VM Life Cycle Management

VM life cycle management

VM life cycle management is a process that

- refers to the entire process of registering, deploying, updating, monitoring VMs
- gets VMs service chained as per your requirements, and
- can be performed using a set of REST APIs or NETCONF commands or the Cisco NFVIS portal.

Uploading VM images to an NFVIS server

Uploading VM images to an NFVIS server is a file transfer process that copies VM images to the default location (/data/intdatastore/uploads) on the host server.

VM image upload methods

You can upload VM images to an NFVIS server in these ways:

- Copy the images from your local system to the NFVIS server—Use the **Image Upload** option from the Cisco NFVIS portal.
- Copy using the **scp** command (scp username@external_server:/path/image.tar.gz intdatastore:image.tar.gz).

Perform resource verification

To display information on all CPUs, VMs pinned to the CPUs, and VMs allocated to the CPUs, use the **show resources CPU-info** command.

```
nfvis# show resources cpu-info
resources cpu-info allocation total-sockets 2
resources cpu-info allocation cores-per-socket 20
```

```
resources cpu-info allocation total-logical-cpus 80
resources cpu-info allocation logical-cpus-used-by-system 4
resources cpu-info allocation logical-cpus-used-by-vnfs 4
resources cpu-info allocation logical-cpus-used-dedicated 0
resources cpu-info allocation logical-cpus-used-sharable 4
CPU  SOCKET  CORE  SYSTEM                LOW      VCPU
ID   ID      ID    USE     NAME     VCPUS  LATENCY  ID
-----------------------------------------------------------
0    0       0     true
20   1       20    true
40   0       0     true
60   1       20    true
1    0       1     false
41   0       1     false
2    0       2     false
42   0       2     false
3    0       3     false
43   0       3     false
4    0       4     false
44   0       4     false
5    0       5     false
45   0       5     false
6    0       6     false
46   0       6     false
7    0       7     false
47   0       7     false
8    0       8     false
48   0       8     false
9    0       9     false
49   0       9     false
10   0       10    false
50   0       10    false
11   0       11    false
51   0       11    false
12   0       12    false
52   0       12    false
13   0       13    false
53   0       13    false
14   0       14    false
54   0       14    false
15   0       15    false
55   0       15    false
16   0       16    false   OTHER76  1      false    0
56   0       16    false
17   0       17    false   OTHER99  1      false    0
57   0       17    false
18   0       18    false   OTHER57  1      false    0
58   0       18    false
19   0       19    false   OTHER95  1      false    0

nfvis# show resources cpu-info allocation
resources cpu-info allocation total-sockets 2
resources cpu-info allocation cores-per-socket 20
resources cpu-info allocation total-logical-cpus 80
resources cpu-info allocation logical-cpus-used-by-system 4
resources cpu-info allocation logical-cpus-used-by-vnfs 4
resources cpu-info allocation logical-cpus-used-dedicated 0
resources cpu-info allocation logical-cpus-used-sharable 4

nfvis# show resources cpu-info vnfs
                LOW      VCPU  SOCKET  CORE  CPU
NAME     VCPUS  LATENCY  ID    ID      ID    ID
-------------------------------------------------
OTHER95  1      false    0     0       19    19
OTHER57  1      false    0     0       18    18
```

```
OTHER99  1      false    0     0       17    17
OTHER76  1      false    0     0       16    16
```

CPU Over-Subscription

Cisco NFVIS does not allow CPU over-subscription for low-latency network appliance VMs (for example, Cisco ISRv and Cisco ASAv). However, the CPU over-subscription is allowed for non low-latency VMs (for example, Linux Server VM and Windows Server VM).

Configure management IP subnet

Change the default subnet for all VMs with management interfaces from the default 10.20.0.1 subnet.

By default, all VMs with management interfaces will be provisioned with an IP address in the subnet of 10.20.0.1. You can change the default subnet by executing commands in a sequence to first delete the existing subnet and then add a new subnet in the network.

Before you begin

Ensure there are no managed VNFs in the system before you change the management network address.

Follow these steps to configure the management IP subnet:

Procedure

Step 1 To delete the existing subnet, use the **no vm_lifecycle networks network int-mgmt-net subnet int-mgmt-net-subnet** command.

Step 2 To create a new subnet, enter the following commands:

Example:

```
configure terminal
vm_lifecycle networks network int-mgmt-net subnet int-mgmt-net-subnet address 105.20.0.0 gateway
105.20.0.1 netmask 255.255.255.0 dhcp false
```

The management IP subnet is configured with the new subnet settings.

Workflow of vm life cycle management

VM life cycle management using REST APIs provides a structured workflow for creating, customizing, deploying, and managing virtual machines in the NFVIS environment.

Note Before performing the VM life cycle management tasks, you will have to upload the VM images to the NFVIS server or http/s, sftp, scp server.

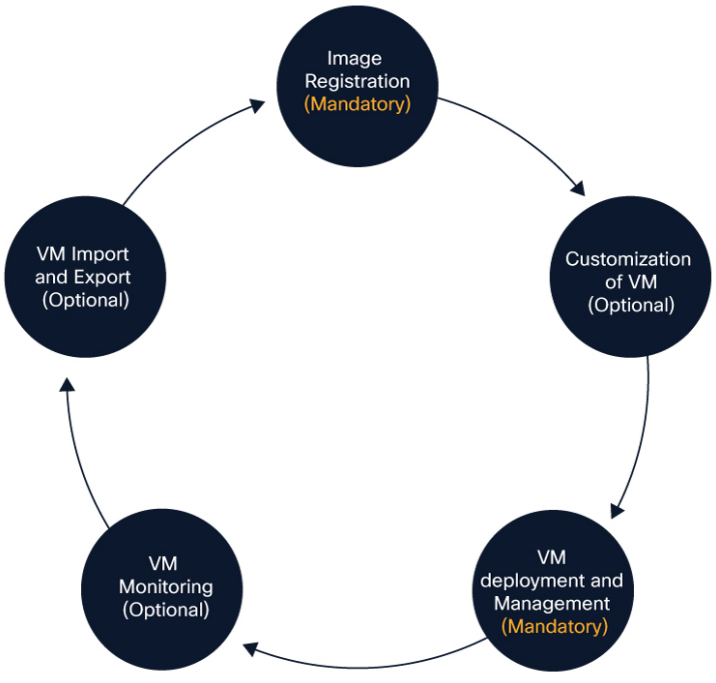
Summary

The key components involved in the VM life cycle management process are:

- VM Image: The source image that serves as the template for creating VM instances
- NFVIS Image Repository: The storage location where registered VM images are maintained
- Remote Servers: HTTP/HTTPS, FTP, or SCP servers that host the source images
- Custom Profiles/Flavors: Specific configurations for VM resources such as CPU and memory
- Deployment APIs: Interfaces that enable VM deployment and parameter configuration
- Management APIs: Tools for monitoring, controlling, and updating deployed VMs

Workflow

Figure 1: VM life cycle management



These stages describe how VM life cycle management works:

1. **Register a VM Image—**To create a VM instance, you must register the source image in the NFVIS image repository. Registering the image is a one-time activity. The source image is expected to be hosted on a remote server. In the Registration API you specify the URL on the remote server, where the file is located. When invoked, device pulls the API info, completes the file transfer and registration. Once registered, you can create one or more VM instances. Remote servers supported for hosting the source images (.tar.gz files) are HTTP/HTTPS servers. Starting with Cisco NFVIS release 4.12.1, remote servers can also be FTP servers or SCP servers.
2. **Customizing the VM—**After registering a VM image, you can optionally create a custom profile or flavor for the VM image if the profiles defined in the image file do not match your requirement. The flavor creation option lets you provide specific profiling details for a VM image, such as the virtual CPU on which the VM will run, and the amount of virtual memory the VM will consume.

Depending on the topology requirement, you can create additional networks and bridges to attach the VM to during deployment.

3. **Deploy a VM—** A VM can be deployed using the deployment API. The deployment API allows you to provide values to the parameters that are passed to the system during deployment. Depending on the VM you are deploying, some parameters are mandatory and others optional.
4. **Manage and Monitor a VM—**You can monitor a VM using APIs and commands that enable you to get the VM status and debug logs. Using VM management APIs, you can start, stop, or reboot a VM, and view statistics for a VM such as CPU usage.

   A VM can also be managed by changing or updating its profile. You can change a VM's profile to one of the existing profiles in the image file; alternatively, you can create a new custom profile for the VM.

   The vNICs on a deployed VM can also be added or updated.

Image Registration

Image registration is a process that

- copies or downloads VM images to the NFVIS server or hosts them on http, https, FTP, or SCP servers
- uses the registration API to specify the file path to the location where the tar.gz file is hosted, and
- enables multiple VM deployments using the registered image once it is in active state.

Image registration characteristics

Image registration is a one-time activity. Once an image is registered on the http or https server and is in active state, you can perform multiple VM deployments using the registered image. All VM images are available in VM packaging and VM package content.

Register images for CUCM applications

Register both the ISO image (installation media) and the OVA file (metadata) with NFVIS for collaboration applications such as Cisco Unified Communications Manager (CUCM), Cisco Unity Connection, and Cisco IM and Presence Service (IM&P).

Collaboration applications require both ISO images and OVA metadata to be properly registered with NFVIS before deployment. The registration process automatically creates profiles and flavors from the OVA metadata.

Procedure

Step 1 Navigate to the Image Repository.

Access the image repository via **Configuration** > **Virtual Machine** > **Images** > **Image Repository**

Step 2 Upload the ISO image.

a. Click **Upload**.

b. Select the ISO file from your local file system.

Example: Bootable_UCSInstall_UCOS_15.0.1.ISO

Step 3 Configure image properties.

a. Update the pre-populated image name to a descriptive name.

Example: CUCM-15.0.1

b. Select the appropriate **VM Type** from the drop-down list. If your application type is not listed, select **Other**.

Step 4 Upload the OVA metadata.

a. Select the **Metadata** check box.

b. Click **Upload OVA** and select the OVA file.

Example: cucm_15.0_vmv17_v1.2.sha512.OVA

Step 5 Configure additional options.

Select the **Dedicated Cores** check box to allocate dedicated CPU cores for better performance.

Step 6 Submit the image registration request.

Click **Upload File** to submit the request.

NFVIS uploads the ISO, uploads and parse the OVA, register the image, and automatically create profiles and flavors from the OVA metadata.

Step 7 Verify successful registration.

a. Wait for the registration to complete.

Confirm the image appears in the Image Repository with an **Active** status.

b. Verify that flavors are populated from the OVA file.

The ISO image and OVA metadata are successfully registered with NFVIS, and profiles and flavors are automatically created from the OVA metadata. The image appears in the Image Repository with an Active status.

Remote server configuration

Here is a sample configuration to configure a remote server:

```
<remote_servers>
    <remote_server>
        <name>myserver</name>
        <base_url>ENTER_IP_ADDRESS</base_url>
        <username>ENTER_USERNAME</username>
        <password>ENTER_PASSWORD_HERE</password>
    </remote_server>
</remote_servers>
```

Register images using the SCP server

Here is a sample configuration to register images using the SCP server:

```
<images>
    <image>
```

```
        <name>C8Kv_IMAGE</name>
       <src>scp://myserver/nfvis/c8kv.tar.gz</src>  <============ Specify the remote server
 name created in device
        <remove_src_on_completion>
true</remove_src_on_completion>
    </image>
</images>
```

Register images using the HTTP/HTTPS server

Here is a sample configuration to register images using the HTTP/HTTPS server:

```
<images>
    <image>
        <name>C8Kv_IMAGE</name>
        <src>http://myserver/nfvis/c8kv.tar.gz</src>    <============ Specify the remote
server name created in device
        <remove_src_on_completion>
true</remove_src_on_completion>
    </image>
</images>
```

Register VM packages using REST API

This example shows the sequence of registering a tar.gz package on Cisco NFVIS using REST API.

