# VMware Telco Cloud Automation User Guide

VMware Telco Cloud Automation 1.8
VMware Telco Cloud Manager 1.8.0
VMware Telco Cloud Automation Control Plane 1.8.0

You can find the most up-to-date technical documentation on the VMware website at:

https://docs.vmware.com/

**VMware, Inc.**
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

# Contents

# Introduction

1

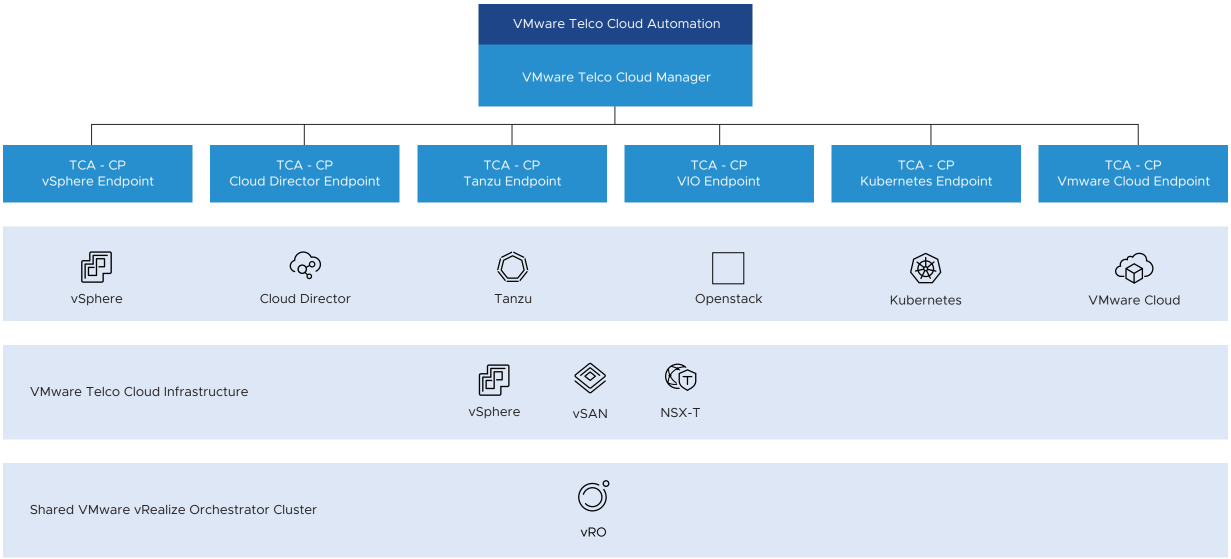
The *VMware Telco Cloud Automation User Guide* provides information about how to use VMware Telco Cloud Automation™. Steps to add your virtual infrastructure and to create and manage network functions and services are covered in this guide.

VMware Telco Cloud Automation is a cloud orchestration solution that accelerates the time-to-market of modern network functions and services. It provides a simplified lifecycle management automation solution, across any network and any cloud. Some of the features of VMware Telco Cloud Automation are:

- A native integration for VIMs and cloud products such as VMware vCloud NFV, vSphere-based clouds, VMware on mega-cloud providers, and Kubernetes clouds. These integrations streamline your CSP orchestrations and optimize your NFV Infrastructure (NFVI) resource use.
- A standard-driven generic VNF manager (G-VNFM) and NFV Orchestration (NFVO) modular components to integrate any multi-vendor Management and Network Orchestration (MANO) architecture.

VMware Telco Cloud Automation consists of two components:

- VMware Telco Cloud Manager™ - Provides Telcos with NFV-MANO capabilities and enables the automation of deployment and configuration of Network Functions and Network Services.
- VMware Telco Cloud Automation Control Plane (TCA-CP) - Provides the infrastructure for placing workloads across clouds using VMware Telco Cloud Automation.


VMware Telco Cloud Automation
VMware Telco Cloud Manager
TCA - CP vSphere Endpoint
TCA - CP Cloud Director Endpoint
TCA - CP Tanzu Endpoint
TCA - CP VIO Endpoint
TCA - CP Kubernetes Endpoint
TCA - CP Vmware Cloud Endpoint
vSphere
Cloud Director
Tanzu
Openstack
Kubernetes
VMware Cloud
VMware Telco Cloud Infrastructure
vSphere
vSAN
NSX-T
Shared VMware vRealize Orchestrator Cluster
vRO

This chapter includes the following topics:

- Common Abbreviations
- API Documentation
- Deployment Architecture
- Supported Features on Different VIM Types

# Common Abbreviations

Some of the frequently used abbreviations that are used in this guide are listed here with their descriptions.

## NFV

Network Functions Virtualization - The process of decoupling a network function from its proprietary hardware appliance and running it as a software application in a virtual machine.

## VNF

Virtual Network Function - Is a part of the NFV architecture that handles specific network functions running on one or more virtual machines.

## Network Service

Individual VNFs can be combined to create a full-scale networking communication service.

## CNF

Cloud-Native Network Function - A CNF is a containerized network function that uses cloud-native principles. CNFs, when running inside telecommunications premises, create a private cloud where the same public cloud principles are used effectively.

## NFVI

Network Functions Virtualization Infrastructure - Is the foundation of the overall NFV architecture. It provides the physical compute, storage, and networking hardware that hosts the VNFs. Each NFVI block can be thought of as an NFVI node and many nodes can be deployed and controlled geographically.

## MANO

Management and Orchestration - Manages the resources in the infrastructure and the orchestration and life cycle of VNFs.

## VIM

Virtualized Infrastructure Manager - Is a functional block of the MANO and is responsible for controlling, managing, and monitoring the NFVI compute, storage, and network hardware, the software for the virtualization layer, and the virtualized resources. The VIM manages the allocation and release of virtual resources, and the association of virtual to physical resources, including the optimization of resources.

## NFVO

NFV Orchestrator - Is a central component of an NFV-based solution. It brings together different functions to make a single orchestration service that encompasses the whole framework and has a well-organized resource use.

## VNFM

VNF Manager - Works with the VIM and NFVO to help standardize the functions of virtual networking and increase the interoperability of software-defined networking elements.

**Note** VNFM works with both VNFs and CNFs.

## NFD

Network Function Descriptor - Is a deployment template that describes a network function deployment and operational requirement. It is used to create a network function where life-cycle management operations are performed.

## Network Function Catalog

Is a functional building block within a network infrastructure. It has well-defined external interfaces and a well-defined functional behavior.

## Network Services Catalog

A Network Services (NS) Catalog is a list of all usable network resources. You can store the deployment templates for a network service here.

### SVNFM

Specific VNFM. SVNFMs are tightly coupled with the VNFs they manage.

### GVNFM

Generic VNFM.

### Kubernetes Pods

Kubernetes Pods are inspired by pods found in nature (pea pods or whale pods). The Pods are groups of containers that share networking and storage resources from the same node. They are created with an API server and placed by a controller. Each Pod is assigned an IP address, and all the containers in the Pod share storage, IP address, and port space (network namespace).

