must first login in to the VMware customer center and download VMware vCenter server version 7.0, as well as vSphere server and client with version 7.0. Cisco Optical Network Controller 24.3.1 is deployed on rack or blade servers within vSphere.
- Install ESXi host version of 7.0 or higher on the servers to support creating Virtual Machines.
- You must have a DNS server. The DNS server can be an internal DNS server if the Cisco Optical Network Controller instance is not exposed to the internet.
- You must have an NTP server or NTP Pool for time synchronization.
- Before the Cisco Optical Network Controller 24.3.1 installation, three networks must be created.
  - **Control Plane Network**:

    The control plane network helps in the internal communication between the deployed VMs within a cluster. If you are setting up a standalone system, this can refer to any private network.
  - **VM Network or Northbound Network**:

    The VM network is used for communication between the user and the cluster. It handles all the traffic to and from the VMs running on your ESXi hosts and this is your Public network through which the UI is hosted.
  - **Eastbound Network**:

    The Eastbound Network helps in the internal communication between the deployed VMs within a cluster. If you are setting up a standalone system, this can refer to any private network.

Note For more details on VMware vSphere, see *VMware vSphere*.

The minimum requirement for Cisco Optical Network Controller 24.3.1 installation is given in the table below.

Table 1: Minimum Requirement

Sizing	CPU	Memory	Disk
XS	16 vCPU	64 GB	800 GB
S	32 vCPU	128 GB	1536 GB

The requirements based on type of deployment are given in the table below.

Table 2: Deployment Requirements

Deployment Type	Requirements
Standalone (SA)	**Control Plane Network:** Can be a private network for standalone setups. Requires 1 IP address. **Gateway:** Required. **DNS Server:** Should be an internal DNS if the node is not exposed to the internet; otherwise, an internet DNS can be used. **Northbound Network (VM Network):** Should be a public network. All communication between the Cisco Optical Network Controller and devices will flow through this network. Requires 1 public IP address. **Gateway:** Required. **DNS Server:**Required. Should be an internal DNS if the node is not exposed to the internet; otherwise, an internet DNS can be used. **Eastbound Network:** Can be a private network for standalone setups. Requires 1 private IP address. **Gateway:** Required. **DNS Server:**Required. Should be an internal DNS if the node is not exposed to the internet; otherwise, an internet DNS can be used.

To create the control plane and virtual management networks follow the steps listed below.

1. From the vSphere client, select the Datacenter where you want to add the ESXi host.
2. Right-click the server from the vCenter inventory and click **Add Networking**.
3. To create a private network for Control Plane and Eastbound Networks, follow the wizard for a Standard Switch addition for each network.
   a. In **Select connection type**, choose **Virtual Machine Port Group for a Standard Switch** and click **Next**.
   b. In **Select target device** , select **New Standard Switch (MTU 1500)** and click **Next**.

c. In **Create a Standard Switch**, click **Next**, and confirm *There are no active physical network adapters for the switch.*

d. In **Connection settings** choose a network label (Control Plane or Eastbound) and select VLAN ID as None(0) click **Next**.

e. In **Ready to complete**, review your configuration and click **Finish**.

After adding the ESXi host, create the Control Plane, Northbound, and Eastbound Networks before deploying.

SSH Key Generation

For accessing SSH, ed25519 key is required. The ed25519 key is different from the RSA key.

Use the CLI given below to generate the ed25519 key.

```
ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/Users/xyz/.ssh/id_ed25519):
./<file-name-of-your-key>.pem
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ./<file-name-of-your-key>.pem
Your public key has been saved in ./<file-name-of-your-key>.pem.pub
The key fingerprint is:
SHA256:zGW6aGn8rxvEq82sA/97jOaHrl9rnoTaYi+TqU3MeRU xyz@abc
The key's randomart image is:
+--[ED25519 256]--+
|                 |
|                 |
|          E      |
|       + + .     |
|         S .     |
|     .+ = =      |
|      o@o*+o     |
|      =XX++=o    |
|     .o*#/X=     |
+----[SHA256]-----+

#Once created you can cat the file with .pub extension for the public key. ( ex:
<file-name-of-your-key>.pem.pub )

cat <file-name-of-your-key>.pem.pub
#The above key has to be used in the deployment template ( SSH Public Key ) in the Deployment
 process
```

Install Cisco Optical Network Controller Using VMware vSphere

To deploy the OVA template, follow the steps given below.

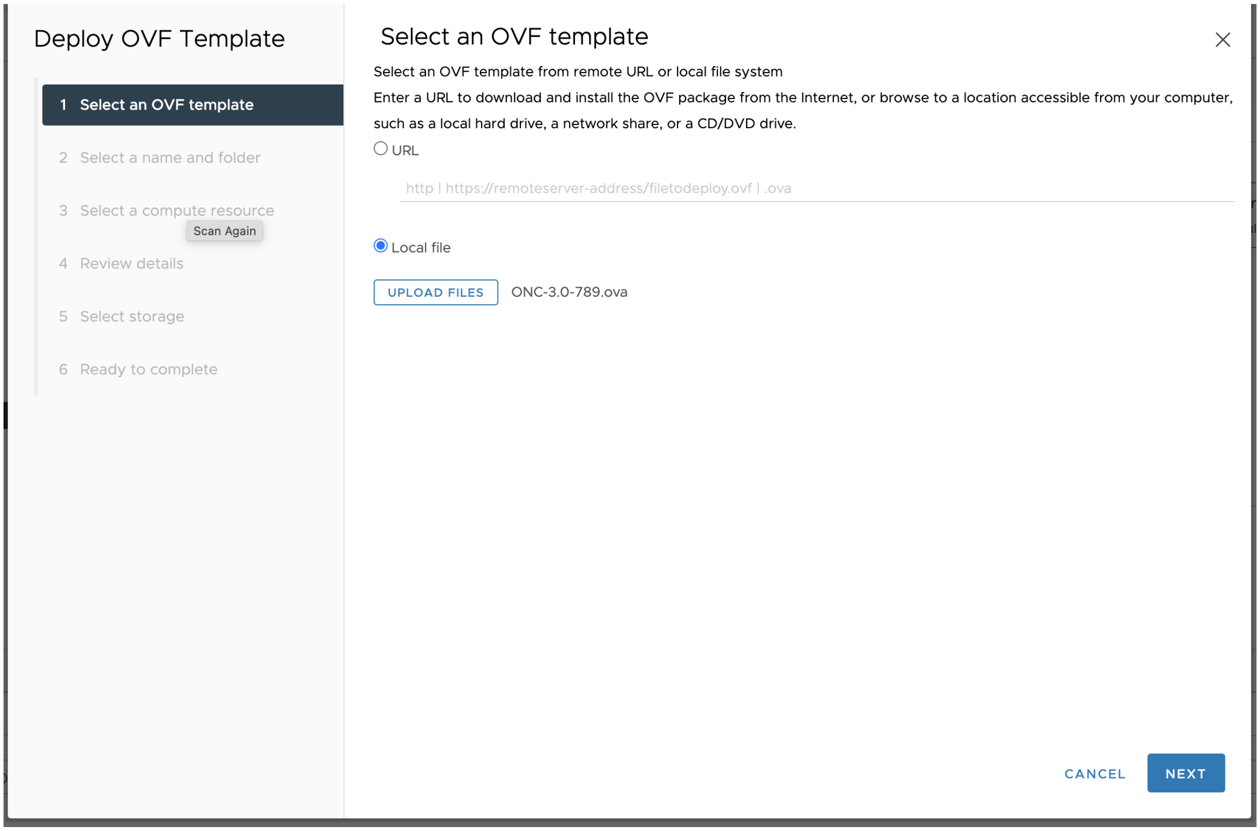
Before you begin

Note During the OVF deployment, the deployment gets aborted if there is an internet disconnection.

Step 1 Right click the ESXi host in the vSphere client screen and click **Deploy OVF Template**.

Step 2 In the **Select an OVF template** screen, select the **URL** radio button for specifying the URL to to download and install the OVF package from the Internet or select the **Local file** radio button to upload the downloaded ova files from your local system and click **Next.**

Figure 1: Select an OVF Template



Select an OVF template from remote URL or local file system

Enter a URL to download and install the OVF package from the Internet, or browse to a location accessible from your computer, such as a local hard drive, a network share, or a CD/DVD drive.