Post image registration

```
curl -v -u -k admin:admin -H Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml
-X POST https://209.165.201.1 /api/config/vm_lifecycle/images -d
'<image><name>WinServer2012R2.iso</name><src>file:///data/intdatastore/uploads/WinServer2012R2.iso/WinServer2012R2.iso</src></image>'

HTTP/1.1 201 Created
```

Get image status

```
curl -k -v -u admin:admin -H Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X
GET https://209.165.201.1/api/operational/vm_lifecycle/opdata/images/image/isrv-03.16.02?deep

HTTP/1.1 200 OK
<image xmlns="http://www.cisco.com/nfvis/vm_lifecycle" xmlns:y="http://tail-f.com/ns/rest"
xmlns:esc="http://www.cisco.com/nfvis/vm_lifecycle">
<name>isrv.03.16.02</name>
<image_id>585a1792-145c-4946-9929-e040d3002a59</image_id>
<public>true</public>
<state>IMAGE_ACTIVE_STATE</state></image>
```

Get registered image status

```
Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X
GET https://209.165.201.1/api/config/vm_lifecycle/images?deep

HTTP/1.1 200 OK
<images xmlns="[http://www.cisco.com/esc/esc|http://www.cisco.com/nfvis/vm_lifecycle]"
xmlns:y="[http://tail-f.com/ns/rest|http://tail-f.com/ns/rest]" 
xmlns:esc="[http://www.cisco.com/nfvis/vm_lifecycle|http://www.cisco.com/nfvis/vm_lifecycle]">
<image>
<name>isrv-9.16.03.01</name>
```

```
<src>http://data/nfvos-pkg/isr/isrv-universalk9.16.03.01.tar.gz</src>
</image>
</images>
```

Delete registered image

```
curl -k -v -u admin:admin -H Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X
DELETE https://209.165.201.1/api/config/vm_lifecycle/images/image/isrv-3.16.0.1a

HTTP/1.1 204 No Content
```

For more information on REST APIs related to image registration, see API Reference for Cisco Enterprise NFVIS.

Register VM image with multiple root disks

If any image requires multiple root disks, you can specify it in the image properties file.

This example shows how to specify multiple root disks in image properties.

```
<image_properties>
 …
 <root_file_disk_bus>virtio</root_file_disk_bus>
 <root_image_disk_format>qcow2</root_image_disk_format>
 <disk_1_file_disk_bus>virtio</ disk_1_file_disk_bus>
 <disk_1_image_format>qcow2</ disk_1_image_format>
 <disk_2_file_disk_bus>virtio</ disk_2_file_disk_bus>
 <disk_2_image_format>qcow2</ disk_2_image_format>
 …
</image_properties>
```

Note Image profiles apply to the first root disk only. The size of the remaining disks remain the same when deployed through different profiles with an exception to vWAAS VNF.

Register a VM image through a root disk

A VM can also be registered using just a disk image (qcow2 or iso) without packaging into a tar.gz. As there will be no image properties in this scenario, the default image properties are used.

Default properties for root disk registration

Property Name	Property Tag	Default Value
Version	<version>	NA
VNF Type	<vnf_typ>	OTHER
VCPU Min	<vcpu_min>	1
VCPU Max	<vcpu_max>	64
Memory Min (MB)	<memory_mb_min>	256
Memory Max (MB)	<memory_mb_max>	1048576
Root Disk Min Size (GB)	<root_disk_gb_min>	1

Property Name	Property Tag	Default Value
Root Disk Max Size (GB)	<root_disk_gb_max>	10240
VNIC Max	<vnic_max>	8
Bootup Time	<bootup_time>	-1
Interface Hot Add	<interface_hot_add>	true
Interface Hot Delete	<interface_hot_delete>	false

Register a remote VM image

Cisco NFVIS allows you to register a VM image that is stored at a remote location or a web server. by specifying the URL to the image location in the source field.

If the web server supports HTTPS, the you can choose to enable Certificate Validation to validate its authenticity. Certificate Validation requires the server's public key to be specified, either in string or file format, in the image registration payload. This allows NFVIS to perform asymmetric encryption and download/register the image file securely over HTTPS.

Example: POST remote image registration from webserver over HTTPS

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Accept:application/vnd.yang.data+xml -H Content-
Type:application/vnd.yang.data+xml -X POST
https://172.29.91.28/api/config/vm_lifecycle/images/ -d '
<image>
 <name>ASAV</name>
 <src>https://172.20.117.124/nfvis/asav982.tar.gz</src>
</image>'

HTTP/1.1 201 Created
```

Example: POST remote image registration from webserver over HTTPS

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Accept:application/vnd.yang.data+xml -H Content-
Type:application/vnd.yang.data+xml -X POST
https://172.29.91.28/api/config/vm_lifecycle/images/ -d '
<image>
 <name>ASAV</name>
 <src>https://172.20.117.124/nfvis/asav982.tar.gz</src>
 <certificate_validation>true</certificate_validation>
 <certificate_file>/data/intdatastore/uploads/pub_key.cert</certificate_file>
</image>'

HTTP/1.1 201 Created
```

Specify storage location for a VM image

Cisco NFVIS allows users to specify the location where the register image should be stored, using the *placement* property tag.

storage name	directory map
datastore1	/data
datastore2	/mnt/extdatastore1
datastore3	/mnt/extdatastore2
nfs:nfs_mount_name	/data/mount/nfs/nfs_mount_name
nfs or nfs:nfs	/data/mount/nfs/
nfs:nfs_storage or nfs_storage	/data/mount/nfs_storage

Note If your preferred storage location is **nfs**, you must have it configured to be mounted on NFVIS using appropriate CLIs before registering the image on it.

Example: VM image storage placement

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Accept:application/vnd.yang.data+xml -H Content-
Type:application/vnd.yang.data+xml -X POST
https://172.29.91.28/api/config/vm_lifecycle/images/ -d '
<image>
 <name>ASAV</name>
 <src>https://172.20.117.124/nfvis/asav982.tar.gz</src>
 <properties>
  <property>
   <name>placement</name>
   <value>nfs:my_nfs_mount</value>
  </property>
 </properties>
</image>'

HTTP/1.1 201 Created
```

Update VM image

Update specific VM image properties after a VM image has been registered to support interface hot add and delete functionality.

You can only update the following image properties after a VM image has been registered:

- interface_hot_add
- interface_hot_delete

Note When using the REST API, the previously set value of the property must be deleted before updating it with the new value.

Procedure

Step 1 Delete the previously set property value.

Example:

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Accept:application/vnd.yang.collection+xml -X DELETE
https://172.29.91.28/api/config/vm_lifecycle/images/image/ISR_IMAGE/properties/property/interface_hot_add/value

HTTP/1.1 204 No Content
```

Step 2 Add (PUT) the new property value to replace the one you deleted in the previous step.

Example:

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X
PUT
https://172.29.91.28/api/config/vm_lifecycle/images/image/ISR_IMAGE/properties/property/interface_hot_add
 --data '<value>true</value>'

HTTP/1.1 201 Created
```

Step 3 Get all properties and values to verify the update.

Example:

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Accept:application/vnd.yang.data+xml -H Content-
Type:application/vnd.yang.data+xml -X GET
https://172.29.91.28/api/config/vm_lifecycle/images/image/ISR_IMAGE/properties?deep

HTTP/1.1 200 OK
<properties xmlns="http://www.cisco.com/nfvis/vm_lifecycle" xmlns:y="http://tail-f.com/ns/rest"
xmlns:vmlc="http://www.cisco.com/nfvis/vm_lifecycle">
  <property>
    <name>interface_hot_add</name>
    <value>true</value>
  </property>
  <property>
    <name>interface_hot_delete</name>
    <value>false</value>
  </property>
</properties>
```

The VM image properties are successfully updated. The system returns the updated properties and values showing the new configuration.

Image Properties

The `image_properties.xml` file is a mandatory component of a VM image package. It contains the essential property configuration data required for the NFVIS image repository to successfully register and deploy a VM. If any mandatory properties are omitted during the registration process, the image registration will fail.

Optional properties can be specified on an image-by-image basis and are not required.

Table 1: Supported Image Properties

Property Name	Description	Property Tag	Possible Values	Mandatory/ Optional
VNF Type	VM functionality provided. Router and firewall are predefined types.	<vnf_type>	Router, firewall, Windows, Linux, and custom_type	Mandatory
Name	Name associated with the VM packaging. This name is referenced for VM deployment.	<name>	Any	Mandatory
Version	Version of the package	<version>	Any	Mandatory
Boot-up time	Boot-up time (in seconds) of the VNF before it can be reachable via ping.	<bootup_time>	Any in seconds, (-1) to not monitor boot-up	Optional, default -1
Root Disk Image Bus	Root image disk bus	<root_file_disk_bus>	virtio, scsi, and ide	Mandatory
Boot Mode	Specifies the mode in which the VNF will boot. Used for Secure Boot feature	<boot_mode>	efi-secure-boot, bios	Optional, default bios
Shim Signature	If using efi-secure-boot boot mode, a shim signature must be provided	<shim_signature>	microsoft, N/A	Required if Boot Mode is specified
Disk-1 bus type	Additional disk1 image disk bus	<disk_1_file_disk_bus>	virtio, scsi, and ide	Optional

Property Name	Description	Property Tag	Possible Values	Mandatory/ Optional
Disk-2 bus type	Disk2 image disk bus	<disk_2_file_disk_bus>	virtio, scsi, and ide	Optional
Disk-10 bus type	Disk10 image disk bus	<disk_10_file_disk_bus>	virtio, scsi, and ide	Optional
Root Disk Image format	Root image disk format	<root_image_disk_format>	qcow2 and raw	Mandatory
Disk-1 Image format	Additional disk 1 image format	<disk_1_image_format>	qcow2 and raw	Optional
Disk-2 Image format	Disk 2 image format	<disk_2_image_format>	qcow2 and raw	Optional
Disk-10 Image format	Disk 10 image format	<disk_10_image_format>	qcow2 and raw	Optional
Serial Console	Serial console supported	<console_type_serial>	true, false	Optional
Minimum vCPU	Minimum vCPUs required for a VM operation	<vcpu_min>		Mandatory
Maximum vCPU	Maximum vCPUs supported by a VM	<vcpu_max>		Mandatory
Minimum memory	Minimum memory in MB required for VM operation	<memory_mb_min>		Mandatory
Maximum memory	Maximum memory in MB supported by a VM	<memory_mb_max>		Mandatory
Minimum root disk size	Minimum disk size in GB required for VM operation	<root_disk_gb_min>		Optional
Maximum root disk size	Maximum disk size in GB supported by a VM	<root_disk_gb_max>		Optional