### CSI

Container Storage Interface. A specification designed to enable persistent storage volume management on Container Orchestrators (COs) such as Kubernetes. The specification allows storage systems to integrate with containerized workloads running on Kubernetes. Using CSI, storage providers, such as VMware, can write and deploy plug-ins for storage systems in Kubernetes without a need to modify any core Kubernetes code.

### CNI

Container Network Interface. The CNI connects Pods across nodes, acting as an interface between a network namespace and a network plug-in or a network provider and a Kubernetes network.

### TCA-CP

VMware Telco Cloud Automation Control Plane. Previously known as VMware HCX for Telco Cloud.

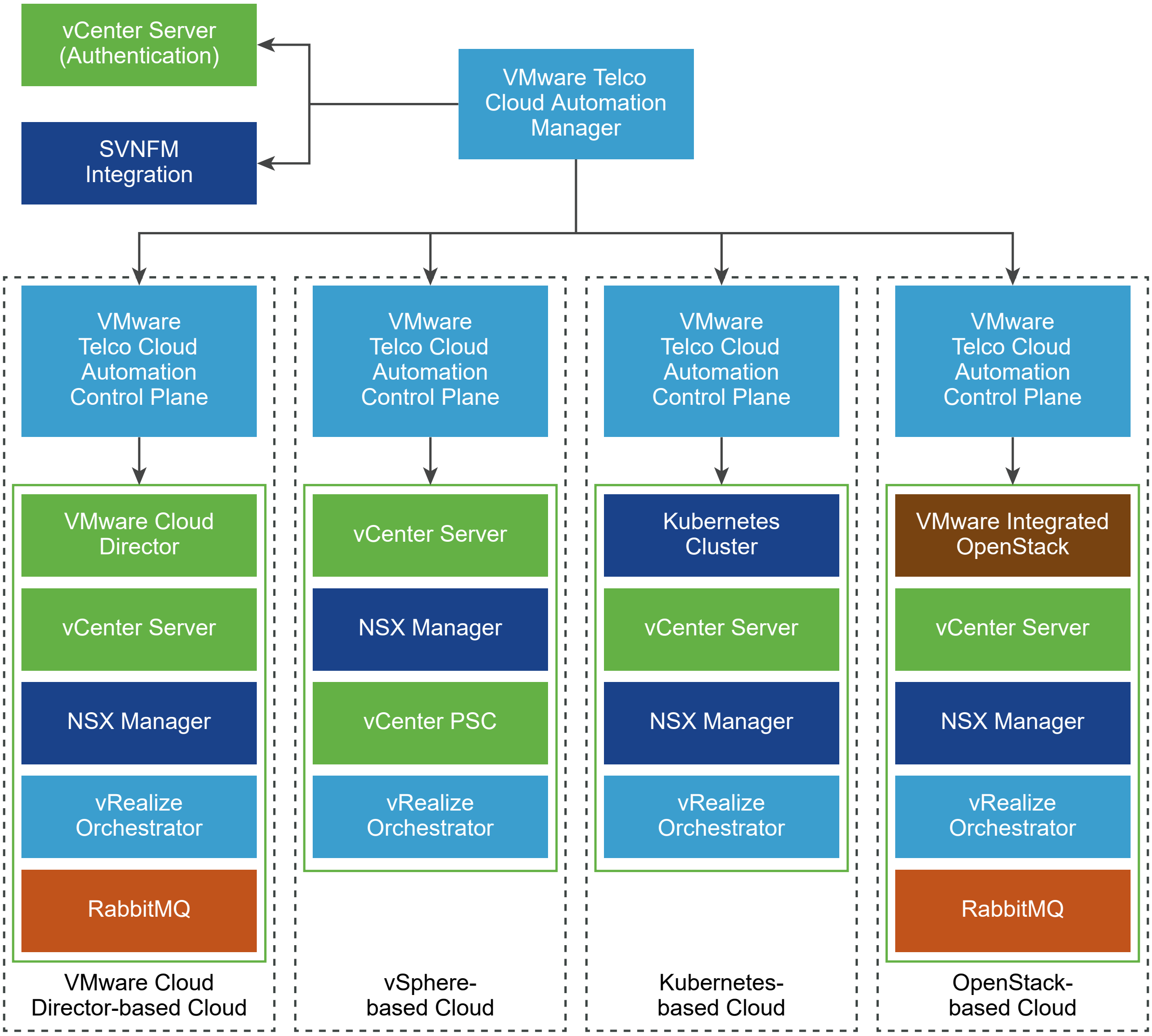
## API Documentation

You can also operate VMware Telco Cloud Automation using APIs.

To view the VMware Telco Cloud Automation API Explorer, Click the **Help** (**?**) icon from the top-right corner of the VMware Telco Cloud Automation user interface and select **API Documentation**.

## Deployment Architecture

The VMware Telco Cloud Automation implements the architecture that is outlined and defined at a high-level through logical building blocks and core components.



- vCenter Server is used for authenticating and signing in to VMware Telco Cloud Automation.
- Any SOL 003 SVNFM can be registered with VMware Telco Cloud Automation.
- VMware Telco Cloud Automation Control Plane (TCA-CP) is deployed on the VIM and paired with VMware Telco Cloud Automation Manager.
- VMware Telco Cloud Automation Manager connects with TCA-CP to communicate with the VIMs. The VIMs are cloud platforms such as vCloud Director, vSphere, Kubernetes Cluster, or VMware Integrated OpenStack.
- vRealize Orchestrator is registered with TCA-CP and is used to run NFV workflows. You can register for each VIM or for the entire network of VIMs. For information about registering vRealize Orchestrator with TCA-CP, see *VMware Telco Cloud Automation Deployment Guide*.
- RabbitMQ is used to track VMware Cloud Director and VMware Integrated OpenStack notifications and is required only for deployments on these clouds.

## Supported Features on Different VIM Types

The following table lists the feature sets that are supported on different VIM types.

Table 1-1. Supported Features on Different VIM Types

| | | VMware Telco Cloud Automation | | | |
|---|---|---|---|---|---|
| **Product** | **Versions** | **Infrastructure Automation** | **CaaS Automation** | **Generic VNF Manager** | **NFV Orchestrator** |
| vSphere | 6.7, 7.0 | | ✓ | ✓ | ✓ |
| | 7.0 U1 | ✓ | ✓ | ✓ | ✓ |
| VMware Cloud Director | 9.7.0.3, 10.1.2, 10.2 | | | ✓ | ✓ |
| vRealize Orchestrator | 7.4, 7.5, 7.6, 8.0, 8.1 | | ✓ | ✓ | ✓ |
| | 8.2 | ✓ | ✓ | ✓ | ✓ |
| NSX-V | 6.4.6 | | | ✓ | ✓ |
| NSX-T | 3.0, 3.1 | | ✓ | ✓ | ✓ |
| | 3.0.2 | ✓ | ✓ | ✓ | ✓ |
| VMware Tanzu Kubernetes Grid | 1.2 | ✓ | ✓ | ✓ | ✓ |
| Kubernetes | 1.18.6*, 1.17.9, 1.17.11, 1.18.8, 1.19.1 | | ✓ | ✓ | ✓ |
| VMware Integrated OpenStack | 6.0, 7.0 | | | ✓ | ✓ |
| vRealize Log Insight | 8.2 | ✓ | | | |

**Note** You cannot create a cluster with Kubernetes version 1.18.6 on VMware Telco Cloud Automation version 1.8. However, it is fully supported for managing the cluster. These clusters are created in VMware Telco Cloud Automation version 1.7.