○ URL

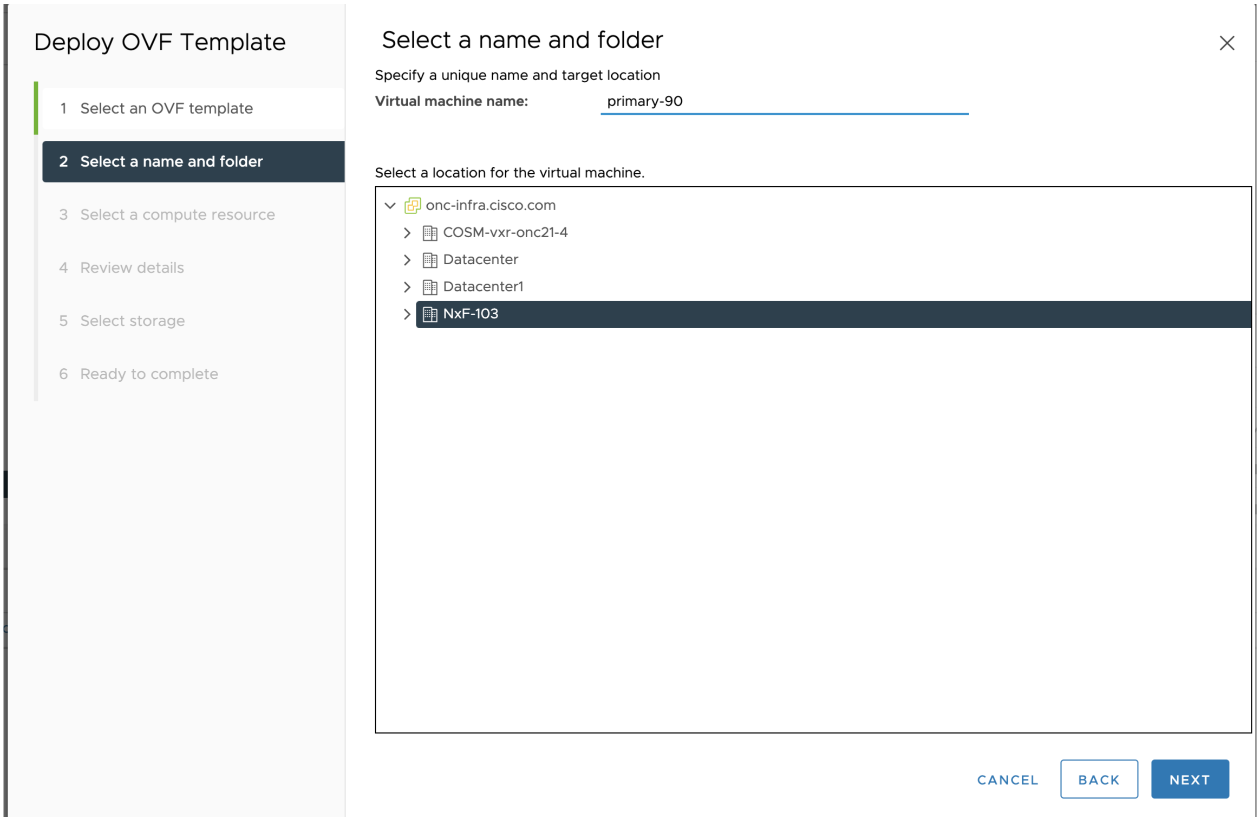
http | https://remoteserver-address/filetodeploy.ovf | .ova

● Local file

ONC-3.0-789.ova

Step 3 In the **Select a name and folder** screen, specify a unique name for the virtual machine Instance. From the list of options, select the location of the VM to be used and click **Next.**

Figure 2: Select a name and folder



Select a name and folder

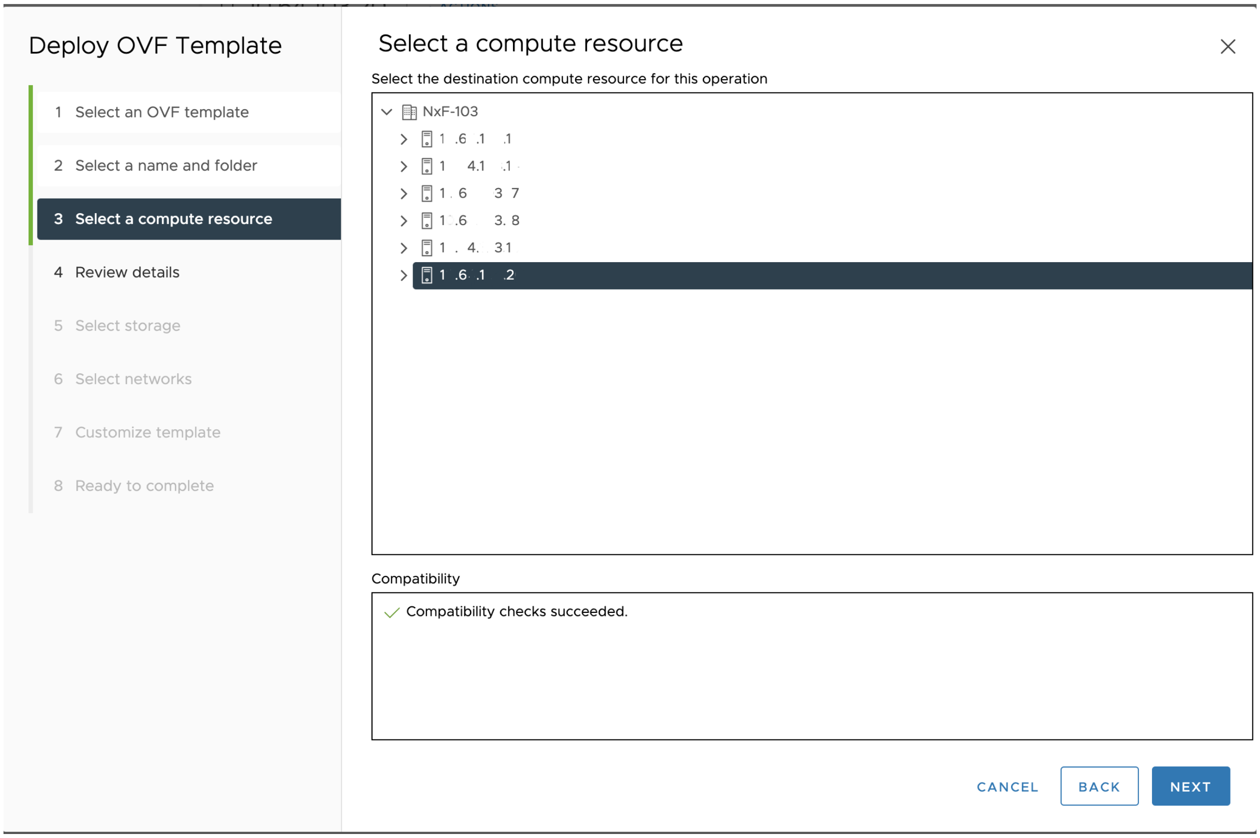
Specify a unique name and target location

Virtual machine name: primary-90

Select a location for the virtual machine.

Step 4 In the **Select a compute resource** screen, select the destination compute resource on which you want to deploy the VM and click **Next.**

Figure 3: Select a Compute Resource



Deploy OVF Template

1 Select an OVF template
2 Select a name and folder
3 Select a compute resource
4 Review details
5 Select storage
6 Select networks
7 Customize template
8 Ready to complete

Select a compute resource

Select the destination compute resource for this operation

NxF-103
1 .6 .1 .1
1 4.1 .1
1 . 6 3 7
1 .6 3. 8
1 . 4. 3 1
1 .6 .1 .2

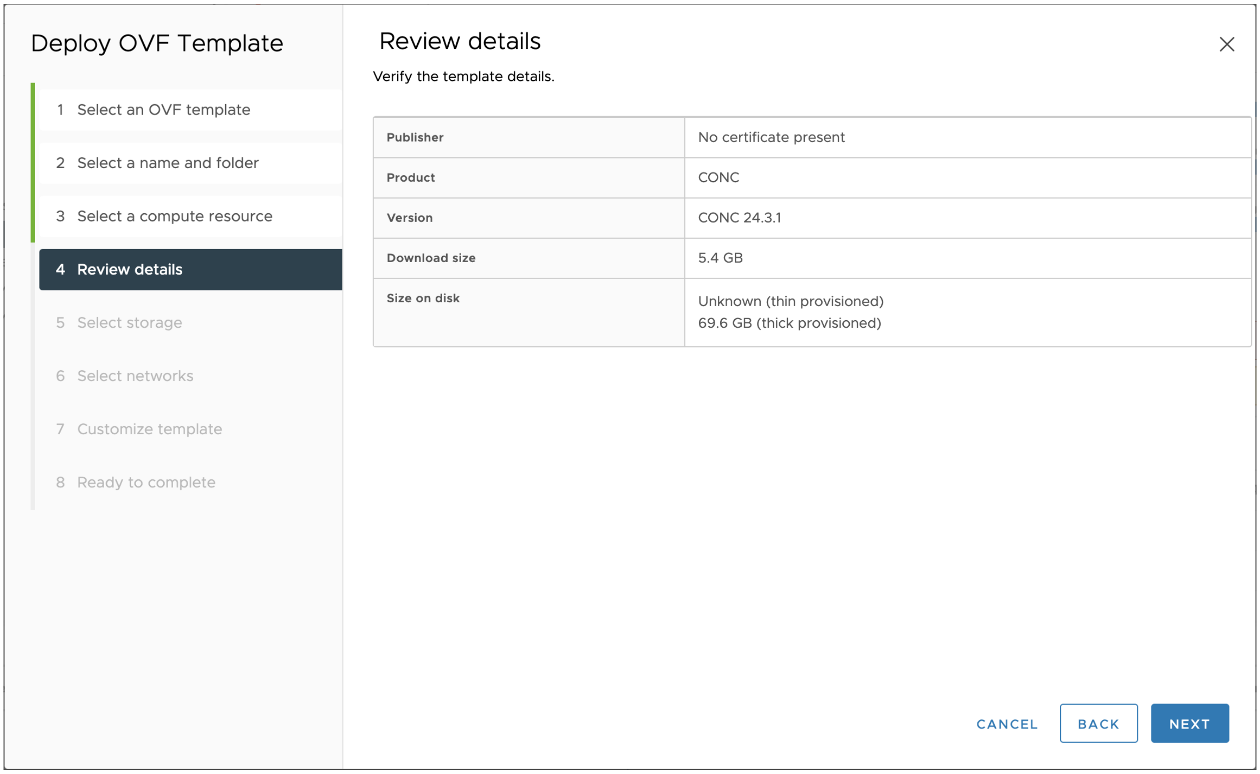
Compatibility

✓ Compatibility checks succeeded.