Property Name	Description	Property Tag	Possible Values	Mandatory/ Optional
Maximum vNICs	Maximum number of vNICs supported by a VM	<vnic_max>		Mandatory
SRIOV support	SRIOV supported by VM interfaces. This should have a list of supported NIC device drivers.	<sriov_supported>	true, false	Optional
SRIOV driver list	List of drivers to enable SRIOV support	< sriov_driver_list>		Optional
PCI passthru support	PCI passthru support by VM interfaces	<pcie_supported>	true, false	Optional
PCIE driver list	List of VNICS to enable PCI passthru support	< pcie _driver_list>		Optional
bootstrap_cloud_init_drive_type	Mounts day0 config file as disk (default is CD-ROM)	<bootstrap_cloud_init_drive_type>	disk, cdrom	Optional
bootstrap_cloud_init_bus_type	Default is IDE	<bootstrap_cloud_init_bus_type>	virtio, ide	Optional

Property Name	Description	Property Tag	Possible Values	Mandatory/ Optional
BOOTSTRAP	Bootstrap files for the VNF. Two parameters are required in the format of dst:src; dst filename including path has to match exactly to what the VM expects; up to 20 bootstrap files are accepted. For example: --bootstrap ovf-env.xml for ISRv and --bootstrap day0-config for ASAv	< bootstrap_file>	File name of the bootstrap file	Optional
Custom properties	List of properties can be defined within the custom_property tree. (Example: For ISRv, the technology packages are listed in this block.) If the Cisco NFV portal is used to deploy the VM, the portal prompts you for inputs for custom properties fields and can pass the values to the bootstrap configuration.	<custom_property>		Optional

Property Name	Description	Property Tag	Possible Values	Mandatory/ Optional
Profiles for VM deployment	List of VM deployment profiles. Minimum one profile is required	<profiles>		Optional
Default profile	The default profile is used when no profile is specified during deployment.	<default_profile>		Optional
Monitoring Support	A VM supports monitoring to detect failures.	<monitoring_supported>	true, false	Mandatory
Monitoring Method	A method to monitor a VM. Currently, only ICMP ping is supported.	<monitoring_methods>	ICMPPing	Mandatory if monitoring is true
Low latency	If a VM's low latency (for example, router and firewall) gets dedicated resource (CPU) allocation. Otherwise, shared resources are used.	<low_latency>	true, false	Mandatory
Privileged-VM	Allows special features like promiscuous mode and snooping . By default, it is false.	<privileged_vm>	true, false	Optional
Disable Spoof Check	Used to disable spoof check for Privledged VMs	<disable_spoof_check>	true, false	Optional
Virtual interface model		<virtual_interface_model>		Optional

Property Name	Description	Property Tag	Possible Values	Mandatory/ Optional
Interface Hot Add	If true, an active VNF's virtual interface can be added/updated without shutting down the VNF.	<interface_hot_add>	true, false	Optional, default true
Interface Hot Delete	If true, an active VNF's virtual interface can be deleted without shutting down the VNF.	<interface_hot_delete>	true, false	Optional, default false
Thick disk provisioning	During deployment, VM will be a fully allocated root disk with size specified by flavor.	<thick_disk_provisioning>	true, false	Optional, default false
Eager Zero	Used in conjunction with Thick disk provisioning. During deployment, root disk is zeroed out to improve I/O operations	<eager_zero>	true, false	Optional, only valid if Thick disk provisioning is enabled. Default false
Profile for VM deployment	A profile defines the resources required for VM deployment. This profile is referenced during VM deployment.	<profile>		Optional
Name	Profile name	<name>	Any	Mandatory
Description	Description of the profile	<description>	Any	Mandatory

Property Name	Description	Property Tag	Possible Values	Mandatory/ Optional
vCPU	vCPU number in a profile	<vcpus>		Mandatory
Memory	Memory - MB in profile	<memory_mb>		Mandatory
Root Disk Size	Disk size - MB in profile .	<root_disk_mb>		Mandatory
VNIC Offload	List of properties that can be set for vnic offload	<vnic_offload>		Optional
Generic Segmentation Offload	Turn generic segmentation offload on or off	<generic_segmentation_offload> (parent: <vnic_offload>)	on, off	Optional
Generic Receive Offload	Turn generic receive offload on or off	<generic_receive_offload> (parent: <vnic_offload>)	on, off	Optional
RX Checksumming	Turn RX checksumming on or off	<rx_checksumming> (parent: <vnic_offload>)	on, off	Optional
TX Checksumming	Turn TX checksumming on or off	<tx_checksumming> (parent: <vnic_offload>)	on, off	Optional
TCP Segmentation Offload	Turn TCP segmentation offload on or off	<tcp_segmentation_offload> (parent: <vnic_offload>)	on, off	Optional

Contents of an image_properties.xml File

```
<?xml version="1.0" encoding="UTF-8"?>
<image_properties>
    <vnf_type>ROUTER</vnf_type>
    <name>ISRV</name>
    <version>16.06.05</version>
    <bootup_time>600</bootup_time>
    <root_file_disk_bus>virtio</root_file_disk_bus>
    <root_image_disk_format>qcow2</root_image_disk_format>
    <vcpu_min>1</vcpu_min>
    <vcpu_max>8</vcpu_max>
    <memory_mb_min>4096</memory_mb_min>
    <memory_mb_max>8192</memory_mb_max>
    <vnic_max>8</vnic_max>
    <vnic_names>vnics:1:GigabitEthernet2</vnic_names>
    <vnic_names>vnics:2:GigabitEthernet3</vnic_names>
    <vnic_names>vnics:3:GigabitEthernet4</vnic_names>
    <vnic_names>vnics:4:GigabitEthernet5</vnic_names>
```

```
    <vnic_names>vnics:5:GigabitEthernet6</vnic_names>
    <vnic_names>vnics:6:GigabitEthernet7</vnic_names>
    <vnic_names>vnics:7:GigabitEthernet8</vnic_names>
    <root_disk_gb_min>8</root_disk_gb_min>
    <root_disk_gb_max>8</root_disk_gb_max>
    <console_type_serial>true</console_type_serial>
    <sriov_supported>true</sriov_supported>
    <sriov_driver_list>igb</sriov_driver_list>
    <sriov_driver_list>igbvf</sriov_driver_list>
    <sriov_driver_list>i40evf</sriov_driver_list>
    <pcie_supported>true</pcie_supported>
    <pcie_driver_list>igb</pcie_driver_list>
    <pcie_driver_list>igbvf</pcie_driver_list>
    <pcie_driver_list>i40evf</pcie_driver_list>
    <monitoring_supported>true</monitoring_supported>
    <monitoring_methods>ICMPPing</monitoring_methods>
    <low_latency>true</low_latency>
    <privileged_vm>true</privileged_vm>
    <cdrom>true</cdrom>
    <bootstrap_file_1>ovf-env.xml</bootstrap_file_1>
    <bootstrap_file_2>iosxe_config.txt</bootstrap_file_2>
    <custom_property>
        <tech_package>ax</tech_package>
        <tech_package>security</tech_package>
        <tech_package>ipbase</tech_package>
        <tech_package>appx</tech_package>
    </custom_property>
    <custom_property>
        <SSH_USERNAME> </SSH_USERNAME>
    </custom_property>
    <custom_property>
        <SSH_PASSWORD> </SSH_PASSWORD>
    </custom_property>
    <profiles>
        <profile>
    <name>ISRv-mini</name>
            <description>ISRv-mini</description>
    <vcpus>1</vcpus>
            <memory_mb>4096</memory_mb>
            <root_disk_mb>8192</root_disk_mb>
</profile>
        <profile>
    <name>ISRv-small</name>
            <description>ISRv-small</description>
    <vcpus>2</vcpus>
            <memory_mb>4096</memory_mb>
            <root_disk_mb>8192</root_disk_mb>
</profile>
        <profile>
    <name>ISRv-medium</name>
            <description>ISRv-medium</description>
    <vcpus>4</vcpus>
            <memory_mb>4096</memory_mb>
            <root_disk_mb>8192</root_disk_mb>
</profile>
    </profiles>
    <default_profile>ISRv-small</default_profile>
</image_properties>
```

Register a remote virtual machine image

Use this task to register a Virtual Machine (VM) image hosted on a remote server (HTTPS or SCP) for use in Cisco NFVIS. This process allows NFVIS to download the ISO image and OVA file, parse the metadata, and automatically create deployment profiles and flavors.

Table 2: Protocols

Protocol	Authentication	Notes
HTTPS	Not supported	No Username and password
SCP	Required	Username and password

Before you begin

Follow these steps to register a remote virtual machine image:

Procedure

Step 1 From the Cisco NFVIS portal, choose **Configuration** > **Virtual Machine** > **Images** > **Image Repository**.

Step 2 Select the remote registration option:

a. Click **Register Image**

b. Select **Remote Image Registration**

Step 3 Configure image properties.

Enter a descriptive name in the **Image name** field.

Step 4 Configure the remote server connection.

a. Select the Protocol from the drop-down list (HTTPS or SCP).

b. Enter the server hostname or IP address in the IP Address field.

c. Enter the directory path to the ISO file in the Image File Path field.

Step 5 Configure authentication.

If you selected FTP, SFTP, or SCP as the protocol, enter the username and password for the remote server.

Note
Note: Authentication is not supported for HTTP or HTTPS protocols.

Step 6 Upload the OVA metadata.

a. Select the **Metadata** check box.

b. Enter the path to the OVA file in the **OVA File Path** field.

Example: /home/user/images/cucm_15.0_vmv17_v1.2.sha512.ova

Note

Note: The OVA file must reside on the same remote server as the ISO file.

Step 7 Configure additional options.

Select the **Dedicated Cores** check box if dedicated CPU cores are required for the virtual machine.

Step 8 Submit the registration request.

Click **Submit** to submit the registration request.

NFVIS downloads the ISO image and OVA file, parses the OVA file, registers the image, and automatically creates profiles and flavors from the metadata.

Step 9 Monitor the registration progress.

Track the registration progress in the Image Repository. Downloading large files can take several minutes.

The VM image is registered, and the system automatically creates the necessary profiles and flavors, making the image available for deployment.

Customization of VM's

VM profiles or flavors

A VM profile or flavor is a configuration template that

- defines VMs in terms of number of parameters for how to run the VM
- specifies parameters such as number of vCPUs, RAM, disk size and so on, and
- provides standardized VM configurations based on requirements.

VM profile creation methods

VM profiles are created using different methods depending on the image package type:

- Flavors are created as part of image registration if you use the tar.gz image packages for registering a VM.
- For other image packages such as .qcow2, iso, and raw, you must define custom flavors based on your requirements.

Note Unless specified otherwise in the deployment payload, the value assigned to the custom image property *default_profile* is used at the time of deploying the VM. Only applicable to *tar.gz* image packages.

Create VM profile using REST API

Create a VM profile to specify the resource allocation and configuration parameters for virtual machines in your environment.

VM profiles define the hardware specifications for virtual machines including memory allocation, disk space, and CPU configuration. Use REST API calls to programmatically create these profiles for consistent VM deployment.

Procedure

Execute the following cURL command to create a VM profile using the REST API.

Example:

```
curl -k -v -u admin:admin -H Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X POST https://209.165.201.1/api/config/vm_lifecycle/flavors
 -d '<flavor>
    <name>windows</name>
    <ephemeral_disk_mb>0</ephemeral_disk_mb>
    <memory_mb>4096</memory_mb>
    <root_disk_mb>12288</root_disk_mb>
    <swap_disk_mb>0</swap_disk_mb>
    <vcpus>2</vcpus>
</flavor>'
```

The VM profile is successfully created with the specified configuration parameters and can be used for virtual machine deployment.