# Getting Started

2

Complete these high-level tasks to start using VMware Telco Cloud Automation.

1. Install and set up:

   - VMware Telco Cloud Automation Control Plane (TCA-CP)
   - VMware Telco Cloud Automation

   For steps to install and set up these components, see the *VMware Telco Cloud Automation Deployment Guide*.

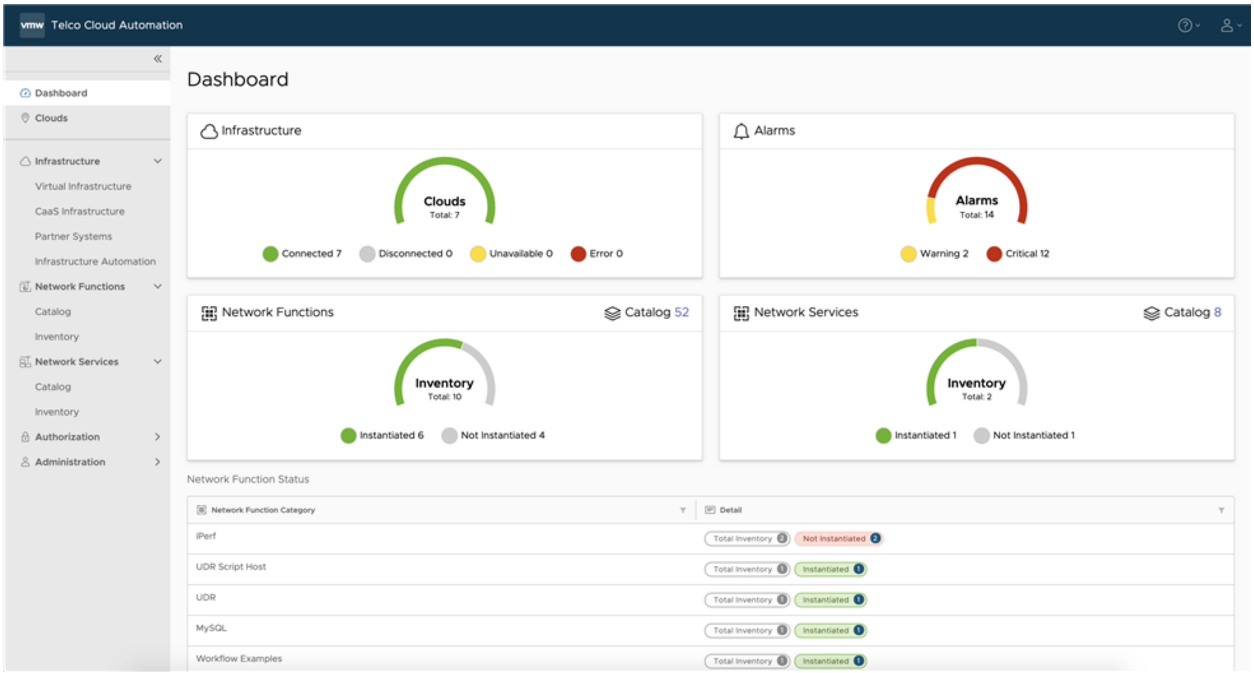
2. Create roles and assign permissions. See Chapter 3 Managing Roles and Permissions.
3. Configure your VIMs. See Chapter 5 Configuring Your Virtual Infrastructure.

This chapter includes the following topics:

- Viewing the Dashboard

## Viewing the Dashboard

The **Dashboard** is the first page that is displayed when you log in to VMware Telco Cloud Automation.



Network Function Status

| Network Function Category | Detail |
|---|---|
| iPerf | Total Inventory 2 Not Instantiated 2 |
| UDR Script Host | Total Inventory 1 Instantiated 1 |
| UDR | Total Inventory 1 Instantiated 1 |
| MySQL | Total Inventory 1 Instantiated 1 |
| Workflow Examples | Total Inventory 1 Instantiated 1 |

The following tiles are displayed:

### Clouds

Displays the number of clouds in your network and their status.

### Alarms

Displays alarms that are in the **Critical** and **Warning** states.

### Network Functions

Displays the number of instantiated and not instantiated network functions and catalogs. To go to the Network Function Catalog page, click the **Catalog** icon.

### Network Services

Displays the number of instantiated and not instantiated network services. To go to the Network Service Catalog page, click the **Catalog** icon.

### Network Function Status

Displays a detailed inventory view of the network functions.

### Total Resource Allocation Across Clouds

Displays the percentage of CPU, memory, and storage allocated across the clouds.

### Resource Utilization

Displays the percentage of CPU, memory, and storage resources used across the clouds.

# Managing Roles and Permissions 3

A role is a predefined set of privileges. Privileges define the rights to perform actions and read properties. For example, the **Virtual Infrastructure Administrator** role allows a user to read, add, edit, and delete VIMs. This role also allows the user to perform all the life-cycle management operations on a Kubernetes cluster template and a Kubernetes cluster instance.

As a vCenter Server user, when you configure vCenter Server in the VMware Telco Cloud Automation appliance, you are assigned the **System Administrator** role to access VMware Telco Cloud Automation. Use this role to create roles and permissions for your users.

A **System Administrator** or a **Role Administrator** of VMware Telco Cloud Automation manages the roles and permissions of users.

This chapter includes the following topics:

## Enabling Users and User Groups to Access VMware Telco Cloud Automation

VMware Telco Cloud Automation uses the vCenter Server authentication and authorization. Users and user groups defined in vCenter Server or its identity provider (IDP) can sign in to VMware Telco Cloud Automation.

To enable a specific vCenter Server user or a user group to access and use VMware Telco Cloud Automation, you must perform the following steps:

1 Log in to VMware Telco Cloud Automation with **System Administrator** credentials.

2 From the left navigation pane, click **Authorization** > **Permissions**.

3 Assign the appropriate Roles to the user or user group. A Role determines the privileges that the user or user group receives for accessing VMware Telco Cloud Automation.

4 To restrict access for your user or user group to specific objects, you can define the restrictions in the **Advance Filter** criteria.

5 Save the permissions.

Users or user groups with the assigned Role can access and use VMware Telco Automation, and perform tasks according to the specified permissions.