Note While selecting the compute resource the compatibility check proceeds till it completes successfully.

Step 5 In the **Review details** screen, verify the template details and click **Next**.

Figure 4: Review Details



Deploy OVF Template

1 Select an OVF template
2 Select a name and folder
3 Select a compute resource
4 Review details
5 Select storage
6 Select networks
7 Customize template
8 Ready to complete

Review details

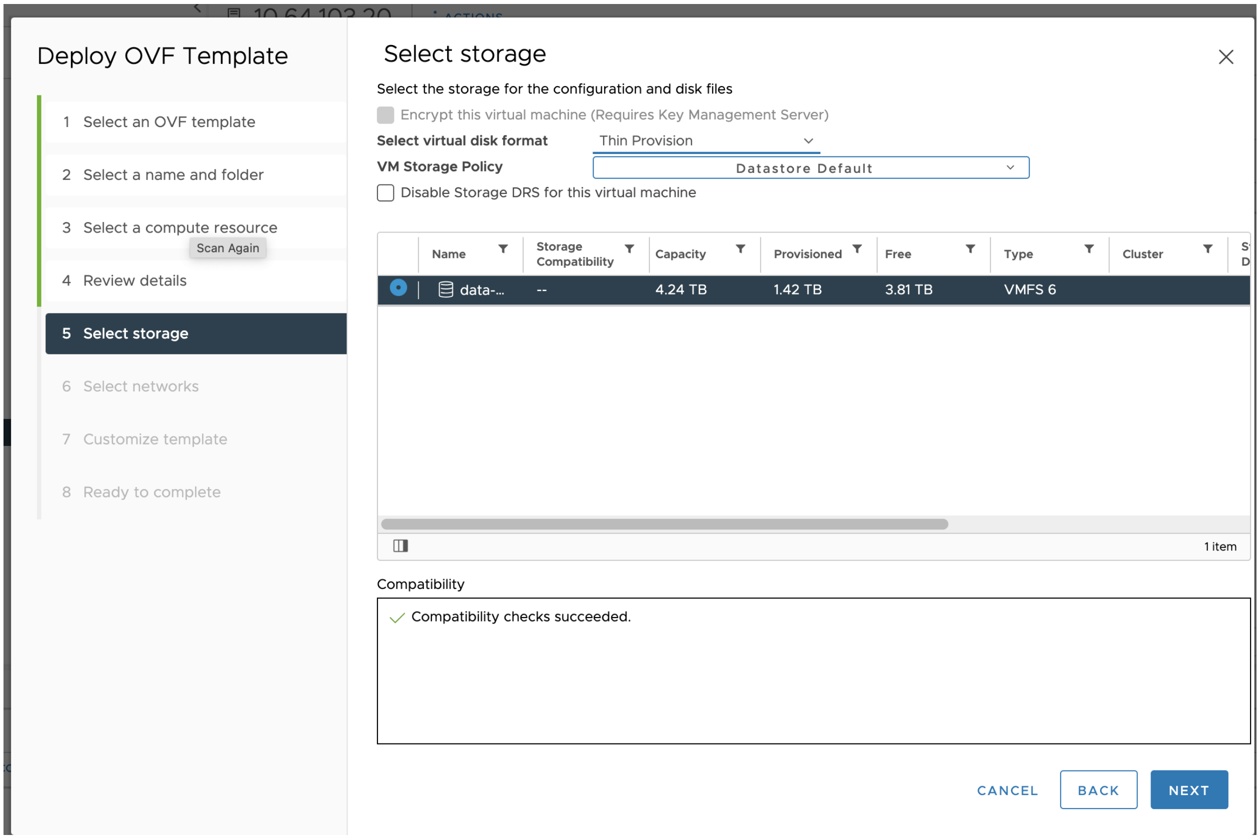
Verify the template details.

Publisher	No certificate present
Product	CONC
Version	CONC 24.3.1
Download size	5.4 GB
Size on disk	Unknown (thin provisioned) 69.6 GB (thick provisioned)

Step 6 In the Select storage screen, select the virtual disk format based on provision type requirement. **VM Storage Policy** is set as *Datastore Default* and click **Next**. Select the **virtual disk format** as *Thin Provision*.

You must select "Thin provision" as the virtual disk format.

Figure 5: Select Storage



Deploy OVF Template

1 Select an OVF template
2 Select a name and folder
3 Select a compute resource
4 Review details
5 Select storage
6 Select networks
7 Customize template
8 Ready to complete

Select storage

Select the storage for the configuration and disk files

□ Encrypt this virtual machine (Requires Key Management Server)

Select virtual disk format Thin Provision

VM Storage Policy Datastore Default

□ Disable Storage DRS for this virtual machine

	Name	Storage Compatibility	Capacity	Provisioned	Free	Type	Cluster
●	data-...	--	4.24 TB	1.42 TB	3.81 TB	VMFS 6	

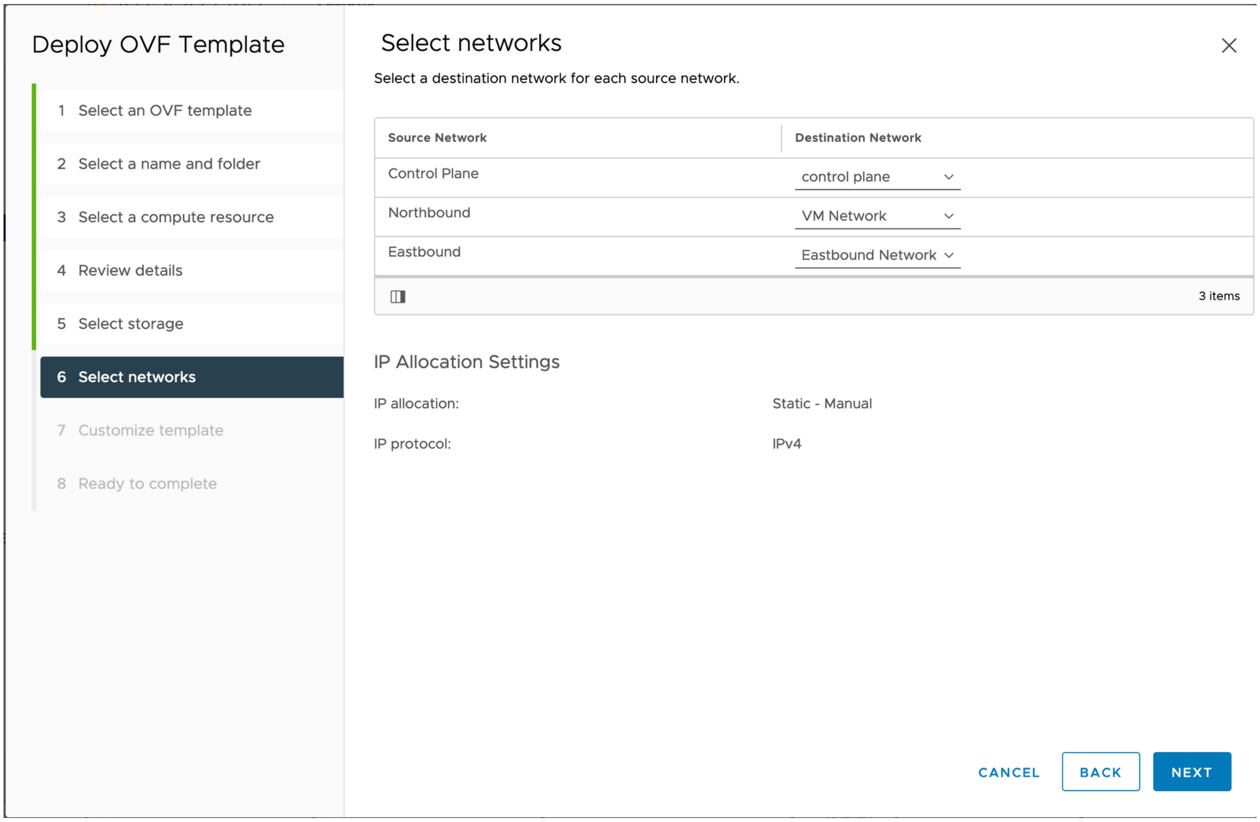
1 item

Compatibility

✓ Compatibility checks succeeded.

Step 7 In the **Select networks** screen, select the control and management networks as **Control Plane, Eastbound,** and **Northbound** from the networks created earlier and **click Next**.