Internal management networks

An internal management network is a separate network within the Cisco NFVIS infrastructure that

- handles communication between the Cisco NFVIS and other management-related functions like port forwarding
- is used for VM monitoring and recovery, and
- is created by Cisco NFVIS as a network named **int-mgmt-net** by default.

Default configuration behavior

Here's an example of the default int-mgmt-net configuration:

```
vm_lifecycle networks network int-mgmt-net
 subnet int-mgmt-net-subnet
  address 10.20.0.0
  netmask 255.255.255.0
  gateway 10.20.0.1
 !
!
```

You can update the default configuration before deploying VMs/VNFs. When the default config is deleted, Cisco NFVIS recreates the internal management network upon rebooting the host.

VNF deployment monitoring: Cisco NFVIS lets you configure the VNF deployment to be monitored. To monitor the VM deployment, an interface is attached to the internal management network (int-mgmt-net). The interface attached contains the NIC ID **0** by default.

Table 3: Example: Interface attachment to internal management network

Internal Management Network in the VM Deployment Payload	IP ADDRESS token/variable in Day-0 config	Behavior

Example configuration using IP address:
`<interface>`
`<nicid>0</nicid>`
`<network>int-mgmt-net</network>`
`<ip_address>10.20.0.21</ip_address>`
`</interface>`
	Token ${NICID_0_IP_ADDRESS}" is either present or absent.	If the IP payload is available, the payload is assigned to the interface.
		If there is a token/variable, the IP ADDRESS is replaced with the token.
		When the IP ADDRESS in the IP payload is unavailable or incorrect, the deployment fails.

Example configuration when no IP ADDRESS is configured:
`<interface>`
`<nicid>0</nicid>`
`<network>int-mgmt-net</network>`
`</interface>`
	${NICID_0_IP_ADDRESS} token is present in the day-0 config.	Cisco NFVIS auto assigns the IP ADDRESS and the token/variable is replaced in the day-0 config with the same IP ADDRESS.

Example configuration when no IP ADDRESS is configured with the NIC ID being absent:
`<interface>`
`<nicid>0</nicid>`
`<network>int-mgmt-net</network>`
`</interface>`
	Not present
	By default, the interface is configured as DHCP. The token is not replaced.
		Note
		Starting from Cisco NFVIS Release 4.12.1, configuring an interface as a DHCP server is supported.
		The IP ADDRESS is auto allocated using the internal DHCP server. Cisco NFVIS utilizes the IP ADDRESS for monitoring the deployment.

Here is a sample day-0 config with the IP address and the token/variable:

```
<?xml version="1.0" encoding="UTF-8"?>
<Environment
xmlns:oe="http://schemas.dmtf.org/ovf/environment/1">
  <PropertySection>
    <Property oe:key="com.cisco.c8000v.config-version.1" oe:value="1.0"/>
    <Property oe:key="com.cisco.c8000v.enable-ssh-server.1" oe:value="True"/>
    <Property oe:key="com.cisco.c8000v.login-username.1" oe:value="${SSH_USERNAME}"/>
    <Property oe:key="com.cisco.c8000v.login-password.1" oe:value="${SSH_PASSWORD}"/>
    <Property oe:key="com.cisco.c8000v.mgmt-interface.1" oe:value="GigabitEthernet1"/>
!!!GigabitEthernet1-nicid(0)-int-mgmt-interface-don't change ip address or don't shutdown
    <Property oe:key="com.cisco.c8000v.mgmt-ipv4-addr.1"
oe:value="${NICID_0_IP_ADDRESS}/${NICID_0_CIDR_PREFIX}"/>         ===> IP address set via
token
    <Property oe:key="com.cisco.c8000v.mgmt-ipv4-network.1" oe:value=""/>
    <Property oe:key="com.cisco.c8000v.license.1" oe:value="${TECH_PACKAGE}"/>
   <Property oe:key="com.cisco.c8000v.ios-config-0001" oe:value="vrf definition Mgmt-intf"/>

    <Property oe:key="com.cisco.c8000v.ios-config-0002" oe:value="address-family ipv4"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0003" oe:value="exit-address-family"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0004" oe:value="address-family ipv6"/>
```

```
    <Property oe:key="com.cisco.c8000v.ios-config-0005" oe:value="exit-address-family"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0006" oe:value="exit"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0007" oe:value="interface
GigabitEthernet1"/>
   <Property oe:key="com.cisco.c8000v.ios-config-0008" oe:value="vrf forwarding Mgmt-intf"/>

    <Property oe:key="com.cisco.c8000v.ios-config-0009" oe:value="ip address
${NICID_0_IP_ADDRESS} ${NICID_0_NETMASK}"/>   ===> IP address set via token
    <Property oe:key="com.cisco.c8000v.ios-config-0010" oe:value="no shut"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0011" oe:value="exit"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0012" oe:value="ip route vrf Mgmt-intf
0.0.0.0 0.0.0.0 ${NICID_0_GATEWAY}"/>
  </PropertySection>
</Environment>
```

Here is a sample day-0 config using DHCP server:

```
<?xml version="1.0" encoding="UTF-8"?>
<Environment
xmlns:oe="http://schemas.dmtf.org/ovf/environment/1">
  <PropertySection>
    <Property oe:key="com.cisco.c8000v.config-version.1" oe:value="1.0"/>
    <Property oe:key="com.cisco.c8000v.enable-ssh-server.1" oe:value="True"/>
    <Property oe:key="com.cisco.c8000v.login-username.1" oe:value="${SSH_USERNAME}"/>
    <Property oe:key="com.cisco.c8000v.login-password.1" oe:value="${SSH_PASSWORD}"/>
    <Property oe:key="com.cisco.c8000v.mgmt-interface.1" oe:value="GigabitEthernet1"/>
!!!GigabitEthernet1-nicid(0)-int-mgmt-interface-don't change ip address or don't shutdown
    <Property oe:key="com.cisco.c8000v.mgmt-ipv4-addr.1" oe:value=""/>
                  ===> IP address left blank here
    <Property oe:key="com.cisco.c8000v.mgmt-ipv4-network.1" oe:value=""/>
    <Property oe:key="com.cisco.c8000v.license.1" oe:value="${TECH_PACKAGE}"/>
   <Property oe:key="com.cisco.c8000v.ios-config-0001" oe:value="vrf definition Mgmt-intf"/>

    <Property oe:key="com.cisco.c8000v.ios-config-0002" oe:value="address-family ipv4"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0003" oe:value="exit-address-family"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0004" oe:value="address-family ipv6"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0005" oe:value="exit-address-family"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0006" oe:value="exit"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0007" oe:value="interface
GigabitEthernet1"/>
   <Property oe:key="com.cisco.c8000v.ios-config-0008" oe:value="vrf forwarding Mgmt-intf"/>

   <Property oe:key="com.cisco.c8000v.ios-config-0009" oe:value="ip address dhcp setroute"/>
              ===> DHCP specified vs IP address
    <Property oe:key="com.cisco.c8000v.ios-config-0010" oe:value="no shut"/>
    <Property oe:key="com.cisco.c8000v.ios-config-0011" oe:value="exit"/>
  </PropertySection>
</Environment>
```

VNF volumes

A VNF can be created and deployed with maximum two volumes. Currently, NFVIS supports a maximum of two volumes per VNF. The empty volume can be used as an extra storage by the VNF. Starting from NFVIS 4.1 release, storage volumes can now be deployed on external datastores and NFS.

Restrictions for VNF volumes

- Volumes cannot be updated for a VNF after a VNF has already been deployed.
- NFVIS currently supports only two volumes per VNF.
- Starting from NFVIS 4.1 release, volumes can be stored in local storage or NFS, and other data stores like datastore1, datastore2 and datastore3.

Example: payload for creating volumes

```
…
<volumes>
  <volume>
    <name>Volume1</name>
    <volid>1</volid>
    <bus>ide</bus>
    <size>1</size>
    <sizeunit>GiB</sizeunit>
    <format>qcow2</format>
    <device_type>disk</device_type>
    <storage_location>local</storage_location>
  </volume>
</volumes>
…
```

Volume storage locations

The following are the accepted values for **storage_location** tags:

- <**storage_location**>*local*</**storage_location**>
- <**storage_location**>*nfs:NFS_MOUNT_NAME*</**storage_location**>
- <**storage_location**>*datastore1*</**storage_location**>
- <**storage_location**>*datastore2*</**storage_location**>
- <**storage_location**>*datastore3*</**storage_location**>

Volume deployment parameters

Cisco NFVIS supports the fields for volume deployment shown in this table.

Property	Allowed Values	Description
NAME	string, { length 1..255 }	NAME of the volume
volid	uint16	Volumes will be presented to the VM sorted by volume ID.
bus	virtio, ide, scsi	The bus type
size	unit 16	Size of the Volume
sizeunit	MiB,GiB,TiB,PiB	Size units. MiB/GiB/TiB/PiB/EiB
format	qcow2, raw	Format of the disk to be created.
device_type	disk	Type of the device being attached to the VM.
storage_location	local, datastore1, datastore2, datastore3, nfs:NFS_MOUNT_NAME	Storage location NAME **Note** Starting from NFVIS 4.1 release, external datastore storage location is supported.

Port forwarding

Port forwarding is a network configuration mechanism that

- redirects incoming traffic from WAN to access the internal management network of the VM
- uses the WAN bridge interface (**WAN-br**) by default for traffic redirection, and
- allows modification of the bridge interface using the **source_bridge** tag in the deployment payload.

Port forwarding configuration

The bridge interface that is used to redirect traffic coming from the WAN side can be modified using the **source_bridge** tag in the deployment payload as shown in this example:

```
<port_forwarding>
  <port>
  <type>ssh</type>
  <protocol>tcp</protocol>
  <vnf_port>22</vnf_port>
  <source_bridge>MGMT</source_bridge>
  <external_port_range>
    <start>20122</start>
    <end>20122</end>
  </external_port_range>
  </port>
</port_forwarding>
```

With this payload, the traffic coming from the WAN side is redirected through the management interface **(MGMT)** instead of the default WAN bridge **(WAN-br)** interface.

vCPU pinning

vCPU pinning is a resource allocation mechanism that

- assigns containers to specific virtual CPUs for dedicated processing
- enables low latency configuration through dedicated core allocation
- reuses the Node.js CPU allocation API to interface with NFVIS CPU allocation scripts.

vCPU pinning configuration

Similar to the NFVIS VM model, containers can also be pinned to a specific vCPU. If a container requires a dedicated core, a low latency flag can be set using the configuration options. By default, no low latency flag is set, this means that the core may be shared with other containers. Because the workflow for container vCPU pinning is the same as VM deployments, VimManager reuses the Node.js CPU allocation API to interface with the NFVIS CPU allocation scripts.

Volumes support for container deployments uses the same datamodel as VM deployments and provides external storage options to the containers. The `storage_location` tag in the payload refers to the file path within the container where the volume is mounted.

Note

- Volumes are equivalent to docker binds.
- All container volumes are stored on datastore1.