# Object Level Access Permissions

You can assign permissions at the object level and associate them to a specific Role.

As a System Administrator, you can restrict a user to access only specific objects. For example, you can assign permissions to VNF Administrators to access only specific VNFs. The **Advance Filter** option allows you to provide object-level permissions to roles.

## About Advance Filters

- If a user or a user group has multiple permissions, the list of objects that they can access is a union of all the objects that can be viewed through each permission.
- Filters that are applied to objects at the parent level are also applied to child objects. For example, you create permissions for your VNF Administrator with filters to view the VNF Catalogs of Nokia. When the VNF Administrator logs in, they can view the VNF Catalogs and the VNFs that belong to Nokia. Here, the parent object is the VNF Catalog and the child object is the VNF.

You can enable **Advance Filter** and assign object-level permissions when you create or edit permissions. For steps to create permissions, see Create Permission.

# Privileges and Roles

To perform specific operations, you require privileges associated with the specific role. VMware Telco Cloud Automation includes a set of system-defined roles and associated privileges. You cannot edit or delete them.

## Privileges and Roles

The following tables list the system-defined privileges and roles:

Table 3-1. System-Defined Privileges

| Privilege | Included Privileges | Accessible Objects |
|---|---|---|
| **System Admin**<br>Administrative privileges for all operations. | ■ **Role Admin**<br>■ **System Audit**<br>■ **Virtual Infrastructure Audit**<br>■ **Virtual Infrastructure Admin**<br>■ **Virtual Infrastructure Consume**<br>■ **Infrastructure Design**<br>■ **Infrastructure Lifecycle Management**<br>■ **Network Function Catalog Design**<br>■ **Network Function Catalog Read**<br>■ **Network Function Catalog Instantiate**<br>■ **Network Function Instance Read**<br>■ **Network Function Instance Lifecycle Management**<br>■ **Network Service Catalog Design**<br>■ **Network Service Catalog Read**<br>■ **Network Service Catalog Instantiate**<br>■ **Network Service Instance Read**<br>■ **Network Service Instance Lifecycle Management**<br>■ **Partner System Read**<br>■ **Partner System Admin**<br>■ **Role Audit** | ■ **Network Function Catalog**<br>■ **Network Service Catalog**<br>■ **Virtual Infrastructure**<br>■ **Network Function Instance**<br>■ **Network Service Instance**<br>■ **Kubernetes Cluster Template**<br>■ **Kubernetes Cluster Instance** |
| **Virtual Infrastructure Admin**<br>Administrative privileges for VIM. | **Virtual Infrastructure Audit** | ■ **Virtual Infrastructure**<br>■ **Kubernetes Cluster Template**<br>■ **Kubernetes Cluster Instance** |
| **Virtual Infrastructure Audit**<br>Read privileges for VIM. | | ■ **Virtual Infrastructure**<br>■ **Kubernetes Cluster Template**<br>■ **Kubernetes Cluster Instance** |
| **Infrastructure Design**<br>Design privileges for the CaaS cluster templates. | | **Kubernetes Cluster Template** |
| **Infrastructure Lifecycle Management**<br>Lifecycle management privileges for the CaaS cluster instances. The included privileges are **Virtual Infrastructure Consume** and **Infrastructure Design**. | | ■ **Kubernetes Cluster Template**<br>■ **Kubernetes Cluster Instance** |
| **Partner System Admin**<br>Administrative privileges for partner systems. | ■ **Partner System Read**<br>■ **Network Function Catalog Read**<br>■ **Virtual Infrastructure Audit** | **Partner System** |

Table 3-1. System-Defined Privileges (continued)

| Privilege | Included Privileges | Accessible Objects |
|---|---|---|
| **Virtual Infrastructure Consume**<br>Deploy privileges for VIM. | **Virtual Infrastructure Audit** | **Virtual Infrastructure** |
| **Network Function Catalog Design**<br>Design privileges for Network Function Catalog. | **Network Function Catalog Read** | **Network Function Catalog** |
| **Network Function Catalog Read**<br>Read privileges for Network Function Catalog. | | **Network Function Catalog** |
| **Network Function Catalog Instantiate**<br>Instantiation privileges for Network Function Catalog | - **Network Function Catalog Read**<br>- **Virtual Infrastructure Consume**<br>- **Network Function Instance Read** | **Network Function Catalog** |
| **Network Function Instance Read**<br>Read privileges for Network Function Instance. | | - **Network Function Instance**<br>- **Network Function Catalog** |
| **Network Function Instance Lifecycle Management**<br>Lifecycle management privileges for Network Function Instance. | - **Network Function Instance Read**<br>- **Network Function Catalog Instantiate**<br>- **Network Function Catalog Read**<br>- **Virtual Infrastructure Consume** | **Network Function Instance** |
| **Network Service Catalog Design**<br>Design privileges for Network Service Catalog. | - **Network Service Catalog Read**<br>- **Network Function Catalog Read** | **Network Service Catalog** |
| **Network Service Catalog Read**<br>Read privileges for Network Service Catalog. | **Network Function Catalog Read** | **Network Service Catalog** |
| **Network Service Catalog Instantiate**<br>Instantiation privileges for Network Service Catalog. | - **Network Service Catalog Read**<br>- **Virtual Infrastructure Consume**<br>- **Network Function Instance Read**<br>- **Network Service Instance Read** | **Network Service Catalog** |
| **Network Service Instance Read**<br>Read privileges for Network Service Instance. | | - **Network Service Instance**<br>- **Network Service Catalog** |
| **Network Service Instance Lifecycle Management**<br>Lifecycle Management privileges for Network Service Instance. | - **Network Service Instance Read**<br>- **Network Service Catalog Instantiate**<br>- **Network Function Catalog Read**<br>- **Network Function Instance Read**<br>- **Virtual Infrastructure Consume**<br>- **Network Function Catalog Read**<br>- **Network Function Catalog Instantiate** | **Network Service Instance** |

Table 3-1. System-Defined Privileges (continued)

| Privilege | Included Privileges | Accessible Objects |
|---|---|---|
| **System Audit**<br>Read privileges for all operations. | ■ **Virtual Infrastructure Audit**<br>■ **Partner System Read**<br>■ **Network Service Instance Read**<br>■ **Network Service Catalog Read**<br>■ **Network Function Instance Read**<br>■ **Network Function Catalog Read**<br>■ **Role Audit** | ■ **Network Function Instance**<br>■ **Network Service Instance**<br>■ **Virtual Infrastructure**<br>■ **Network Function Catalog**<br>■ **Network Service Catalog** |
| **Role Admin**<br>Administration privileges for all roles operations. | **Role Audit** | |
| **Role Audit**<br>Read privileges for all Role operations. | | |