Figure 6: Select Networks



Deploy OVF Template

Select networks

Select a destination network for each source network.

Source Network	Destination Network
Control Plane	control plane
Northbound	VM Network
Eastbound	Eastbound Network

3 items

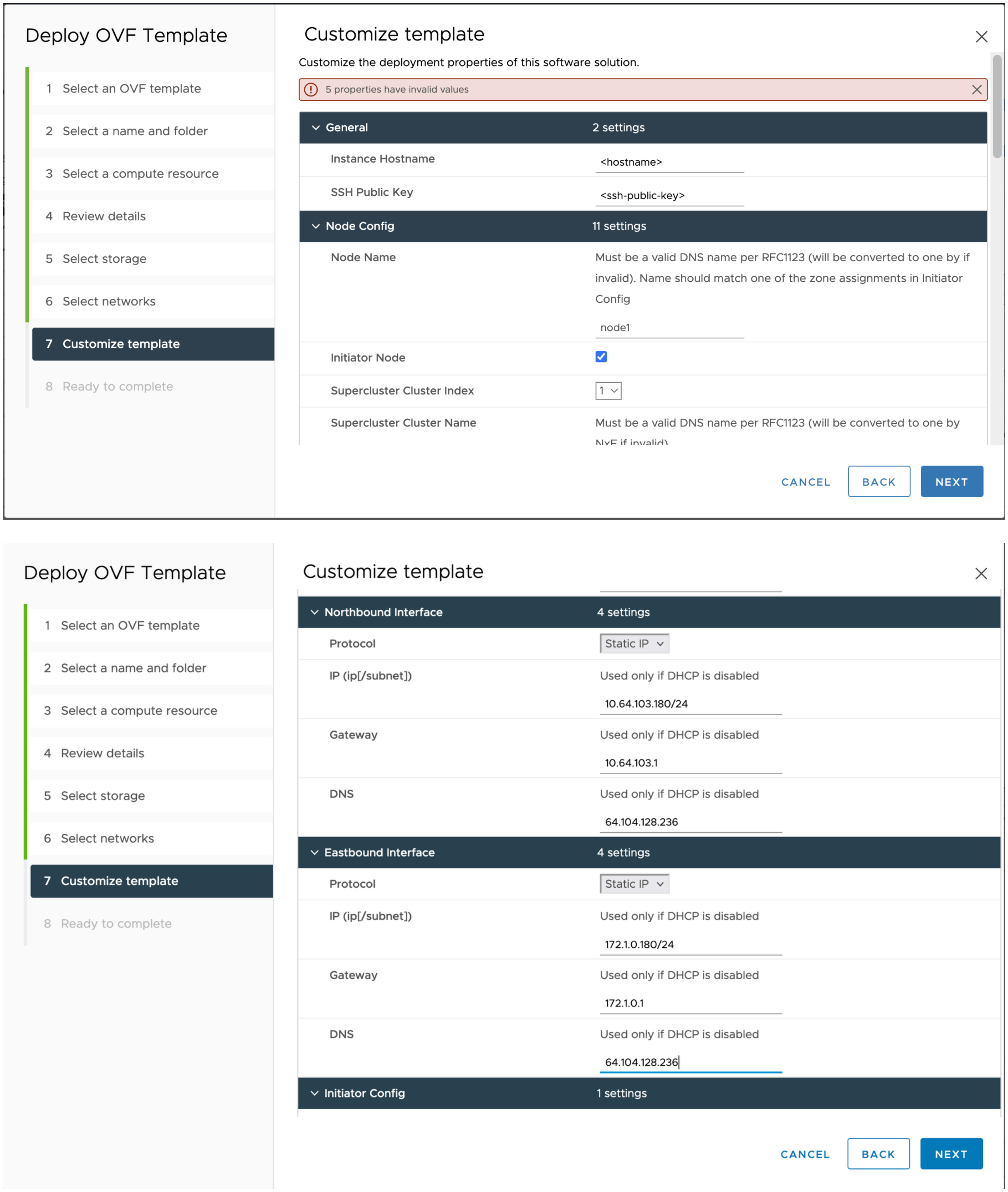
IP Allocation Settings

IP allocation: Static - Manual

IP protocol: IPv4

Step 8 In the **Customize template** screen, set the values using the following table as a guideline for deployment.

Figure 7: Customize Template



Deploy OVF Template

1 Select an OVF template
2 Select a name and folder
3 Select a compute resource
4 Review details
5 Select storage
6 Select networks
7 Customize template
8 Ready to complete

Customize template

Customize the deployment properties of this software solution.

5 properties have invalid values

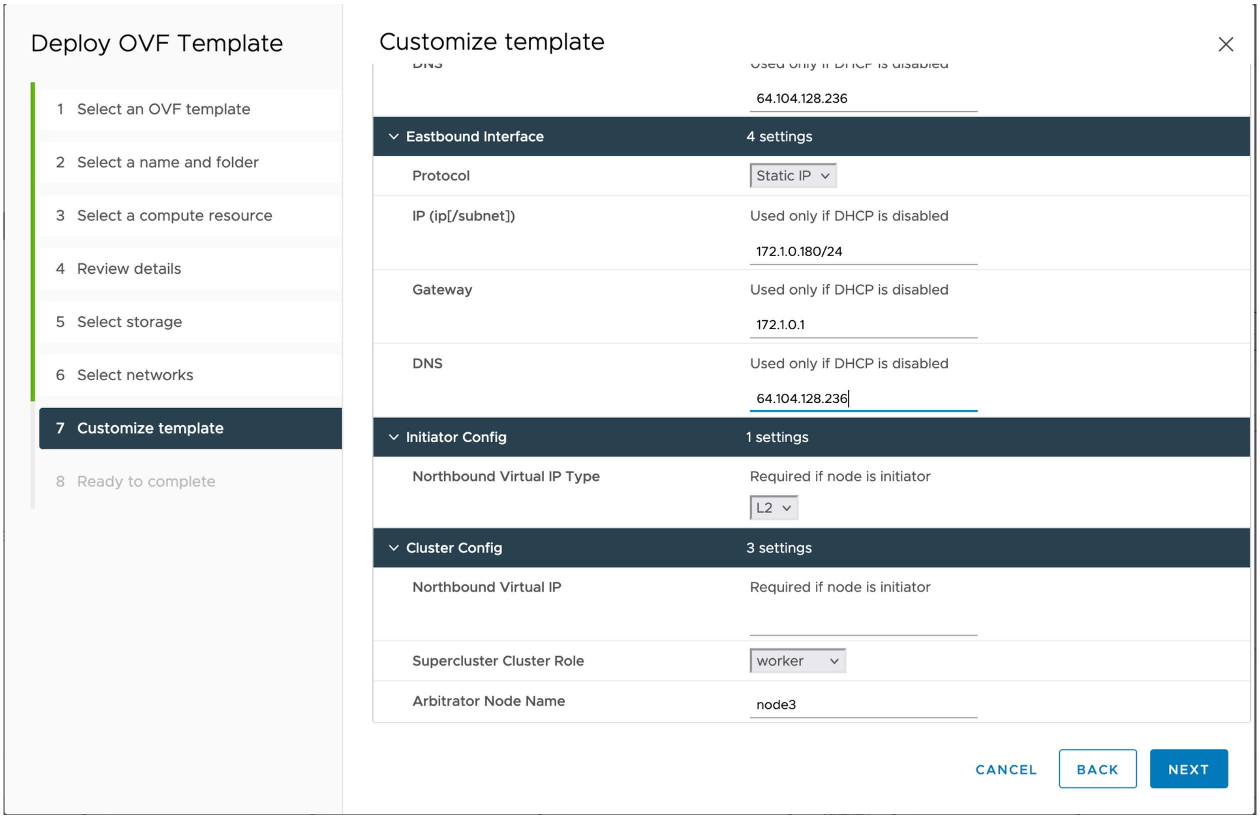
General	2 settings
Instance Hostname	<hostname>
SSH Public Key	<ssh-public-key>
Node Config	11 settings
Node Name	Must be a valid DNS name per RFC1123 (will be converted to one by if invalid). Name should match one of the zone assignments in Initiator Config node1
Initiator Node	☑
Supercluster Cluster Index	1
Supercluster Cluster Name	Must be a valid DNS name per RFC1123 (will be converted to one by NxF if invalid)

Deploy OVF Template

1 Select an OVF template
2 Select a name and folder
3 Select a compute resource
4 Review details
5 Select storage
6 Select networks
7 Customize template
8 Ready to complete

Customize template

Northbound Interface	4 settings
Protocol	Static IP
IP (ip[/subnet])	Used only if DHCP is disabled 10.64.103.180/24
Gateway	Used only if DHCP is disabled 10.64.103.1
DNS	Used only if DHCP is disabled 64.104.128.236
Eastbound Interface	4 settings
Protocol	Static IP
IP (ip[/subnet])	Used only if DHCP is disabled 172.1.0.180/24
Gateway	Used only if DHCP is disabled 172.1.0.1
DNS	Used only if DHCP is disabled 64.104.128.236
Initiator Config	1 settings