Example: Container volume configuration

To mount a container using a volume or a specific path, use this configuration:

```
<vm_group>
    <name>docker-nginx</name>
    <image>nginx</image>
    . . .
    <volumes>
      <volume>
        <name>test1</name>
        <volid>1</volid>
        <storage_location>/etc/datastore</storage_location> <!-- the path to mount inside
the container -->
        <size>120</size>
        <sizeunit>MiB</sizeunit>
      </volume>
      <!-- using default size allocations - 10 gibibytes (GiB) -->
      <volume>
        <name>test2</name>
        <volid>2</volid>
        <storage_location>/var/logs</storage_location>
      </volume>
```

For more information, see VNF volumes, on page 24.

VM deployment and management

VM deployment

VM deployment is a virtualization process that

- creates and configures virtual machines through API or CLI methods
- requires prior registration of VM images before deployment can begin, and
- accepts mandatory and optional parameters to customize the deployment configuration.

VM name requirements

The VM name must meet these requirements:

- Must contain an uppercase character and a lowercase character.
- Must contain a digit.
- Must contain one of the following special characters: dot (.), underscore (_) and hyphen (-).
- Must not have more than 256 characters.

Note Ensure that you have registered a VM image before attempting to deploy it. For more details, see *Image Registration*.

Example: deploy VMs using REST API

This example demonstrates how to deploy a VM using REST API with a sample payload and how to verify the deployment status.

This is a sample payload of deploying a VM.

```
curl -k -v -u admin:admin -H Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X POST
https://209.165.201.1/api/config/vm_lifecycle/tenants /tenant/admin/deployments --data '
<deployment>
  <name>ISRdep</name>
  <vm_group>
    <name>ISRvmgrp</name>
    <image>ISR_IMAGE</image>
    <bootup_time>500</bootup_time>
    <recovery_wait_time>0</recovery_wait_time>
    <interfaces>
      <interface>
        <nicid>0</nicid>
        <network>int-mgmt-net</network>
        <ip_address>10.20.0.21</ip_address>
        <port_forwarding>
          <port>
            <type>ssh</type>
            <protocol>tcp</protocol>
            <vnf_port>22</vnf_port>
            <external_port_range>
              <start>20022</start>
              <end>20022</end>
            </external_port_range>
          </port>
        </port_forwarding>
      </interface>
    </interfaces>
    <kpi_data>
      <kpi>
        <event_name>VM_ALIVE</event_name>
        <metric_value>1</metric_value>
        <metric_cond>GT</metric_cond>
        <metric_type>UINT32</metric_type>
        <metric_collector>
          <type>ICMPPing</type>
          <nicid>0</nicid>
          <poll_frequency>3</poll_frequency>
          <polling_unit>seconds</polling_unit>
          <continuous_alarm>false</continuous_alarm>
        </metric_collector>
      </kpi>
    </kpi_data>
    <rules>
      <admin_rules>
        <rule>
          <event_name>VM_ALIVE</event_name>
          <action>ALWAYS log</action>
          <action>TRUE servicebooted.sh</action>
          <action>FALSE recover autohealing</action>
        </rule>
      </admin_rules>
    </rules>
    <config_data>
      <configuration>
        <dst>bootstrap_config</dst>
        <variable>
```

```
          <name>TECH_PACKAGE</name>
          <val>ax</val>
        </variable>
      </configuration>
    </config_data>
  </vm_group>
</deployment>'
```

Verify VM deployment

This example shows how to get the operational data for a VM deployment using the command

```
show vm_lifecycle opdata tenants tenant admin deployments
<deployment_name>/<deployment_id>/<vmgroup_name>
```

```
nfvis# show vm_lifecycle opdata tenants tenant admin deployments ROUTER
deployments ROUTER
 deployment_id SystemAdminTenantIdROUTER
 vm_group ROUTER
  bootup_time 600
  vm_instance d1c462e9-2706-4868-befd-d8f7806b9444
   name     ROUTER
   host_id  NFVIS
   hostname nfvis
   interfaces interface 0
    model       virtio
    type        virtual
    port_id     vnic0
    network     int-mgmt-net
    subnet      N/A
    ip_address  10.20.0.2
    mac_address 52:54:00:fe:34:53
    netmask     255.255.255.0
    gateway     10.20.0.1
   interfaces interface 1
    type        virtual
    port_id     vnic1
    network     GE0-1-SRIOV-1
    subnet      N/A
    mac_address 52:54:00:e4:13:67
 state_machine state SERVICE_ACTIVE_STATE
VM
NAME    STATE
------------------------
ROUTER  VM_ALIVE_STATE
```

Note If the deployment was accepted first but failed later, NFVIS sends a notification with error status and message. Also on VM deployment failure, the operational data may or may not show the complete data about the failed VM.

VM deployment parameters

VNFs can be deployed using multiple mandatory and option parameters.

Parameter	Notes	Example
IMAGE	Mandatory. The IMAGE needs to be registered and active when it is being referred in deployment.	**<IMAGE>ISR_IMAGE</IMAGE>**
bootup_time	This parameter is no longer mandatory from 3.12 and later, provided that it is specified in IMAGE properties. **Accepted Values:** • For unmonitored VMs: -1 • For monitored VMs: Number of seconds	**<bootup_time>500</bootup_time>**
vim_vm_name	Optional. If a custom VM name is provided at the time of deployment, it may be used for all commands that accept VM name as an input.	

Parameter	Notes	Example
kpi_data	Mandatory for monitored VMs.	
rules	Mandatory for monitored VMs.	
config_data	Mandatory if the day-0 configuration has variables that have tokens assigned to them.	
Encrypted config VARIABLE	Optional. Only one value for a VARIABLE is allowed to be encrypted.	**<VARIABLE>** **<name>TEST_VARIABLE</name>** **<encrypted_val>test_value</encrypted_val>** **</VARIABLE>**

Parameter	Notes	Example
placement	Optional. The **placement** tag under VM group points to the location where the VNF would be deployed. This parameter supports deploying a VNF in a local data store (default-if not specified), external data store (datastore2), or NFS.	**<placement>** **<type>zone_host</type>** **<host>nfs:nfs_storage</host>** **</placement>**
volumes	Optional. Up to 2 volumes could be added to a deployment. Location of the volumes can be local or NFS (needs NFS mount name to be specified in case of NFS)	

Parameter	Notes	Example
port_forwarding	Optional. If port forwarding is included, all elements under it are mandatory.	**<port_forwarding>** **<port>** **<type>ssh</type>** **<protocol>tcp</protocol>** **<vnf_port>22</vnf_port>** **<external_port_range>** **<start>20022</start>** **<end>20022</end>** **</external_port_range>** **</port>** **</port_forwarding>**
Ngio interface	Optional. Used in config_data. To enable NIM support on a Cisco ISRv running on Cisco ENCS, you must use the variables in the ISRv deployment payload.	**<VARIABLE> <name>ngio</name> <val>enable</val> </VARIABLE>**
interface model	Optional. If the model is not specified for an interface, the default model is used. For Windows, the default model is rtl8139.	**<interface><nicid>3</nicid><network>wan-net</network><model>virtio</mode**

Parameter	Notes	Example
VNC	Optional. If the VNC password is not specified, there is no default password.	**<VNC><password>vnc_password</password></VNC>**

VM bootstrap configuration options with VM deployment

VM bootstrap configuration options with VM deployment are methods that

- enable inclusion of bootstrap configuration (day zero configuration) of a VM in the VM deployment payload
- provide flexibility in how configuration files are handled during deployment, and
- support different approaches based on deployment requirements and infrastructure setup.

Bootstrap configuration methods

You can include the bootstrap configuration (day zero configuration) of a VM in the VM deployment payload in these ways:

- Bundle bootstrap configuration files into the VM package: In this method, the bootstrap configuration variables can be assigned tokens. Token names must be in bold text. For each variable with a token assigned, key-value pairs must be provided during deployment in the deployment payload.
- Bootstrap configuration as part of the deployment payload: The entire bootstrap configuration is copied to the payload without tokens.
- Bootstrap configuration file in the NFVIS server: In this method, the configuration file is copied or downloaded to the NFVIS server, and referenced from the deployment payload with the filename, which includes the full path.

For examples on how to use bootstrap configuration options in the deployment payload, see the API Reference for Cisco Enterprise Network Function Virtualization Infrastructure Software.

VNF deployment placement

VNF deployment placement is a parameter mechanism that

- allows you to specify where a VNF should be deployed using the placement tag for parameter deployment
- supports various placement parameters with accepted values, and
- enables precise control over VNF location within the Cisco NFVIS environment.

Placement configuration requirements

See VNF Deployment Parameters for more information on supported placement parameters and their accepted values.

Note If you are placing the VNF deployment on **nfs**, ensure that you have configured this storage option to be mounted on NFVIS using appropriate CLIs before deploying the VNF.

Example: VM deployment using placement

```
curl -k -v -u admin:admin -H Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X POST
https://209.165.201.1/api/config/vm_lifecycle/tenants
/tenant/admin/deployments --data
'<deployment>
<name>WINIsodep</name>
<vm_group>
  <name>WINIsovmgrp</name>
  <image>WinServer2012R2.iso</image>
  <flavor>windows</flavor>
  <bootup_time>-1</bootup_time>
  <recovery_wait_time>0</recovery_wait_time>
  <kpi_data>
   <enabled>true</enabled>
  </kpi_data>
   <scaling>
   <min_active>1</min_active>
   <max_active>1</max_active>
   <elastic>true</elastic>
   </scaling>
   <placement>
   <type>zone_host</type>
   <enforcement>strict</enforcement>
   <host>datastore1</host>
   </placement>
   <recovery_policy>
   <recovery_type>AUTO</recovery_type>
   <action_on_recovery>REBOOT_ONLY</action_on_recovery>
   </recovery_policy>
</vm_group>
</deployment>'
```

Deploy VNF using granular RBAC

Deploy Virtual Network Functions (VNFs) using granular role-based access control (RBAC) to ensure proper access management and security.

The system administrator can define a set of groups and assign VNFs to these groups. When you create a user, you can assign that user to one of these groups to control VNF access. For more information on granular RBAC and resource groups, see Granular role-based access control.

Procedure

Step 1 Create a VM with a group name.

Example:

```
nfvis(config)# vm_lifecycle tenants tenant admin deployments deployment sample resource_group
localgroup2 vm_group test image centos flavor small vim_vm_name sample bootup_time -1 recovery_wait_time
 0 recovery_policy recovery_type AUTO action_on_recovery REBOOT_ONLY monitor_on_error false
nfvis(config-vm_group-test)# commit
```

Step 2 Update a VM from one resource group to another.

Example:

```
nfvis(config)# vm_lifecycle tenants tenant admin deployments deployment sample
nfvis(config-deployment-sample)# no resource_group
nfvis(config-deployment-sample)# commit
Commit complete.
nfvis(config-deployment-sample)# resource_group localgroup
nfvis(config-deployment-sample)# commit
Commit complete.
```

Step 3 View the list of VMs with their group information.

Example:

```
nfvis# show vm_lifecycle deployments

Name          Monitored  State           Internal-State           GroupInfo
-------------------------------------------------------------------------------
centos1       No         ALIVE           VM_INERT_STATE           localgroup
cetos_global  No         ALIVE           VM_INERT_STATE                                Denotes global
sample        No         ALIVE           VM_INERT_STATE           localgroup
```

Step 4 View the overall group, users and VM mappings.