Table 3-2. System Defined Roles

| Role | Privileges |
|---|---|
| **System Administrator**<br>The users assigned to this role can perform all the available actions in VMware Telco Cloud Automation. | ■ **Role Admin**<br>■ **System Audit**<br>■ **Virtual Infrastructure Audit**<br>■ **Virtual Infrastructure Admin**<br>■ **Virtual Infrastructure Consume**<br>■ **Infrastructure Design**<br>■ **Infrastructure Lifecycle Management**<br>■ **Network Function Catalog Design**<br>■ **Network Function Catalog Read**<br>■ **Network Function Catalog Instantiate**<br>■ **Network Function Instance Read**<br>■ **Network Function Instance Lifecycle Management**<br>■ **Network Service Catalog Design**<br>■ **Network Service Catalog Read**<br>■ **Network Service Catalog Instantiate**<br>■ **Network Service Instance Read**<br>■ **Network Service Instance Lifecycle Management**<br>■ **Partner System Read**<br>■ **Partner System Admin**<br>■ **Role Audit** |
| **Network Function Designer**<br>The users assigned to this role can perform all the network function actions such as designing, uploading, and managing the Network Function Catalogs. | ■ **Network Function Catalog Read**<br>■ **Network Function Instance Read** |

## Table 3-2. System Defined Roles (continued)

| Role | Privileges |
|---|---|
| **Network Function Deployer**<br>The users assigned to this role can perform all the network function actions related to the life-cycle management operations such as Instantiate, Scale, Heal, and other actions available on a Network Function instance. | ■ **Network Function Instance Read**<br>■ **Network Function Catalog Instantiate**<br>■ **Network Function Catalog Read**<br>■ **Virtual Infrastructure Consume**<br>■<br>■ **Virtual Infrastructure Audit**<br>■ **Network Function Instance Lifecycle Management** |
| **Virtual Infrastructure Administrator**<br>The users assigned to this role can perform all the virtual infrastructure-related actions in VMware Telco Cloud Automation. | ■ **Virtual Infrastructure Audit**<br>■ **Virtual Infrastructure Admin**<br>■ **Virtual Infrastructure Consume**<br>■ **Infrastructure Design**<br>■ **Infrastructure Lifecycle Management** |
| **Virtual Infrastructure Auditor**<br>The users assigned to this role can view all the virtual infrastructure entities in VMware Telco Cloud Automation. | **Virtual Infrastructure Audit** |
| **Network Service Designer**<br>The users assigned to this role can perform all the network service actions such as designing, uploading, and managing the Network Service Catalogs. | ■ **Network Service Catalog Design**<br>■ **Network Service Catalog Read**<br>■ **Network Function Catalog Read** |
| **Network Service Deployer**<br>The users assigned to this role can perform all the network service actions related to the life-cycle management operations such as Instantiate, Scale, Heal, and other actions available on a Network Service instance. | ■ **Network Service Instance Read**<br>■ **Network Service Catalog Instantiate**<br>■ **Network Service Catalog Read**<br>■ **Network Function Instance Read**<br>■ **Virtual Infrastructure Consume**<br>■ **Network Function Catalog Read**<br>■ **Network Function Catalog Instantiate**<br>■ **Virtual Infrastructure Audit**<br>■ **Network Service Instance Lifecycle Management**<br>■ **Network Function Instance Lifecycle Management** |
| **System Auditor**<br>The users assigned to this role can view all the entities in VMware Telco Cloud Automation. | ■ **System Audit**<br>■ **Virtual Infrastructure Audit**<br>■ **Network Service Instance Read**<br>■ **Network Service Catalog Read**<br>■ **Network Function Catalog Read**<br>■ **Network Function Instance Read**<br>■ **Partner System Read**<br>■ **Role Audit** |
| **Role Administrator**<br>The users assigned to this role can perform all the object access control related actions in VMware Telco Cloud Automation. | ■ **Role Admin**<br>■ **Role Audit** |

Table 3-2. System Defined Roles (continued)

| Role | Privileges |
|---|---|
| **Partner System Administrator**<br>The users assigned to this role can perform all the partner system-related actions in VMware Telco Cloud Automation. | ■ **Partner System Read**<br>■ **Partner System Admin**<br>■ **Network Function Catalog Read**<br>■ **Virtual Infrastructure Audit** |
| **Partner System Read Only**<br>The users assigned to this role can view all the partner system entities in VMware Telco Cloud Automation. | **Partner System Read** |
| **Role Auditor**<br>The users assigned to this role can view all the object access control related roles and permissions in VMware Telco Cloud Automation. | **Role Audit** |

# Creating Roles and Permissions

Apart from the predefined roles and privileges that are available in VMware Telco Cloud Automation, you can create custom roles and assign specific privileges to them. You can also assign specific access permissions to users and user groups.

## Create a Role

Create a role and assign specific permissions.

### Prerequisites

You must be a **System Administrator** or a **Role Administrator** to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. From the top-right corner, click the drop-down menu next to the **User** icon. Go to **Authorization** > **Roles**.
3. Click **Create Role**.
4. Enter the role name, an optional description, and select the privileges to be associated with that role.
5. Click **Save**.

### Results

Your role is created successfully and is displayed under the list of roles.

### What to do next

- To edit your role, click **Edit**.
- To delete a role, click **Delete**. To delete a role, you must delete all its associated permissions.

You can now create permissions for your role.

## Create Permission

Create permissions that are applicable only to specific users and user groups.

### Prerequisites

You must be a **System Administrator** or a **Role Administrator** to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. From the top-right corner, click the drop-down menu next to the **User** icon. Go to **Authorization** > **Permissions**.

   The existing permissions are displayed.
3. Click **Create Permission**.
4. In the **Create Permission** page, enter the following information:
   - **Role** - Select the role to associate the permission with.
   - **Name** - Enter a unique name for the permission.
   - **Description** - Enter an optional description about the permission.
   - **vCenter User(s) / User Group(s)** - Enter the vCenter user name or the group name to associate the permission with. The format to enter the group name is `domain\groupName`. To validate the user and user group name and to associate the permissions, click **Validate**.
   - **Configure Advanced Filters** - Select this option if you want to add advanced filters such as specific object type, attribute, metric, and their values. For example, you can associate the permissions that you create for a **Network Function Deployer** to access a specific Network Function Catalog, a Network Function Instance, or a Virtual Infrastructure. Click **Add**.
5. Click **Save**.

### Results

Your permission is created successfully and is displayed under the list of permissions.

# Working with Tags

4

Tags allow you to attach metadata to a network function and VIM deployment profiles. Tagging makes it easier to search and sort these objects, and to assign specific rules to the object.

For example, you can assign an SSD tag to your VNFs. This way, you can gently enforce users to deploy these VNFs only on VIMs having SSD as the storage profile.

## Adding Tags to VIMs

You can add tags when adding a VIM, or you can edit an existing VIM to add tags to it. For more information, see Chapter 5 Configuring Your Virtual Infrastructure.

## Adding Tags to Network Function Catalogs

You can add tags when onboarding a network function, or you can edit an existing network function catalog to add tags to it. For more information, see Onboarding a Network Function.