DNS — Used only if DHCP is disabled — 64.104.128.236

Eastbound Interface — 4 settings

Protocol	Static IP
IP (ip[/subnet])	Used only if DHCP is disabled 172.1.0.180/24
Gateway	Used only if DHCP is disabled 172.1.0.1
DNS	Used only if DHCP is disabled 64.104.128.236

Initiator Config — 1 settings

Northbound Virtual IP Type	Required if node is initiator L2

Cluster Config — 3 settings

Northbound Virtual IP	Required if node is initiator
Supercluster Cluster Role	worker
Arbitrator Node Name	node3

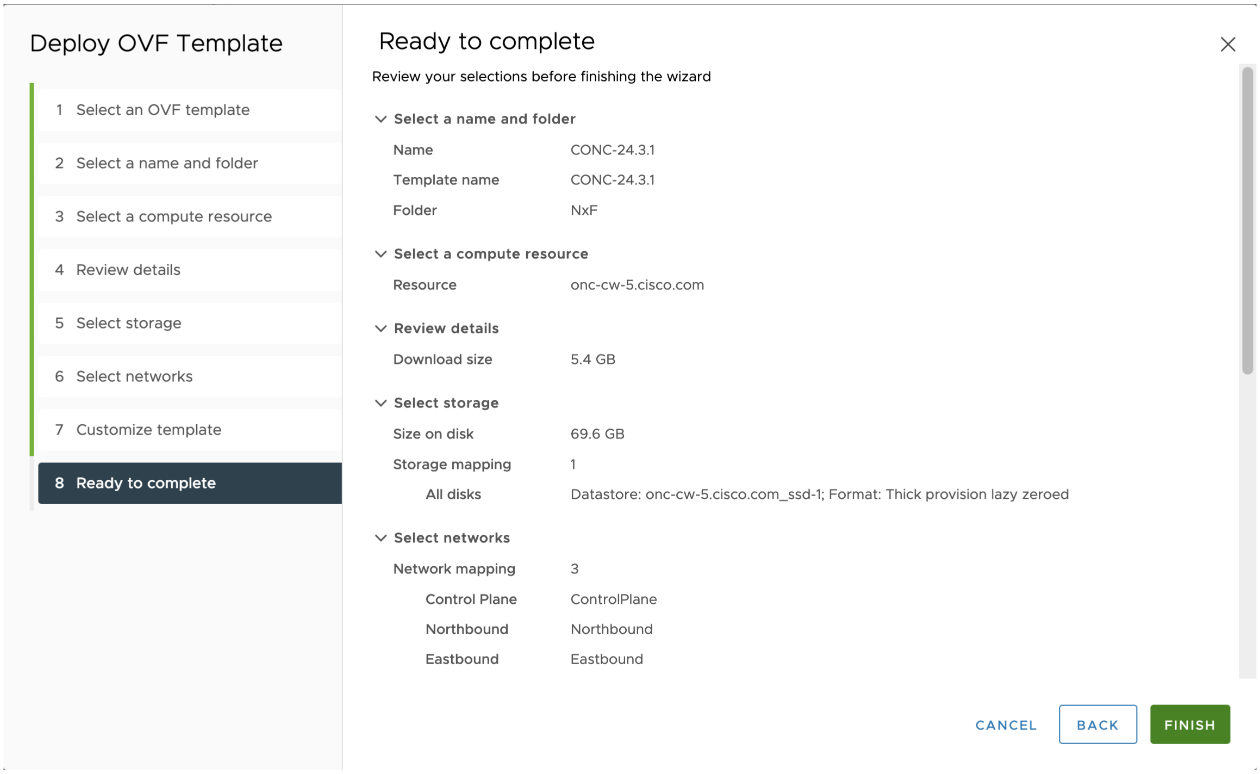
Table 3: Customize Template

Key	Values
Instance Hostname	*<instance-name>*
SSH Public Key	<*ssh-public-key*>. Used for SSH access that allows you to connect to the instances securely without the need to manage credentials for multiple instances. SSH public key must be a ed25519 key.
Node Name	node1 Must be a valid DNS name per RFC1123.1.2.4 • Contain at most 63 characters. • Contain only lowercase alphanumeric characters or '-'.3 • Start with an alphanumeric character. • End with an alphanumeric character. • Node Name should be the same as instance name.
Initiator Node	Select the Checkbox

Supercluster Cluster Index	1 If you want to add your Cisco Optical Network Controller instance to a GeoHA SuperCluster in the future, use different Super Cluster Index values for each instance.
Supercluster Cluster Name	cluster1 Must be a valid DNS name per RFC1123 If you want to add your Cisco Optical Network Controller instance to a GeoHA SuperCluster in the future, use unique Super Cluster Names for each instance.
Data Volume Size (GB)	200GB
NTP Pools (comma separated)	debian.pool.ntp.org
NTP Servers (comma separated)	1.ntp.esl.cisco.com
Cluster Join Token	Can be left with the default value
Control Plane Node Count	1
Control Plane IP	<Private IP for the Instance> Control Plane Network
Initiator IP	<Same IP as Control Plane> Control Plane Network
Protocol	Static IP
IP (ip[/subnet]) - if not using DHCP	<Public IP for the Instance> Northbound Network
Gateway - if not using DHCP	<Gateway IP for the Instance> Northbound Network
DNS	DNS Server IP
Protocol	Static IP
IP (ip[/subnet]) - if not using DHCP	< IP for the Instance> Eastbound Network Can be a private IP
Gateway - if not using DHCP	<Gateway IP for the Network> Eastbound Network
DNS	DNS Server IP
Northbound Virtual IP Type	L2
Northbound Virtual IP	Same as Northbound IP
Supercluster Cluster Role	worker
Arbitrator Node Name	node3

Step 9 In **Review the details** screen, review all your selections and click **Finish**. To check or change any properties from the review screen anytime, before clicking Finish **click BACK** to go back to the previous screen **Customize template** to ad your changes.

Figure 8: Ready to Complete



Ready to complete

Review your selections before finishing the wizard

Select a name and folder

Name	CONC-24.3.1
Template name	CONC-24.3.1
Folder	NxF

Select a compute resource

Resource	onc-cw-5.cisco.com

Review details

Download size	5.4 GB

Select storage

Size on disk	69.6 GB
Storage mapping	1
All disks	Datastore: onc-cw-5.cisco.com_ssd-1; Format: Thick provision lazy zeroed

Select networks

Network mapping	3
Control Plane	ControlPlane
Northbound	Northbound
Eastbound	Eastbound

Step 10 After the VM is created, try connecting to the VM using the pem key which was generated earlier, see SSH Key Generation above. For this, use the private key that is generated along with the public key during customizing the public key options.

Step 11 Log in to the VM using the private key.

Note:

- After the nodes are deployed, the deployment of OVA progress can be checked in the Tasks console of vSphere Client. After Successful deployment Cisco Optical Network Controller takes around 30 minutes to boot.
- By default, the user ID is admin, and only the password needs to be set.

Step 12 **SSH to the node** and execute the following CLI command.

```
##Command to change permissions of key file
chmod 400 <file-name-of-your-key>.pem

ssh -i [ed25519 Private key] nxf@<northbound-ip>/<dns name assigned too the IP>
Enter passphrase for key '<file-name-of-your-key>.pem':
```

Note Private key is created as part of the key generation with just the **.pem** extension, and it must be set with the least permission level before using it.

Step 13 **SSH to the node** and execute the following CLI command.

```
ssh -i [ed25519 Private key] nxf@<northbound-vip>
Enter passphrase for key '<file-name-of-your-key>.pem':
```

Note Private key is created as part of the key generation with just the **.pem** extension, and it must be set with the least permission level before using it.

Step 14 After you SSH into the node, use the sedo system status command to check the status of all the pods.

```
sedo system status
```

```
┌─────────────────────────────────────────────────────────────────────────────────────┐
│ System Status (Fri, 20 Sep 2024 08:21:27 UTC)                                       │
├────────┬───────────────────────────────┬───────┬─────────┬──────────┬──────────────┤
│ OWNER  │ NAME                          │ NODE  │ STATUS  │ RESTARTS │ STARTED      │
├────────┼───────────────────────────────┼───────┼─────────┼──────────┼──────────────┤
│ onc    │ monitoring                    │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-alarm-service             │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-apps-ui-service           │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-circuit-service           │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-collector-service         │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-config-service            │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-devicemanager-service     │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-inventory-service         │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-nbi-service               │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-netconfcollector-service  │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-osapi-gw-service          │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-pce-service               │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-pm-service                │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-pmcollector-service       │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-topology-service          │ node1 │ Running │ 0        │ 3 hours ago  │
│ onc    │ onc-torch-service             │ node1 │ Running │ 0        │ 3 hours ago  │
│ system │ authenticator                 │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ controller                    │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ flannel                       │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ ingress-proxy                 │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ kafka                         │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ loki                          │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ metrics                       │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ minio                         │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ postgres                      │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ promtail-cltmk                │ node1 │ Running │ 0        │ 12 hours ago │
│ system │ vip-add                       │ node1 │ Running │ 0        │ 12 hours ago │
└────────┴───────────────────────────────┴───────┴─────────┴──────────┴──────────────┘
```

Note

- The different pods along with their statuses including active and standby modes are all displayed in the different terminal sessions for each pod.
- All the services with owner *onc* must display the status as *Running.*