Example:

```
nfvis# show group-summary

Group_Name             Policies                     Users                     Vms
-------------------------------------------------------------------------------------------------
localgroup         resource-access-control        local_oper            centos1.centos1
                                                  sample.test
-------------------------------------------------------------------------------------------------
localgroup2        resource-access-control         test_admin
-------------------------------------------------------------------------------------------------
                          NA                        admin            cetos_global.cetos_global
                                                   localAdmin
-------------------------------------------------------------------------------------------------
```

The VNF is deployed with granular RBAC configuration, and you can view the resource group assignments and user mappings for proper access control.

Deploy CUCM application

Deploy a CUCM virtual machine on Cisco NFVIS using a Bootstrap ISO for automated Day0 configuration.

This procedure is performed after successfully registering the application image in NFVIS. The deployment creates a functional CUCM virtual machine with automated initial configuration.

Before you begin

- Ensure you have a registered CUCM application image (ISO and OVA) in NFVIS.
- Ensure you have a Bootstrap ISO with Day0 configuration for the CUCM application.

Refer to the CUCM documentation for instructions on creating the Bootstrap ISO with Day0 configuration: *Touchless Installation Task Flow*

Follow these steps to deploy a CUCM virtual machine on Cisco NFVIS with a Bootstrap ISO for Day0 configuration:

Procedure

Step 1 From the NFVIS portal, click **Configuration** > **Deploy**.

Step 2 Select a VM type from the drop-down list.

If your application type is not listed, select **OTHER**.

Step 3 From the **Image** drop-down list, select the desired image.

Images are populated based on the VM type selected. For example, `Bootable_UCSInstall_UCOS_15.0.1.ISO`.

Step 4 From the **Flavor** drop-down list, select the appropriate flavor based on your sizing requirements.

Flavors are automatically created from the OVA during image registration.

Step 5 Configure network connections.

a. In the **Network Design** section, locate the VM icon on the canvas.

b. Add additional networks as required.

c. Drag a line from the VM icon to the desired network to connect to the appropriate network(s) for your deployment.

Step 6 Select the **Bootstrap ISO** check box.

Step 7 Click **Upload** and select your Day0 configuration ISO.

Step 8 Click **Deploy**.

The system validates the configuration. If validation errors occur, they are displayed for correction.

Step 9 Track the deployment progress and wait until the VM state changes to **ACTIVE** status.

After the VM is ACTIVE, click the **Terminal** button in the VM row. Monitor the installation progress via the console and follow application-specific installation prompts.

Table 4: Virtual machine deployment statuses

Status	Description	Action
DEPLOYING	VM is being created	Wait for completion
ACTIVE	VM is running	Access console, verify operation
ERROR	Deployment failed	Check logs, troubleshoot
SHUTDOWN	VM is powered off	Power on if needed

Note

The deployment steps for Cisco Unity Connection, and Cisco IM and Presence Service (IM&P) on NFVIS are the same as those for Cisco Unified Communications Manager (CUCM). These steps include:

a. Image Registration – Register the ISO and OVA metadata (same process as CUCM).

b. Virtual Machine (VM) Deployment – Deploy the VM with a Bootstrap ISO (same process as CUCM).

c. Virtual Machine (VM) Management – Perform power operations, console access, and network updates (same as CUCM).

For application-specific documentation, refer to the respective Cisco documentation for Cisco Unity Connection and Cisco IM and Presence Service.

The CUCM virtual machine is successfully deployed on NFVIS with the VM status showing as ACTIVE, and the Day0 configuration from the Bootstrap ISO is applied automatically during the installation process.

VM states

VM states represent the lifecycle phases of virtual machines or VNFs deployed on NFVIS. These states track the progression from initial deployment through operational status to termination.

VM States	Description
VM_UNDEF_STATE	The initial STATE of a VM or VNF before deployment of this VM.
VM_DEPLOYING_STATE	The VM or VNF is being deployed on to the NFVIS.
VM_MONITOR_UNSET_STATE	The VM or VNF is deployed in NFVIS but the monitoring rules are not applied.
VM_MONITOR_DISABLED_STATE	Due to a VM action request or recovery workflow, the monitoring or KPI rules applied to the VM were not enabled.
VM_STOPPING_STATE	VM or VNF is being stopped.
VM_SHUTOFF_STATE	VM or VNF is in stopped or SHUTOFF STATE.
VM_STARTING_STATE	VM or VNF is being started.
VM_REBOOTING_STAT	VM or VNF is being rebooted.
VM_INERT_STATE	VM or VNF is deployed but not ALIVE. The KPI MONITOR is applied and waiting for the VM to become ALIVE.
VM_ALIVE_STATE	VM or VNF is deployed and successfully booted up or ALIVE as shown in the KPI metric.
VM_UNDEPLOYING_STATE	The deployment of a VM or VNF is being terminated.
VM_ERROR_STATE	The VM or VNF is in an ERROR STATE because the deployment or some other operation has failed.

NFVIS container deployment

NFVIS container deployment is a network function virtualization process that

- follows the same workflow as VM deployment
- supports SRIOV/Host Interface, Day 0 Configuration, Configuration Options, vCPU Pinning, and Volumes, and
- excludes import and export operations for containers.

Container deployment restrictions and capabilities

NFVIS container deployment has the following restriction:

- Import and export of containers is not supported.

The container deployment process in container lifecycle management supports the following features:

- SRIOV/Host Interface
- Day 0 Configuration
- Configuration Options
- vCPU Pinning
- Volumes

SRIOV/Host interface configuration

To attach an SRIOV or host interface to the container's interface, specify the SRIOV or host interface in the configuration. Either physical interfaces or SRIOV VFs can be specified in the configuration.

To attach an SRIOV or host interface to the container's interface, use this configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<vm_lifecycle xmlns:ns0="http://www.cisco.com/nfvis/vm_lifecycle"
xmlns="http://www.cisco.com/nfvis/vm_lifecycle">
    <tenants>
        <tenant>
            <name>admin</name>
            <deployments>
                <deployment>
                    <name>ubuntu-2</name>
                    <vm_group>
                        ...
                        <interfaces>
                            <interface>
                                <nicid>0</nicid>
                                <network>int-LAN-vf-1</network>
                            </interface>
                        </interfaces>
                    </vm_group>
                </deployment>
            </deployments>
        </tenant>
    </tenants>
</vm_lifecycle>
```

Day 0 configuration

Day 0 support for container deployments follows the same process as that of VM deployments. Refer to VM Bootstrap Configuration Options with a VM Deployment.

Configuration options setup

There are multiple boot options available, that allow you to customize the container behavior.

To customize the container behavior using different configuration options, use this configuration:

```
<config_data>
  <config_type>CONFIG_DATA_OPTIONS</config_type> <!-- type is required or options below are
 ignored -->
  <config_options>
    <options>
      <option><name>ENV_VARIABLE</name><value>env=test</value></option>
      <option><name>ENV_VARIABLE</name><value>env2=test2</value></option>
    </options>
  </config_options>
  <configuration>
     <dst>/etc/ovf-env.xml</dst>
     . . .
```

Table 5: Supported configuration options

Option Name	Type	Description
ENV_VARIABLE	String	Environment VARIABLE to set within container: name/value pair (example: varName=varValue)
LOW_LATENCY	Boolean	vCPU pinning option: When the value is True, the container is allocated a dedicated core.

VNF deployment update

After you have deployed a VNF, you can update it in terms of its flavor, CPU topology, or interfaces.

Update VNF flavor

Update a VNF deployment to have a different flavor from the one you deployed it with. The flavor can also be custom-defined.

VNF flavor updates allow you to modify CPU and memory resources for your virtual network functions to meet changing requirements.

Note Before updating a VNF with another flavor, we recommend that you check whether CPUs are avaialble for the required update.

Updating a VNF flavor only supports CPU and Memory changes and does not support disk size change.

Before you begin

Follow these steps to update a VNF flavor:

Procedure

Step 1 Get a list of available flavors.

Example:

```
curl -k -v -u admin:admin -X GET
https://209.165.201.1/api/operational/vm_lifecycle/flavors?deep
```

Step 2 Check the CPU usage of the system.

Example:

```
curl --tlsv1.2 -k -i -u admin:Esc123# -H
Accept:application/vnd.yang.data+json -H content-
type:application/vnd.yang.data+json -X GET
https://<nfvis_ip>/api/operational/resources/cpu-info/allocation
```

Step 3 Update the flavor of the VNF.

Example:

```
curl -k -v -u admin:admin -H Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml
-X PUT
https://<nfvis_ip>/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/<deploymentID>
/vm_group/<VMGroupName>/flavor
--data
'<flavor><FlavorName></flavor>
```

The VNF flavor is updated with the new CPU and memory configuration.

Changing the flavor of a VNF from flavor from ASAv5 to ASAv10

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H Content-Type:application/vnd.yang.data+xml -X PUT
https://172.29.91.32/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ASAdep/vm_group/ASAvmgrp/
flavor --data '<flavor>ASAv10</flavor>
```

Update CPU topology

This task allows you to update a VM to use a custom-defined CPU topology that you created when creating a VM flavor.

Updating the CPU topology of a VNF involves updating a VM to a custom-defined topology. The process is similar to updating a VNF flavor—by replacing the name of the CPU topology. For more details, see VM profiles or flavors, on page 21.

Procedure

Step 1 Update the CPU topology using the curl command.

Example:

```
curl -k -v -u admin:admin -H
Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml
-X PUT
https://<nfvis_ip>/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/<deploymentID>/vm_group/
<VMGroupName>/flavor
--data
'<flavor><FlavorName_withCPUTopology></flavor>'
```

Example of updating ISRv to include a CPU topology

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H Content-
Type:application/vnd.yang.data+xml -X PUT
https://172.29.91.32/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ROUTER/vm_group/ROUTER/flavor

--data '<flavor>Isrv_CPUToplogy</flavor>'
```

Step 2 Verify the changed configuration using the virsh command.

Example:

```
support virsh dumpxml <uid>
```

The following is an example of the output you would see.