## Overriding Tags

You can override tags when objects are not compatible with each other. For example, if you have a cloud with a CNF tag and you want to instantiate a network function catalog with the VNF tag, you can override the tag. On the **Select Cloud** pop-up window, expand **Advanced Filters**, deselect the **CNF** tag, and click **Apply**.

**Note** When you override a tag, you are explicitly bypassing the system validations and verifying the success yourself.

# Configuring Your Virtual Infrastructure

5

Before creating and instantiating network functions and services, you must add your virtual infrastructure to VMware Telco Cloud Automation.

**Note** VMware Telco Cloud Automation supports vSphere, vCloud Director, Kubernetes Cluster, VMware Tanzu, VMware Integrated OpenStack, VMware Cloud on AWS, Google VMware Engine (GVE), and Microsoft Azure VMware Solution (AVS).

You can add a virtual infrastructure from the **Infrastructure** > **Virtual Infrastructure** page. The Virtual Infrastructure page provides a graphical representation of clouds that are distributed geographically. Details about the cloud such as Cloud Name, Cloud URL, Cloud Type, Tenant Name, Connection Status, and Tags are also displayed. To view more information such as TCA-CP URL, Location, User Name, Network Function Inventory, and so on, click the **>** icon on a desired cloud.

This chapter includes the following topics:

## Add a Cloud to VMware Telco Cloud Automation

The first step to managing network functions and services is to add a cloud to VMware Telco Cloud Automation.

### Prerequisites

- You must have the **Virtual Infrastructure Admin** privileges to perform this task.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Navigate to **Infrastructure** > **Virtual Infrastructure** and click **+ Add**.

The Add New Virtual Infrastructure Account page is displayed.

3 Select a cloud type. Based on the cloud type you select, enter the following virtual infrastructure details:

**Note** VMware Telco Cloud Automation auto-imports self-signed certificates. To import, click **Import** from the pop-up window and continue.

a For **vCloud Director** and **VMware Integrated OpenStack (VMware VIO)**:

| | |
|---|---|
| **Cloud Name** | Enter a name for your virtual infrastructure. |
| **Cloud URL** | Enter the TCA-CP cloud appliance URL. This URL is used for making HTTP requests. |
| **Tags** | Enter the labels to associate with your cloud. |
| **Username** | Enter the user name of a cloud user having edit permissions on the cloud.<br>■ The format for a vCloud Director-based cloud is `username@organization–name`.<br>■ The role for vCloud Director is **Organization Administrator**.<br>■ The role for VMware Integrated OpenStack (VIO) is **Project Administrator**. |
| **Password** | Enter the infrastructure user password. |
| **Tenant Name** | Enter the organization name for vCloud Director. Enter the project name for VIO. |

b For Kubernetes and VMware Tanzu:

| | |
|---|---|
| **Cloud Name** | Enter a name for your virtual infrastructure. |
| **Cloud URL** | Enter the TCA-CP cloud appliance URL. This URL is used for making HTTP requests. |
| **Tags** | Enter the labels to associate with your cloud. |
| **Cluster Name** | Enter the cluster name that you provided when registering the Kubernetes Cluster in TCA-CP Manager. |
| **Kubernetes Config** | Enter the YAML `kubeconfig` file for your Kubernetes Cluster. |

c For VMware vSphere, Microsoft Azure VMware Solution (AVS), and Google VMware Engine (GVE):

| | |
|---|---|
| **Cloud Name** | Enter a name for your virtual infrastructure. |
| **Cloud URL** | Enter the TCA-CP cloud appliance URL. This URL is used for making HTTP requests. |
| **Tags** | Enter the labels to associate with your cloud. |

| | |
|---|---|
| **Username** | Enter the user name of a cloud user having edit permissions on the cloud. The format for the vSphere cloud is `username@domain-name`. |
| **Password** | Enter the infrastructure user password. |

4 Optionally, you can add tags to your cloud. Tags are used for filtering and grouping clouds, network functions, and network services.

5 Click **Validate**.

   The configuration is validated.

6 Click **Add**.

### Results

The cloud is added to your virtual infrastructure. You can see an overview of your virtual infrastructure on the **Infrastructure** > **Virtual Infrastructure** page together with a map showing the physical location of each cloud.

### What to do next

You can click **+ Add** to configure additional clouds in your virtual infrastructure. You can click **Edit** or **Delete** to modify your existing infrastructure.

For vCloud Director, vSphere, and VIO, you must configure the deployment profiles for your cloud.

## Configure the Compute Profile

If you have added a vCloud Director, vSphere, or VIO cloud, you must add a compute profile. Compute Profiles allow you to specify the underlying resource where the virtual network functions are deployed.

### Prerequisites

You must have the **Virtual Infrastructure Admin** privileges to perform this task.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Navigate to **Infrastructure** > **Virtual Infrastructure** and select the desired cloud.

3 Click **Manage Compute Profile**.

4 Under **Compute Profiles**, click **Add**.

5. Enter the following information:

   Compute Profiles allow you to specify the underlying resource where the network functions are deployed.

   - For vCloud Director clouds:
     - **Name** - Name of the compute profile.
     - **Description** - A brief description about the profile.
     - **OrgVdc** - Select the Organization vDC from the pop-up window.
     - **Storage Profile** - Select the storage profile from the pop-up window.
     - **Tag(s)** - Enter the labels to associate your compute profile with.
     - **Location** - Enter the cloud location to add the compute profile. To add the compute profile to the current cloud, select **Same as VIM**.
   - For VIO clouds:
     - **Name** - Name of the compute profile.
     - **Description** - A brief description about the profile.
     - **AvailabilityZone** - Select the Availability Zone.
     - **Tag(s)** - Enter the labels to associate your compute profile with.
     - **Location** - Enter the cloud location to add the compute profile.
   - For VMware vSphere clouds:
     - **Name** - Name of the compute profile.
     - **Description** - A brief description about the profile.
     - **Compute** - Select the resource pool or cluster.
     - **Datastore** - Select the datastore for the resource pool or cluster.
     - **Edge Cluster** - Select the Edge Cluster from vCenter NSX-T.
     - **Folder** - Select the folder to deploy the virtual machines.
     - **Tag(s)** - Enter the labels to associate your compute profile with.
     - **Location** - Enter the cloud location to add the compute profile.
6. Click **Add**.

### Results

The compute profile is added to your cloud. To view the compute profile, navigate to **Infrastructure** > **Virtual Infrastructure** and click the **>** icon against the cloud name.

The **Resource Status** column in the Virtual Infrastructure page displays the resource use of those clouds that are configured with vCloud Director, vSphere, or VIO VIMs.

### What to do next

To edit a compute profile, navigate to **Infrastructure** > **Virtual Infrastructure** and click the cloud name. In the cloud details page, go to the desired compute profile and click the **Edit** icon.

## Edit a Virtual Infrastructure Account

You can edit an existing virtual infrastructure account to updates details such as Cloud Name, Cloud URL, User Name, Password, and so on.