Step 15 You can check the current version using the **sedo version** command.

```
sedo version
```

```
┌──────────────────────────────────────────────────────────────────────────────────────┐
│ Installer: CONC 24.3.1                                                               │
├───────────┬──────────────────────────────────────────────────────────┬──────────────┤
│ NODE NAME │ OS VERSION                                               │ KERNEL VERSION │
├───────────┼──────────────────────────────────────────────────────────┼──────────────┤
│ node1     │ NxFOS 3.0-408 (f2beddad9abeb84896cc13efcd9a87c48ccb5d0c) │ 6.1.0-23-amd64 │
└───────────┴──────────────────────────────────────────────────────────┴──────────────┘

┌──────────────────────────────────────────────────────────────────────────┬───────────────────┬─────────┐
│ IMAGE NAME                                                               │ VERSION
│ NODES │
├──────────────────────────────────────────────────────────────────────────┼───────────────────┼─────────┤
│ docker.io/library/alpine                                                 │ 3.20.0
│ node1 │
│ docker.io/rancher/local-path-provisioner                                 │ v0.0.27
│ node1 │
│ dockerhub.cisco.com/cisco-onc-docker/dev/ciscotestautomation/pyats         │ 23.7.1-beta2
│ node1 │
│ quay.io/coreos/etcd                                                        │ v3.5.12
│ node1 │
│ registry.k8s.io/coredns/coredns                                            │ v1.11.1
│ node1 │
│ registry.k8s.io/kube-apiserver                                             │ v1.30.2
│ node1 │
│ registry.k8s.io/kube-controller-manager                                    │ v1.30.2
│ node1 │
│ registry.k8s.io/kube-proxy                                                 │ v1.30.2
│ node1 │
│ registry.k8s.io/kube-scheduler                                             │ v1.30.2
│ node1 │
│ registry.k8s.io/pause                                                      │ 3.9
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/alarmservice             │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/circuit-service          │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/collector-service        │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/config-service           │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/devicemanager-service    │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/inventory-service        │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/monitoring               │ release2431_latest
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/nbi-service              │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/netconfcollector-service │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/onc-apps-ui-service      │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/osapi-gw-service         │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/pce_service              │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/pm-service               │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/pmcollector-service      │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/topology-service         │ 24.3.1-3
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/torch                    │ 24.3.1-3
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/authenticator                            │ 3.0-348
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/bgp                                      │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/controller                               │ 3.0-384
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/firewalld                                │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/flannel                                  │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/ingress-proxy                            │ 3.0-370
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/iptables                                 │ 3.0-370
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/kafka                                    │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/loki                                     │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/metrics-exporter                         │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/minio                                    │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/service-proxy                            │ 3.0-370
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/syslog-forwarder                         │ 3.0-340
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/timescale                                │ 3.0-359
│ node1 │
└─────────────────────────────────────────────────────────────────────┴───────────────────┴──────┘
```

Step 16 SSH to the node and set the initial UI password for the admin user.

```
sedo security user set admin --password
```

Step 17 To check the default admin user ID, use the command sedo security user list. To change the default password, use the command sedo security user admin set --password on the CLI console of the VM or through the web UI.

Step 18 Use a web browser to access *https://<virtual ip>:8443/* to access the Cisco Optical Network Controller Web UI. Use the admin id and the password you set to log in to Cisco Optical Network Controller.

Note Access the web UI only after all the `onc` services are running. Use the **sedo system status** to verify that all services are running.

CHAPTER 2

Install Cisco Optical Network Controller Using OpenStack

To deploy the Cisco Optical Network Controller using OpenStack, follow the instructions in this task. The deployment leverages a Heat Orchestration Template to automate the creation of necessary components and configurations.

Heat Orchestration Template

A Heat orchestration template will be provided to create the required components for the instance. The template includes configurations for block storage, security groups, and network settings.

Components Created by Heat Template

- **Block Storage**: Image and data volumes are created and attached to the instance.
- **Security Groups**: Security groups for network ports are established.
- **Network Configuration**: A control plane network and subnet are created as a private network, and a northbound port will be created.
- **Join Token**: Random text is generated to be used as a join token.
- **Cloud-Init Configuration**: The cloud-init is prefilled based on the parameters that are obtained during stack launch.

Before you begin

- **OpenStack Version**: 2024.1

  See OpenStack Documentation for release 2024.1 for details on how to use OpenStack.

- **Upload Image**: Upload the Cisco Optical Network Controller (qcow2) image to the server..

  Use the following CLI command to upload the image to the OpenStack project.

  ```
  openstack image create --disk-format=qcow2 --file <path-to-image>.qcow2 \
     --shared \
     --property hw_firmware_type='uefi' \
     --property hw_machine_type='q35' \
     --property architecture='x86_64' \
     --progress \
     "Image Name"
  ```

  After you perform these commands, the qcow2 image is available for deployment in OpenStack.

Note Install the OpenStack Command Line Interface (CLI) and source the OpenStack Cloud RC file or clouds.yaml before running the command. For installation instructions, see Install the OpenStack command-line clients.

- **Configure Network**: Use the northbound network for Cisco Optical Network Controller to expose the UI and REST APIs. Cisco Optical Network Controller uses this northbound network to connect to the devices.
- **Create Flavors:** It is optional to have physical disks/ephemeral storage. While creating a flavor both physical disks/ephemeral storage cane be set to 0GB as block storage volumes handle both the image and data volumes.

  To create a flavor, in the OpenStack Dashboard, select the admin project from the drop-down list, select **Admin** > **Compute** > **Flavors** > **Create Flavor** and enter the parameters for the flavor.

  Note You need administrative access to OpenStack to create flavors.

  The minimum requirement for Cisco Optical Network Controller 24.3.1 installation are in the following table.

  Table 4: Minimum Requirement

Sizing	CPU	Memory	Disk
XS	16 vCPU	64 GB	800 GB
S	32 vCPU	128 GB	1.5 TB

- **Create Key Pair**: Create a key pair using the ed25519 algorithm. Upload Public SSH Key to OpenStack by going to **Project** > **Compute** > **Key Pairs** and select Import Public Key.

  Run the following command in a UNIX-based environment to create an SSH key pair:

```
ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/Users/xyz/.ssh/id_ed25519):
./<file-name-of-your-key>.pem
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in ./<file-name-of-your-key>.pem
Your public key has been saved in ./<file-name-of-your-key>.pem.pub
The key fingerprint is:
SHA256:zGW6aGn8rxvEq82sA/97jOaHrl9rnoTaYi+TqU3MeRU xyz@abc
The key's randomart image is:
+--[ED25519 256]--+
|                 |
|                 |
|          E      |
|       + + .     |
|        S .      |
|    .+ = =       |
|     o@o*+o      |
|     =XX++=o     |
|    .o*#/X=      |
+----[SHA256]-----+

#Once created you can cat the file with .pub extension for the public key. ( ex:
<file-name-of-your-key>.pem.pub )

cat <file-name-of-your-key>.pem.pub
#The above key has to be used in the deployment template ( SSH Public Key ) in the
Deployment process
```

Follow the prompts to save the key. The key pair will be used to access Cisco Optical Network Controller after the installation.

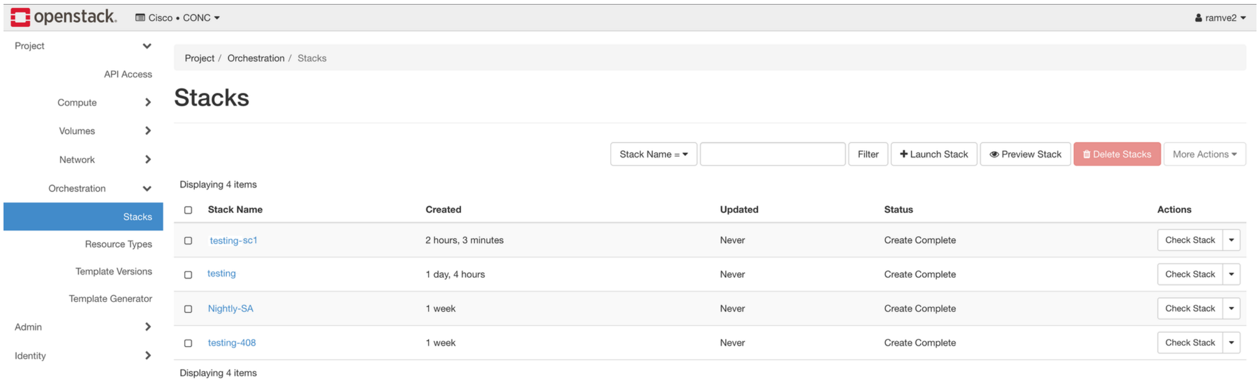
- You must have an NTP server or NTP Pool for time synchronization.
- You must have a DNS server. The DNS server can be an internal DNS server if the Cisco Optical Network Controller instance is not exposed to the internet.

Perform the following steps to install Cisco Optical Network Controller using OpenStack.

Step 1 Log in to OpenStack.

Step 2 Select **Project** > **Orchestration** > **Stacks** from the sidebar.