```
<vcpu placement='static'>3</vcpu>
  <cputune>
    <vcpupin vcpu='0' cpuset='11'/>
    <vcpupin vcpu='1' cpuset='10'/>
    <vcpupin vcpu='2' cpuset='9'/>
    <emulatorpin cpuset='21-23'/>
  </cputune>
  . . .
  <cpu mode='host-passthrough' check='none'>
    <topology sockets='1' cores='3' threads='1'/>
  </cpu>
```

The VNF is successfully updated with the new CPU topology configuration.

VNF interface updates

VNF interface updates are VM deployment operations that

- support adding an interface, deleting an interface, and moving a VNIC of a VM from one interface to another
- can be performed as hot updates when a VM is in ACTIVE STATE or cold updates when a VM is in VM_SHUTOFF_STATE, and
- depend on custom image properties set during image registration.

Hot and cold update methods

All the options related to updating interfaces can be done in two ways: hot or cold.

Hot Update: Hot update refers to an update operation that runs when a VM is in ACTIVE STATE. In such cases, the VM does not reboot during the update.

Cold Update: Cold update refers to an update operation that runs after a VM is put in a VM_SHUTOFF_STATE. In such cases, the VM reboots during the update.

Note The need to do a hot or cold update depends on the custom image properties set during image registration. Refer to the table below for various custom image properties related to hot and cold updates.

Custom Image Property	Value	Interface Update Description	Default Value
interface_hot_add	true	Hot add an interface to a VM	true
interface_hot_add	false	Cold add an interface to VM	
interface_hot_delete	true	Hot delete a VM interface	false
interface_hot_delete	false	Cold delete a VM interface	

Starting from ISRv 17.1, interface_hot_add and interface_hot_delete are set to true by default.

The VNF interface update prerequisites are:

- The VNF should support hot add or hot delete operations for the interface.
- The custom image properties for interface update should be set during image registration to allow values other than the default.
- The VM to be updated needs to be in one of the following states: VM_ALIVE_STATE, VM_ERROR_STATE or VM_SHUTOFF_STATE.

This table shows which hot interface update operations are supported for various interface types.

Interface Type	Hot Add	Hot Delete	Hot Update for Moving VNIC
VIRTIO	Yes	Yes	Yes
SRIOV	Yes	Yes	Yes
DPDK	Yes	Yes	Yes

Note NFVIS also supports moving VNICs from one interface to another. For example, you can move a VNIC from a VIRTIO interface to SRIOV, or from SRIOV to DPDK, and so on.

If the VNIC is updated to a different interface type like SRIOV or DPDK, the configuration of the VNIC will not be preserved.

Syslog is not generated in ISRv when an interface is updated from a DPDK enabled network to another DPDK enabled network.

Interface update of any SRIOV interface to remove from one interface and add the same to another or swap, in a single transaction fails. Use two separate requests for a successful update.

Update interfaces

This task enables you to perform various interface management operations on VM deployments, including adding new interfaces, removing existing interfaces, and moving VNICs between networks.

Interface updates are necessary when modifying VM deployment configurations to change network connectivity, add additional network connections, or optimize network topology.

Procedure

Step 1 Add a single interface to a VM deployment.

Example:

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H Content-Type:application/vnd.yang.data+xml -X PUT
https://<NfvisIpAddress>/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ASAdep/vm_group/ASAvmgrp/interfaces
 --data '
<interfaces>
<interface>
        <nicid>0</nicid>
        <network>int-mgmt-net</network>
    </interface>
    <interface>
        <nicid>newNIC</nicid>
        <network>networkName</network>
    </interface>
    </interfaces>'
```

Step 2 Add multiple interfaces to a VM deployment.

Example:

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X PUT
https://172.29.91.32/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ASAdep/vm_group/ASAvmgrp/interfaces
 --data '
<interfaces>
<interface>
          <nicid>0</nicid>
          <network>int-mgmt-net</network>
      </interface>
      <interface>
          <nicid>1</nicid>
          <network>wan-net</network>
      </interface>
      <interface>
      <nicid>2</nicid>
      <network>lan-net</network>
      </interface>
      </interfaces>'

HTTP/1.1 204 No Content
```

Step 3 Delete an interface from a VM deployment.

Remove the required nicID along with content between <interface> and </interface> tags.

Example:

```
curl -k -v -u admin:<password> -H
Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X PUT
https://<NfvisIpAddress>/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ASAdep/vm_group/ASAvmgrp/interfaces
 --data '
<interfaces>
<interface>
        <nicid>0</nicid>
        <network>int-mgmt-net</network>
    </interface>
**** Note: Remove the required nicID along with content between <interface> and </interface> ******

    </interfaces>'
```

Example:

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X PUT
https://172.29.91.32/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ASAdep/vm_group/ASAvmgrp/interfaces
 --data '
    <interfaces>
      <interface>
              <nicid>0</nicid>
              <network>int-mgmt-net</network>
         </interface>
         <interface>
             <nicid>1</nicid>
             <network>wan-net</network>
       </interface>
     </interfaces>'

HTTP/1.1 204 No Content
```

In this example, NIC ID 2 has been excluded from the REST API for it to be deleted from the deployment.

Step 4 Move VNICs from one network to another.

Example:

```
curl -k -v -u admin:<password> -H
Accept:application/vnd.yang.data+xml -H
Content-Type:application/vnd.yang.data+xml -X PUT
https://<NfvisIpAddress>/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ASAdep/vm_group/ASAvmgrp/interfaces
 --data '
<interfaces>
    <interface>
        <nicid>0</nicid>
        <network>int-mgmt-net</network>
    </interface>
    <interface>
        <nicid>selectedNicId</nicid>
         <network>NewNetworkSelected</network>
    </interface>
</interfaces>'
```

Example:

```
curl -k -v -u admin:Esc123# -H
Accept:application/vnd.yang.data+xml -H Content-Type:application/vnd.yang.data+xml -X PUT
https://172.29.91.32/api/config/vm_lifecycle/tenants/tenant/admin/deployments/deployment/ASAdep/vm_group/ASAvmgrp/interfaces
 --data '
<interfaces>
         <interface>
                 <nicid>0</nicid>
                 <network>int-mgmt-net</network>
```

```
            </interface>
            <interface>
                <nicid>1</nicid>
                <network>wan2-net</network>
          </interface>
</interfaces>'

HTTP/1.1 204 No Content
```

This example shows how to move nicid 1 from wan-net to wan2-net.

The interface configuration is updated on the VM deployment. You should receive an HTTP 204 No Content response indicating successful completion of the operation.

Access VNFs

In Cisco NFVIS, you can access VNFs in three ways after they have been deployed:

- Through VNC Console
- Through Serial Console.
- Through Port Forwarding if VM is deployed with port_forwarding configuration in deployment payload

Access VNFs using VNC console

Access VNFs through a VNC client to manipulate the VNF through the NFVIS portal.

A VNC console allows you to access VNFs through a VNC client. This method enables you to manipulate the VNF through the NFVIS portal.

Before you begin

Follow these steps to access the VNF using the VNC console:

Procedure

Step 1 Using the NFVIS portal, from the **Manage Deployments**, click **Configuration** > **Virtual Machine** > **Manage**

Step 2 Select the desired VM from the list.

The VM must be in active status.

Step 3 Click **Terminal** in the VM row.

A console window opens in your browser.

Step 4 For added security, enable VNC passphrase by including the following contents in your configuration.

Example:

```
<vm_lifecycle>
<tenants>
<name>admin</name>
<deployments>
<deployment>
…
<vnc>
```

```
<password>PASSWORD</password>
</vnc>
</deployment>
</deployments>
</tenant>
</tenants>
</vm_lifecycle>
```

Access to the VNC console can be restricted through a passphrase.

You can now access and manipulate VNFs through the VNC console in your browser.

Access VMs using serial console

The serial console allows you to access the VM using the serial interface provided by the VM itself.

This method is applicable only if the VM supports serial interfaces in both, its image and its image properties.

Before you begin

Follow these steps to access VMs using serial console:

Procedure

To access the VNF through the serial console, use the command `vmConsole` followed by the name of the VNF.

Example:

```
nfvis# show system deployments
NAME                                          ID  STATE
-----------------------------------------------------
sj-02-sj02-isrv.sj-02-sj02-isrv               1   running
sj-02-sj02-linux.sj-02-sj02-linux             2   running
sj-02-sj02-vwaas.sj-02-sj02-vwaas             3   running
sj-02-sj02-firewall.sj-02-sj02-firewall       4   running

nfvis#
nfvis#
nfvis# vmConsole sj-02-sj02-vwaas.sj-02-sj02-vwaas
Connected to domain sj-02-sj02-vwaas.sj-02-sj02-vwaas
Escape character is ^]

sj-02-sj02-vwaas#
sj-02-sj02-vwaas#
sj-02-sj02-vwaas#
telnet> send escape

nfvis#
```

When you access the VM console using the ENCS console port on the device, you cannot use ctrl+] to exit the VM console. You must use ctrl+] and then enter **`send escape`** to exit the VM console.

You have successfully accessed the VM through the serial console interface and can now interact with the VM directly.

Access a VM using port forwarding

This task enables you to access a virtual machine through port forwarding configuration, which allows secure remote access to VMs by mapping external ports to internal VM ports.

Port forwarding is used when you need to access a VM that is deployed behind a network address translation (NAT) or firewall. This method creates a mapping between external and internal ports to enable connectivity.

Procedure

Step 1 Deploy a VNF using the deployment payload with **port_forwarding** configuration:

Example:

```
<port_forwarding>
  <port>
    <type>ssh</type>
    <protocol>tcp</protocol>
    <vnf_port>22</vnf_port>
    <external_port_range>
      <start>20122</start>
      <end>20122</end>
    </external_port_range>
  </port>
  <port>
    <type>telnet</type>
    <protocol>tcp</protocol>
    <vnf_port>23</vnf_port>
    <external_port_range>
      <start>20123</start>
      <end>20123</end>
    </external_port_range>
  </port>
</port_forwarding>
```

Step 2 Log into VNF using SSH and port number given in the example payload (20122):

Example:

```
USER-M-G2PT:~ user$ ssh cisco@172.29.91.28 -p 20122
Password:

isrv-encs#
```

You have successfully accessed the VM using port forwarding. The SSH connection is established and you can now manage the virtual machine remotely through the forwarded port.

VM monitoring

VM monitoring is a system capability that

- monitors deployed VMs periodically based on metrics defined in the KPI section of deployment data model
- can be enabled or disabled by modifying the <actionType> tag, and

- uses bootup_time settings to control monitoring behavior.

VM monitoring configuration details

VM monitoring behavior is controlled through specific settings and parameters:

- If the bootup_time is set at -1, it signifies that VM monitoring is disabled.
- You are not required to set a boot up time during image registration. However, you must set it during VM deployment.
- If a qcow2 image is used during registration, the bootup_time defaults to -1.

Disabling VM monitoring

This example shows how to disable monitoring for a VM.

```
curl -k -v -u "admin:password" -H
"Accept:application/vnd.yang.data+xml" -H
"Content-Type:application/vnd.yang.data+xml" -X POST
https://<NFVIS_IP>/api/operations/vmAction --data '<vmAction>
<actionType>DISABLE_MONITOR</actionType><vmName><vm-instance name></vmName></vmAction>'
```

VM import and export

VM export with selective disk

You can exclude certain disks or volumes from a VM export.

```
nfvis# show running-config vm_lifecycle tenants tenant admin deployments
vm_lifecycle tenants tenant admin
 deployments deployment linuxdep
  vm_group gp_01
   image              Linux_IMAGE
   flavor             centos-disk-large
   vim_vm_name        linux_vm
   bootup_time        -1
   recovery_wait_time 0
   recovery_policy action_on_recovery REBOOT_ONLY
   interfaces interface 0
    model   virtio
    network wan-net
   !
   scaling min_active 1
   scaling max_active 1
   placement zone_host
    host datastore1
   !
   vmexport_policy disk_exclusion Linux_IMAGE:512G-file.qcow2 =====> Disk to be excluded
on Export
   !
  !
 !
!
```

NFVIS VM import and export

NFVIS VM import and export is a backup and recovery mechanism that

- allows you to backup or export (vmExportAction) VMs for disaster recovery
- enables you to restore or import (vmImportAction) VMs from backup files, and
- provides VM-level backup and recovery separate from full NFVIS system backup.

VM export and import limitations

VM export and import operations have specific requirements and limitations:

- The imported VM cannot change datastore.
- The original registered image must exist.
- The OVS network name must be identical to the one used by original deployment.
- VM export is dependant on the amount of free space available in the deployed datastore, regardless of the free space available in the destination datastore. For example, when the VM is deployed in the intdatastore (default), you should ensure that the available free space is at least twice that of the deployed VM.