In this example, we edit the virtual infrastructure details of vCloud Director.

### Prerequisites

You must have the **Virtual Infrastructure Admin** privileges to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Navigate to **Infrastructure** > **Virtual Infrastructure** and select the desired virtual infrastructure to edit.
3. Click the **Edit** icon.

   The Edit Virtual Infrastructure Account page is displayed.
4. Under **Virtual Infrastructure Details**, edit the desired details.
5. To Validate the information, click **Validate**.
6. To update the virtual infrastructure account details, click **Update**.

# Viewing Your Cloud Topology

6

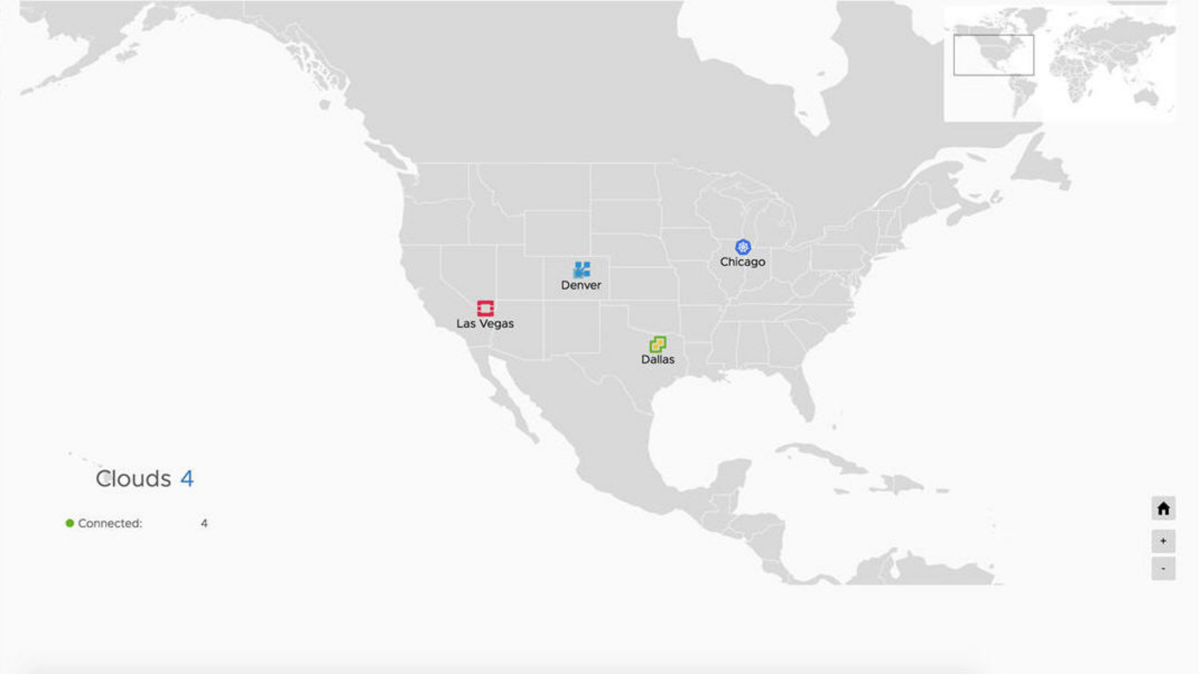
VMware Telco Cloud Automation provides a visual topology of your cloud sites across geographies. It enables administrators to manage network functions and services.

To view your cloud sites and services, perform the following steps:

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 From the left navigation pane, click **Clouds**.



### Results

The **Clouds** page displays the cloud sites that are registered to VMware Telco Cloud Automation.

### What to do next

To view details of a cloud site such as Cloud Name, Cloud Type, User Name, and Status, point to the cloud site.

# Working with Infrastructure Automation

7

Infrastructure Automation can deploy the entire SDDC at Central, Regional or the Cell Site. It automatically deploys the SDDC components like vCenter, NSX, vSAN, vRO, vRLI, TCA-CP, etc on the target hosts. It simplifies the deployments and management of the telecommunication infrastructure.

This chapter includes the following topics:

## Introduction to Infrastructure Automation

Automatic deployment of the telecommunication infrastructure.

You can manage the telecommunication infrastructure through Infrastructure Automation. It also deploys the application on various sites based on the site-specific requirements.

The Infrastructure Automation has four stages.

1 Prerequisites

   Validations related to the prerequisites of each component before deployment.

2 Configuration and Bootstrapping

   Configurations related to networking, appliances, ISO and domains.

3 Automated SDDC Deployment

   Automatic deployment of the SDDC.

4 Ready for Network Functions

   All site is ready. You can configure and initiate the network functions.

### Prerequisites

Infrastructure Automation validates various prerequisites for each site before beginning the actual deployment.

Different sites have different prerequisites that must be fulfilled before beginning the actual deployment. Infrastructure Automation validates all these prerequisites to ensures easy and fast deployment.

## Host

- All hosts in a domain are homogeneous.
- Each host has a minimum of one solid-state disk (SSD) and three solid-state disk/hard disk drives for vSAN.
- Each host requires two physical NICs connected to the same physical switch.

## Physical Switch

- Jumbo Frames enabled on the Physical Switch.
- DHCP enabled on the NSX host overlay network.
- Each ESXi server has a minimum of two physical NICs connected to the switch in trunk mode. Access to all the VLANs (Management Network, vMotion Network, vSAN, NSX Host Overlay Network, NSX Edge Overlay Network, Uplink 1 and Uplink 2) on the trunk port.

## Domain

- Each domain has a minimum of four hosts.
- DNS name configured for all the appliances in all the domains.
- ESXi servers are installed for each domain through the PXE server or ISO image.
- A common web server to access the software images at the central site.
- Central and regional sites have the internet connectivity.

## Network

- TCA/TCA-CP will require an unrestricted communication to connect.tec.vmware.com and hybridity-depot.vmware.com over port TCP 443 for license activation and updates.
- Time synchronization through NTP for all VLANs.
- Unique VLANs are created for following networks on the physical switch:

| Network | MTU | Description |
|---|---|---|
| Management Network | 1500 | Used to connect the management components of the software like vCenter, ESXi, NSX Manager, VMware Telco Cloud Automation, and VMware Telco Cloud Automation Control Panel. |
| vMotion Network | 9000 | Used for the live migration of the virtual machines. It is a L2 routable network and used only for the vMotion traffic within a data center. |
| vSAN | 9000 | Used for the vSAN traffic. It is a L2 routable network and used only for the vSAN traffic within a data center. |

| Network | MTU | Description |
|---|---|---|
| NSX Host Overlay Network | 9000 | Used for the NSX Edge overlay traffic. Requires a routable with Host overlay VLAN in the same site. This network requires a DHCP server to provide IPs to the NSX host overlay vmk interfaces. The DHCP pool should equal the number of ESXi hosts on this network. |
| NSX Edge Overlay Network | 9000 | Used for the overlay traffic between the hosts and Edge Appliances. |
| Uplink 1 | 9000 | Used for the uplink traffic. Uplink 1 is in the same subnet as the Top of Rack (ToR) switch uplink address. |
| Uplink 2 | 9000 | A redundant path for the uplink traffic. Uplink 2 is in the same subnet as the Top of Rack switch uplink address. |

- Each ESXi server has a minimum of two physical NICs connected to the switch in trunk mode. Access to all the VLANs (Management Network, vMotion Network, vSAN, NSX Host Overlay Network, NSX Edge Overlay Network, Uplink 1 and Uplink 2) on the trunk port.
- Name resolution through DNS for all appliances in all the domains.
- DNS records for all appliances with forward and reverse resolution.