Figure 9: OpenStack Stacks Screen



Stacks

Displaying 4 items

□	Stack Name	Created	Updated	Status	Actions
□	testing-sc1	2 hours, 3 minutes	Never	Create Complete	Check Stack
□	testing	1 day, 4 hours	Never	Create Complete	Check Stack
□	Nightly-SA	1 week	Never	Create Complete	Check Stack
□	testing-408	1 week	Never	Create Complete	Check Stack

Displaying 4 items

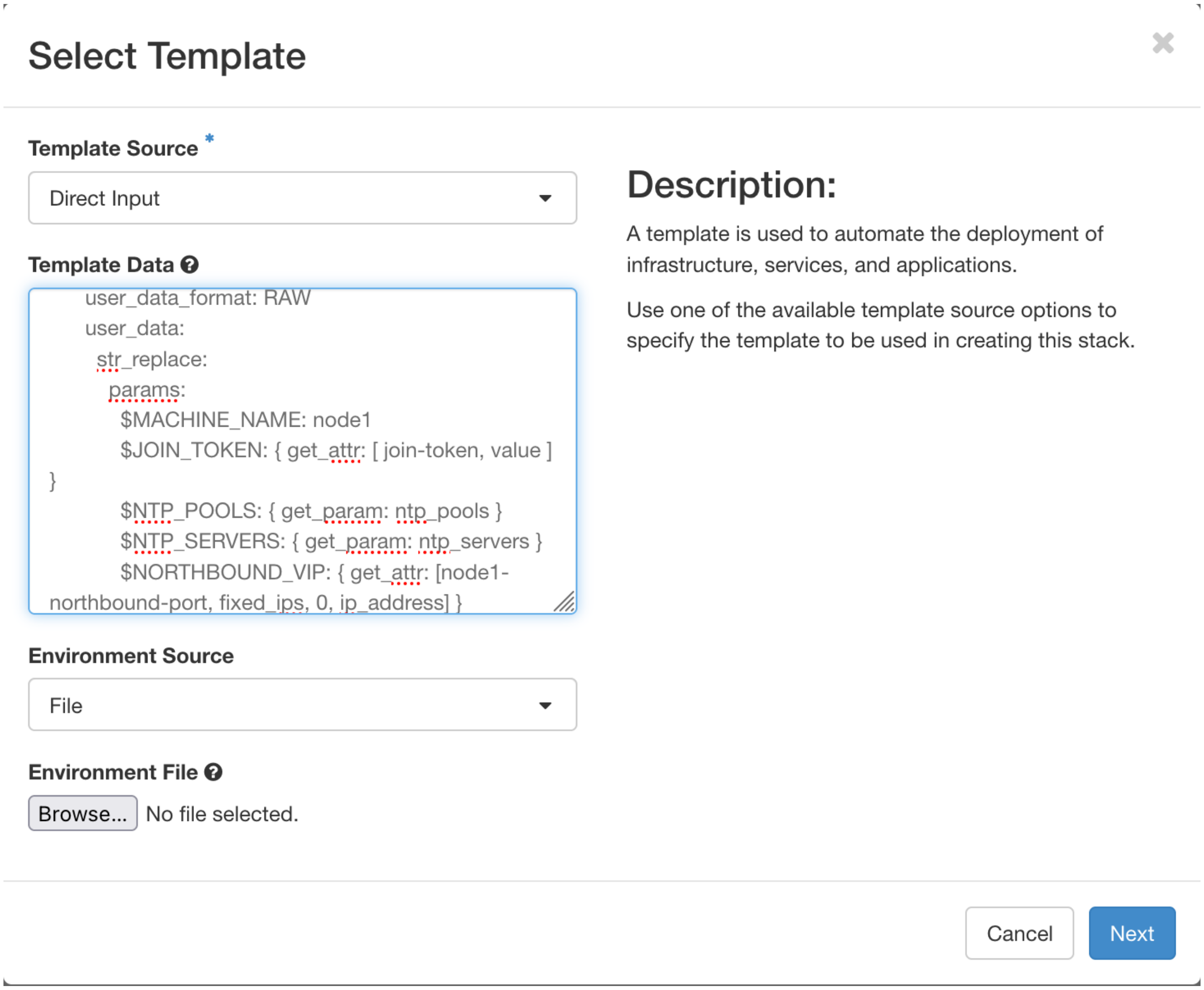
Step 3 Launch Stack.

a) Click **Launch Stack**

b) Choose **Template as File** and Upload the Heat orchestration template file or choose **Direct Input** and paste the contents of the file.

Note Incorrect indentation causes parsing errors. Validate the file with a YAML validator.

Figure 10: Select Template



```
      user_data_format: RAW
      user_data:
        str_replace:
          params:
            $MACHINE_NAME: node1
            $JOIN_TOKEN: { get_attr: [ join-token, value ]
}
            $NTP_POOLS: { get_param: ntp_pools }
            $NTP_SERVERS: { get_param: ntp_servers }
            $NORTHBOUND_VIP: { get_attr: [node1-
northbound-port, fixed_ips, 0, ip_address] }
```

Description:

A template is used to automate the deployment of infrastructure, services, and applications.

Use one of the available template source options to specify the template to be used in creating this stack.

The following sample is a Heat Orchestration Template file for Cisco Optical Network Controller.

```
heat_template_version: "2021-04-16"
description: "NxFOS Heat Template"
parameters:
  instance_flavor:
    type: string
    label: Instance Flavor
    constraints:
    - custom_constraint: nova.flavor
  image_name:
    type: string
    label: CONC Image Name
    constraints:
    - custom_constraint: glance.image
  northbound_network:
    type: string
    label: Northbound Network
    constraints:
    - custom_constraint: neutron.network
  northbound_subnet:
    type: string
    label: Northbound Subnet
  northbound_vip:
    type: string
    label: Northbound VIP address
    default: "10.1.1.1"
  control_key_pair:
    type: string
    label: Control plane SSH key-pair
    constraints:
    - custom_constraint: nova.keypair
  data_volume_size_gb:
    type: number
    label: Data volume size in GB
    default: 200
  ntp_pools:
    type: comma_delimited_list
    description: List of NTP pools
    default: "0.pool.ntp.org,1.pool.ntp.org"
  ntp_servers:
    type: comma_delimited_list
    description: List of NTP servers
    default: ""

resources:
  # Security Groups
  control-sec-group:
    type: OS::Neutron::SecurityGroup
    properties:
      rules:
      # K8s
      - { protocol: tcp, remote_ip_prefix: 10.1.0.0/24, port_range_min: 443, port_range_max:
443 }
      - { protocol: tcp, remote_ip_prefix: 10.1.0.0/24, port_range_min: 6443, port_range_max:
6443 }
      - { protocol: tcp, remote_ip_prefix: 10.1.0.0/24, port_range_min: 10250, port_range_max:
 10250 }

      # Etcd (Port 2379 + 2380)
      - { protocol: tcp, remote_ip_prefix: 10.1.0.0/24, port_range_min: 2379, port_range_max:
2380 }

      # Flannel CNI
      - { protocol: udp, remote_ip_prefix: 10.1.0.0/24, port_range_min: 8472, port_range_max:
8472 }

      # Ping between nodes
      - { protocol: icmp, remote_ip_prefix: 10.1.0.0/24 }

  northbound-sec-group:
    type: OS::Neutron::SecurityGroup
    properties:
      rules:
      # SSH (Debug purposes only)
      - { protocol: tcp, remote_ip_prefix: 0.0.0.0/0, port_range_min: 22, port_range_max: 22 }

      # Northbound ingress-proxy
     - { protocol: tcp, remote_ip_prefix: 0.0.0.0/0, port_range_min: 8443, port_range_max: 8443
 }

  # Networks
  control-plane-network:
    type: OS::Neutron::Net
    properties:
      admin_state_up: true
  control-plane-subnet:
    type: OS::Neutron::Subnet
    properties:
      network_id: { get_resource: control-plane-network }
      gateway_ip: null
      cidr: "10.1.0.0/24"
      ip_version: 4

  # Control Ports
  node1-control-port:
    type: OS::Neutron::Port
    properties:
      security_groups: [ { get_resource: control-sec-group } ]
      network: { get_resource: control-plane-network }
      fixed_ips:
      - subnet_id: { get_resource: control-plane-subnet }
        ip_address: "10.1.0.10"

  # Northbound Ports
  node1-northbound-port:
    type: OS::Neutron::Port
    properties:
      security_groups: [ { get_resource: northbound-sec-group } ]
      network: { get_param: northbound_network }
      fixed_ips:
      - subnet_id: { get_param: northbound_subnet }
        ip_address: { get_param: northbound_vip }

  # Join Token
  join-token-id:
    type: OS::Heat::RandomString
    properties:
      character_classes:
      - class: lowercase
      - class: digits
      length: 6

  join-token-secret:
    type: OS::Heat::RandomString
    properties:
      character_classes:
      - class: lowercase
      - class: digits
      length: 16

  join-token:
    type: OS::Heat::Value
    properties:
      type: string
      value:
        list_join: [ '.', [ { get_resource: join-token-id }, { get_resource: join-token-secret
 } ] ]

  # Data Volumes
  node1-data-volume:
    type: OS::Cinder::Volume
    properties:
      size: { get_param: data_volume_size_gb }

  # Instances
  node1:
    type: OS::Nova::Server
    properties:
```

```
   networks:
   - port: { get_resource: node1-control-port }
   - port: { get_resource: node1-northbound-port }
   flavor: { get_param: instance_flavor }
   key_name: { get_param: control_key_pair }
   block_device_mapping_v2:
   - device_name: vda
    image: { get_param: image_name }
    volume_size: 50
    delete_on_termination: true
   - device_name: vdb
    volume_id: { get_resource: node1-data-volume }
    boot_index: -1
    delete_on_termination: true
   user_data_format: RAW
   user_data:
    str_replace:
     params:
      $MACHINE_NAME: node1
      $JOIN_TOKEN: { get_attr: [ join-token, value ] }
      $NTP_POOLS: { get_param: ntp_pools }
      $NTP_SERVERS: { get_param: ntp_servers }
      $NORTHBOUND_VIP: { get_attr: [node1-northbound-port, fixed_ips, 0, ip_address] }
      $POSTGRES_CONFIG: '{"config": {"max_connections": "1000","idle_session_timeout":
"900000"},"resources": {"requests": {"memory": "3.22%","cpu": "3.33%"},"limits": {"memory":
"9.66%","cpu": "11%"}}}'
      $KAFKA_CONFIG:
'{"enabled":true,"resources":{"requests":{"memory":"7.52%","cpu":"3.33%"},"limits":{"memory":"10.74%","cpu":"5.4%"}},"config":{"message.max.bytes":15000012}}'

     template: |
      #cloud-config
      fs_setup:
      - label: data
       device: /dev/vdb
       filesystem: ext4

      mounts:
      - [ "/dev/vdb", "/data" ]

      ntp:
       enabled: true
       ntp_client: chrony
       pools: $NTP_POOLS
       servers: $NTP_SERVERS

      nxf:
       minControlPlaneCount: 1
       node:
        name: $MACHINE_NAME
        controlPlaneInterface: enp3s0
        vip:
         northbound:
          interface: enp4s0

       initiator:
        vip:
         northbound:
          ip: $NORTHBOUND_VIP
        postgres: $POSTGRES_CONFIG
        kafka: $KAFKA_CONFIG
        minio:
         resources:
          limits:
           memory: "5.37%"
```

```
joinToken: $JOIN_TOKEN
security:
  localUsers:
  - username: admin
    displayName: NxF Admin
    description: NextFusion Default Administrator
    locked: true
    mustChangePassword: false
    expiresInDays: 0
    access:
    - permission/admin
```