To export a VM ensure that:

- Backup file must be saved to NFVIS datastore or USB.
- Provide a backup name for NFVIS to append .vmbkp extension to the backup name.

You can only create and save a VM backup to datastores. The backup file has .vmbkp extension. To verify the backup:

```
nfvis# show system file-list disk local | display xpath | include backup

/system/file-list/disk/local[si-no='84']/name tiny_backup.vmbkp
nfvis# show system file-list disk local 84
SI NO  NAME              PATH               SIZE  TYPE              DATE MODIFIED
-------------------------------------------------------------------------------------
84     tiny_backup.vmbkp  /mnt/extdatastore1  17M   VM Backup Package  2019-01-31 19:31:32
```

To import a VM ensure that:

- The Backup file is placed under NFVIS datastores or USB.
- The registered image used by the original deployed VM is in the same datastore, with same properties.
- The exported VM does not exist on the system.
- OVS network used by the original deployment should exist.
- Restored VM is created with the same datastore with same deployment properties.
- The full path name to backup file is used (for example, /mnt/extdatastore1/backup.vmbkp, not extdatastore1:backup)

```
nfvis# vmImportAction importPath /mnt/extdatastore1/tiny_backup.vmbkp
System message at 2019-01-31 19:53:32...
```

```
Commit performed by admin via ssh using maapi.
```

An optional unique MAC UID support is added to VM import.

```
vmImportAction importPath <vm backup file with location> uniqueMacUid
```

Specifying the uniqueMACUid flag ensures that the imported VM is not deployed with the same UID and interface MAC addresses.

VM backup using vmExportAction:

Export and import failures

These examples show export failures:

- Original deployment is not deleted

```
nfvis# vmImportAction importPath /mnt/extdatastore1/tiny_backup.vmbkp
Error: Exception from action callback: Deployment Configuration :
'SystemAdminTenantIdtiny' already exists , can not be imported/restored due to conflict!
```

- OVS network used by original deployment is deleted.

```
nfvis# vmImportAction importPath /mnt/extdatastore1/tiny_backup.vmbkp
Error: Exception from action callback: Restoration Request rejected, see logs for root
 cause
```

Table 6: Feature comparison table for VM backup using vmExportAction

Features	NFVIS 4.18.2a and Later Releases
Default file location for backup vmExportAction vmName sample exportName vmbackup exportPath <datastore>:	/data/intdatastore/vmbackup.vmbkp /mnt/extdatastore1/vmbackup.vmbkp /mnt/extdatastore2/vmbackup.vmbkp
Default file location for backup on USB vmExportAction vmName sample exportName backup02 exportPath usb:usb1	/mnt-USB/usb1/vmbackup.bkup
Check disk space before backup	Supported
VM backup format	Diff disk backup
Backup image and flavor	Supported
VM live export snapshot	Supported
VM Export with Selective Disk	Supported

VM restore using vmImportAction:

Table 7: Feature comparison table for VM restore using vmImportAction

Features	NFVIS 4.18.2a and Later Releases
Default file location for backup vmImportAction importPath <datastore>:vmbackup.vmbkp	/data/intdatastore/vmbackup.vmbkp /mnt/extdatastore1/vmbackup.vmbkp /mnt/extdatastore2/vmbackup.vmbkp
Default file location for restore on USB vmImportAction importPath usb:usb1/vmbackup.vmbkp	/mnt-USB/usb1/vmbackup.vmbkp
Check disk space before backup	Supported
Restore backing images and flavors	Supported
VM Export with Selective Disk	Supported

Additional capabilities

Recover a Cisco CUCM application using a recovery ISO

This task allows you to recover or repair a Cisco CUCM application when the virtual machine requires restoration from a corrupted or damaged state.

Use this procedure when a Cisco CUCM application needs recovery or repair. The process involves using recovery ISO media to restore the application to a functional state through the NFVIS portal interface.

Before you begin

Ensure you have access to the NFVIS portal and the appropriate recovery ISO file for your CUCM application version.

Follow these steps to recover a Cisco CUCM application using a recovery ISO:

Procedure

Step 1 From the NFVIS portal, click **Configuration** > **Virtual Machine** > **Images** > **Image Repository**.

Step 2 Upload the Recovery ISO.

If the Recovery ISO registration is not allowed, upload the respective VM OVA as well and register it.

Name the image descriptively. Example: CUCM-15-recovery-ISO.

Step 3 Shut down the VM.

a. Click **Configuration** > **Virtual Machine** > **Manage**.

b. Select the VM.

c. Click **Switch Power** to shut down the VM.

   Wait for the VM state to move to a **SHUTDOWN** state.

Step 4 Attach the Recovery ISO to the CD-ROM.

a. With the VM in the **SHUTDOWN** state, click the **CD-ROM** button.

b. Click **ATTACH**.

c. Select the Recovery ISO from the drop-down list.

d. Click **Submit**.

   The Recovery ISO is now attached to the VM.

Step 5 Power on and boot from the Recovery ISO.

a. Click **Switch Power** to start the VM.

b. Click **Terminal** to access the console.

c. Enter the Boot Menu by pressing the appropriate key during startup (typically F12 or ESC).

d. Select the Recovery ISO device from the boot menu.

   The VM boots from the Recovery ISO.

e. Follow the application-specific recovery procedure.

   This step involves interacting with the application's recovery prompts via the console.

Step 6 Detach the CD-ROM after recovery.

a. Shut down the VM.

b. Click the **CD-ROM** button.

c. Select the **DETACH** action.

d. Select the Recovery ISO disk.

e. Click **Submit**.

Step 7 Power on the VM for normal operation.

The Cisco CUCM application is recovered and restored to a functional state. The VM boots normally and the application is ready for use.

Recover the password for a Cisco CUCM application

Use this procedure to reset the password for a Cisco CUCM application while the VM is in an active state.

Cisco NFVIS allows you to eject and insert ISO media, which is a required step in the password recovery process for CUCM applications. This procedure covers only the NFVIS media operations. For the complete application-specific password recovery procedure, refer to the Reset or Change CUCM OS Admin and Security Password guide in the official CUCM documentation.

Before you begin

Follow these steps to recover the password for a Cisco CUCM application:

Procedure

Step 1 From the NFVIS portal, choose **Configuration** > **Virtual Machine** > **Manage**.

Step 2 Verify the VM state.

Ensure the VM is running and in an ACTIVE state as password recovery is performed on a running VM.

Step 3 Eject the current ISO.

a. Select the VM.

b. Click the **CD-ROM** button.

c. Select **Eject**.

d. Select the ISO image to eject.

e. Click **Submit**.

Step 4 Insert the required ISO.

a. Click the **CD-ROM** button.

b. Select **Insert**.

c. Select the required ISO from the drop-down list.

d. Select the appropriate Device ID for the CD-ROM.

e. Click **Submit**.

Step 5 Complete the password recovery.

a. Access the VM console by clicking the **Terminal** button.

b. Follow the application-specific password recovery steps provided in the CUCM documentation.

The ISO media is successfully swapped, and the application-specific password recovery process is initiated.

What to do next

Refer to the official CUCM documentation for any final application-specific configuration steps required after the password reset.

CDROM attachment and detachment

CDROM attachment and detachment

CDROM attachment and detachment is a VM management capability that

- allows you to connect or disconnect ISO images to a VM
- supports various use cases, including system recovery, operating system installation, and software deployment, and
- enables the VM to access content from a virtual CDROM drive.

CDROM attachment and detachment process

Attaching a CDROM to a VM is a two-step process:

1. Image Registration: First, you must register the desired ISO image using the existing image registration workflows provided by NFVIS. This makes the ISO available for selection. For information about ISO image registration, see Image Registration.
2. CDROM Action: After registration, you can proceed to attach the registered ISO to the target VM by utilizing the dedicated CDROM actions available in the **Manage Deployments** page.

   To detach an ISO image, you would use the corresponding detachment action in the **Manage Deployments** page. Both attachment and detachment operations require the VM to be in a shutdown state.

Requirement: prerequisites for CDROM attachment and detachment

To successfully perform CDROM attachment or detachment operations, the VM must be in a shutdown state. These operations cannot be initiated or completed on an active VM.

Manage CDROM attachment and detachment

Attach or detach ISO images to and from virtual machines for recovery and maintenance operations.

CDROM attachment and detachment operations allow you to mount ISO images to virtual machines for recovery purposes or maintenance tasks. These operations can only be performed when the VM is in a shutdown state.

Before you begin

- Ensure the ISO image you intend to attach is registered using the image registration workflows. For more information, see Image Registration.
- Verify that the VM is in a shutdown state. CDROM attach and detach operations cannot be performed on an active VM.

Follow these steps to manage CDROM attachment and detachment:

Procedure

Step 1 From the **Manage Deployments** page, click the **CD-ROM** icon.

Step 2 Choose the operation you want to perform.

- To attach an ISO image, click **Attach**.
- To detach an ISO image, click **Detach**.

Step 3 Select the appropriate image or disk.

Table 8:

If...	Then...
You are attaching a CDROM	From the **Images** drop-down list, choose the desired recovery ISO from the list of registered ISO images.
You are detaching a CDROM	From the **Disk Name** drop-down list, choose the disk name corresponding to the recovery ISO you wish to detach.

Step 4 Click **Submit**.

A confirmation message appears indicating the CDROM operation is complete.

After attaching a CDROM, to use the attached recovery ISO, you must start the VM and access its boot menu to select the ISO disk for booting. After detaching a CDROM, you can start the VM for normal operation. To confirm detachment, click **CD-ROM** > **Detach**. The ISO image should no longer be listed in the **Disk Name** drop-down list.

VM graceful stop

Graceful VM stop

A graceful stop is a VM power management option that

- sends an ACPI signal to the operating system within the VM
- allows services and applications to terminate in an orderly manner before the VM powers off, and
- contrasts with forceful shutdown, which immediately cuts power to the VM.

Key features and behavior

- Default Behavior: When initiating a VM shutdown, the **Graceful stop** checkbox is selected by default for VMs deployed with type as **OTHER**.
- Timeout: The graceful stop operation has a timeout period of 20 minutes.

- Error State: If the VM does not shut down gracefully within the 20-minute timeout, it will transition to an error state.
- Forceful Fallback: If a graceful stop fails or the VM enters an error state, you can uncheck the **Graceful stop** option to perform a forceful shutdown.

Perform a graceful stop

Perform a graceful stop to safely shut down a VM while allowing running processes to complete and prevent data loss.

Use a graceful stop when you need to shut down a VM while ensuring that all running processes complete properly and the system shuts down cleanly. This method is the recommended approach for routine VM shutdowns.

Procedure

Step 1 From the **Manage Deployments** page, click the **Switch Power** icon for the VM that you want to stop or shutdown.

The **Graceful stop** checkbox is selected by default for VMs deployed with type as **OTHER**.

Step 2 Click **OK** to confirm VM shutdown.

The graceful shutdown process begins in the background.

Step 3 Monitor the VM status in the management interface.

If the VM does not shut down within 20 minutes, it will automatically move to an error state. In this case, you may need to perform a forceful shutdown by repeating this procedure and unchecking the **Graceful stop** option.

The VM status transitions from **active** to **shutdown**, indicating the graceful stop was successful.