Step 4 In the Launch Stack dialog box, enter the Stack Parameters.

Table 5: Stack Parameters

Key	Value
Stack Name	Name of the stack, which will be used as part of the Node name.
Creation Timeout (minutes)	Can be left to default. Value can be changed to support the respective environment.
Password for the user	Enter the password of the OpenStack account used to log in.
Control Plane SSH Key Pair	Select the key pair (Should be an ed25519 SSH key).
Data Volume Size in GB	Enter the size of the data volume size based on the Cisco Optical Network Controller profiles.
CONC Image Name	Select the Cisco Optical Network Controller Image (qcow2).
Instance Flavor	Select the respective Cisco Optical Network Controller flavor based on the profiles.
Northbound Network	Select the Northbound Network.
Northbound Subnet	Enter the name in the text field of the Northbound Subnet.
Northbound VIP Address	Public IP, which will be used for both management and Northbound communications.
NTP Pools	Enter the NTP Pools. Leave empty if you are using an NTP Server.
NTP Server	Enter the NTP Server. Leave empty if you are using an NTP Pool.

Step 5 Click **Launch**.
This creates the stack. Use the PEM key to SSH into the node.

Note Wait for the stack creation status to change to **Create Complete** before you try to SSH into the node. Stack creation can take up to 10 minutes.

Step 6 **SSH to the node** and execute the following CLI command.

```
ssh -i [ed25519 Private key] nxf@<northbound-vip>
Enter passphrase for key '<file-name-of-your-key>.pem':
```

Note Private key is created as part of the key generation with just the **.pem** extension, and it must be set with the least permission level before using it.

Step 7 After you SSH into the node, use the sedo system status command to check the status of all the pods.

```
sedo system status
```

```
┌──────────────────────────────────────────────────────────────────────────────────┐
│ System Status (Fri, 20 Sep 2024 08:21:27 UTC)                                    │
├────────┬──────────────────────────────┬───────┬─────────┬──────────┬─────────────┤
│ OWNER  │ NAME                         │ NODE  │ STATUS  │ RESTARTS │ STARTED     │
├────────┼──────────────────────────────┼───────┼─────────┼──────────┼─────────────┤
│ onc    │ monitoring                   │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-alarm-service            │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-apps-ui-service          │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-circuit-service          │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-collector-service        │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-config-service           │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-devicemanager-service    │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-inventory-service        │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-nbi-service              │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-netconfcollector-service │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-osapi-gw-service         │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-pce-service              │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-pm-service               │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-pmcollector-service      │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-topology-service         │ node1 │ Running │ 0        │ 3 hours ago │
│ onc    │ onc-torch-service            │ node1 │ Running │ 0        │ 3 hours ago │
│ system │ authenticator                │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ controller                   │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ flannel                      │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ ingress-proxy                │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ kafka                        │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ loki                         │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ metrics                      │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ minio                        │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ postgres                     │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ promtail-cltmk               │ node1 │ Running │ 0        │ 12 hours ago│
│ system │ vip-add                      │ node1 │ Running │ 0        │ 12 hours ago│
└────────┴──────────────────────────────┴───────┴─────────┴──────────┴─────────────┘
```

Note
- All the services with owner *onc* must display the status as *Running.* After stack creation, it can take up to 20 minutes for all services to reach the *Running* state.

Step 8 SSH to the node and set the initial UI password for the admin user.

```
sedo security user set admin --password
```

Step 9 You can check the current version using the **sedo version** command.

```
sedo version
```

```
┌──────────────────────────────────────────────────────────────────────────────────────┐
│ Installer: CONC 24.3.1                                                               │
├───────────┬──────────────────────────────────────────────────────────┬───────────────┤
│ NODE NAME │ OS VERSION                                               │ KERNEL VERSION│
├───────────┼──────────────────────────────────────────────────────────┼───────────────┤
│ node1     │ NxFOS 3.0-408 (f2beddad9abeb84896cc13efcd9a87c48ccb5d0c) │ 6.1.0-23-amd64│
└───────────┴──────────────────────────────────────────────────────────┴───────────────┘

┌─────────────────────────────────────────────────────────────────────┬────────────┬───┐
│ IMAGE NAME                                                          │ VERSION    
│ NODES │
├──────────────────────────────────────────────────────────────────────┼──────────────────┼───────┤
│ docker.io/library/alpine                                                   │ 3.20.0
│ node1 │
│ docker.io/rancher/local-path-provisioner                                   │ v0.0.27
│ node1 │
│ quay.io/coreos/etcd                                                        │ v3.5.12
│ node1 │
│ registry.k8s.io/coredns/coredns                                            │ v1.11.1
│ node1 │
│ registry.k8s.io/kube-apiserver                                             │ v1.30.2
│ node1 │
│ registry.k8s.io/kube-controller-manager                                    │ v1.30.2
│ node1 │
│ registry.k8s.io/kube-proxy                                                 │ v1.30.2
│ node1 │
│ registry.k8s.io/kube-scheduler                                             │ v1.30.2
│ node1 │
│ registry.k8s.io/pause                                                      │ 3.9
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/alarmservice             │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/circuit-service          │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/collector-service        │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/config-service           │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/devicemanager-service    │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/inventory-service        │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/monitoring               │ release2431_latest
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/nbi-service              │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/netconfcollector-service │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/onc-apps-ui-service      │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/osapi-gw-service         │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/pce_service              │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/pm-service               │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/pmcollector-service      │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/topology-service         │ 24.3.1-5
│ node1 │
│ registry.nxf-system.svc:8443/cisco-onc-docker/dev/torch                    │ 24.3.1-5
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/authenticator                            │ 3.0-348
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/bgp                                      │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/controller                               │ 3.0-384
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/firewalld                                │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/flannel                                  │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/ingress-proxy                            │ 3.0-370
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/iptables                                 │ 3.0-370
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/kafka                                    │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/loki                                     │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/metrics-exporter                         │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/minio                                    │ 3.0-365
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/service-proxy                            │ 3.0-370
│ node1 │
│ registry.sedona.ciscolabs.com/nxf/syslog-forwarder                         │ 3.0-340
│ node1 │ registry.sedona.ciscolabs.com/nxf/timescale                                │ 3.0-359
        │ node1 │
└─────────────────────────────────────────────────────────────────────┴───────────────────┴──────┘
```

Step 10 To check the default admin user ID, use the command **sedo security user list**.

Step 11 Use a web browser to access *https://<virtual ip>:8443/* to access the Cisco Optical Network Controller Web UI. Use the admin id and the password that you set to log in to Cisco Optical Network Controller.

Note Access the web UI only after all the `onc` services are running. Use the **sedo system status** command to verify that all services are running.