**Note** You can create custom naming schemes for the appliances. You can also select the naming schemes from **Appliance Naming Scheme** from the drop-down menu when configuring the global parameters or override the naming schemes when configuring domains. The options available for appliance naming scheme are:

- {applianceName}-{domain-Name}
- {applianceName}
- Custom

For example: If the naming scheme is set to {appliancename}-{domainname}, the name for a Virtual Center appliance is vc-cdc1.telco.example.com, where:

- vc is the appliance name.
- cdc1 is the domain name.
- telco.example.com is the DNS suffix.

## License

The licenses for the following components are required:

- VMware vSphere (ESXi)
- VMware NSX-T Data Center
- VMware Telco Cloud Automation
- VMware Telco Cloud Automation Control Panel
- VMware vCenter Server
- VMware vSAN

- (Optional) VMware vRealize Log Insight

**Note** The actual license requirements may change based on the components installed.

## Supported Software Version

The following table lists the software versions deployed through Infrastructure Automation.

| Software | Version |
|---|---|
| VMware Cloud Builder | 4.1 |
| VMware Telco Cloud Automation | 1.8.0 |
| VMware vRealize Orchestrator | 8.2 |
| VMware vRealize Log Insight | 8.2 |
| VMware Tanzu Kubernetes Grid | 1.2.0 |
| ESXi (supported with Cloud Builder ESXi 7.0 Update 1) | 7.0 Update 1 |
| Kubernetes | 1.17.9, 1.17.11, 1.18.8, 1.19.1 |

## Configuration and Bootstrapping

The configuration and bootstrapping involve configuring the global settings, appliance, images, and domain-specific settings.

The configuration and bootstrapping use two tabs.

- **Configuration**

  On this tab, you can configure global settings, appliances, and images or virtualization files (OVF).

- **Domains**

  On this tab, you can configure network and licenses for various sites. For example, you can configure a central site, a regional site, a compute cluster, or a cell site group. You can also add hosts.

## Automated SDDC Deployment

After the configuration and bootstrapping phase is complete, Infrastructure Automation deploys the software defined data center (SDDC) .

**Note** The SDDC deployment starts only when the minimum number of hosts are registered for a domain and the domain is enabled.

As a part of the SDDC deployment, the following software components are installed according to the domain type.

| Site | Deployment Type |
|---|---|
| Central | A full SDDC with Telco Cloud Automation. It includes:<br>- VMware vCenter<br>- VMware NSX<br>- VMware vRealize Orchestrator<br>- VMware vRealize Log Insight<br>- VMware Telco Cloud Automation<br>- VMware Telco Cloud Automation Control Panel |
| Regional | A full SDDC, includes:<br>- VMware vCenter<br>- VMware NSX<br>- VMware vRealize Orchestrator<br>- VMware Telco Cloud Automation Control Panel<br>The central site controls operations in the regional site. |
| Compute Cluster | A vCenter cluster. A central site or a regional site manages the compute cluster |
| Cell Site Group | A set of ESXi hosts, where RAN is deployed. A central or regional site manages the hosts in the Cell Site Group. |

## Ready for Network Function

The final step of Infrastructure Automation.

Infrastructure Automation deploys all the required application for the sites and makes the site ready for network functions. Design and deployment of network services and function can start. You can now create and initiate the network functions.

## Roles

You can perform different operations based on your role.

Based on the roles and associated permissions, a user can perform different roles in Infrastructure Automation.

### System Administrator

A system administrator can manage all the sites.

A system administrator performs the management of the existing sites that are configured and deployed. A system administrator performs operations that include:

- Adding new licenses.
- Adding new sites.
- Modifying the existing configurations.

**Note** The operations a system administrator can perform depend on the permissions available to the system administrator.

### Management User

A management user can deploy and configure all the sites.

A management user performs day-one operations. The operations include:

- Configuring the sites.
- Adding the licenses.
- Adding the host.

# Deployment Configurations

You can configure the global settings, appliance settings, and provide link to ISO images to deploy.

## Configure Global Settings

You can configure networking parameters.

You can configure **Service** settings and **Proxy Config** settings on the **Global Settings** page.

**Note** You can override the values for each domain when configuring the domains.

Procedure

1 Click the **Configuration** tab under the **Infrastructure Automation**.

2 Click **Global Settings**.

3 To modify the global parameters, click **Edit**.

4 Provide the required details for **Service** parameters.

| Field | Description |
|---|---|
| **DNS Suffix** | Address of the DNS suffix for each appliance. For example: telco.example.com |
| **DNS Server** | The IP address of the DNS server. You can add multiple DNS server IP, separated by comma. |
| **NTP Server** | Name of the NTP server. For example: time.vmware.com. You can add multiple NTP server address, separated by comma. |

5 To use the proxy server, enable **Proxy Config**. Click the **Enabled** button.

6 Provide the required details for **Proxy** parameters.

| Field | Description |
|---|---|
| **Protocol** | Proxy protocol. Select the value from the drop-down menu. |
| **Proxy Server** | IP of the proxy server. |

| Field | Description |
|---|---|
| **Proxy Port** | Port of the proxy server. |
| **Proxy Username** | Optional. User name to access the proxy server. |
| **Proxy Password** | Optional. Password corresponding to the user name to access the proxy server. |
| **Proxy Exclusion** | Optional. List of IP and URLs to exclude from proxy. You can use special characters to provide regex URLs, for example *.abx.xyz.com. |

7 Provide the address of the **SaaS** server. For example, `connect.tec.vmware.com`. It is used for both activation and software updates.

8 Provide the **Appliance Naming Scheme**. Select the value from the drop-down menu. This naming scheme is used for all the appliances added to VMware Telco Cloud Automation.

## Configure Appliances

Configure the IP index and password of various appliances available under the **Appliance Configuration**.

You can configure the IP index and password for all the appliances available in Infrastructure Automation.

**Note** IP index is the index of the IP address in the subnet which is configured in the **Networks** under **Domain** section. The IP for each appliance is derived by adding the IP Index to the subnet address, so that the administrator does not need to provide an IP for each appliance in each domain. It is recommended to follow a common IP addressing scheme for all the domains. However, if required, you can override the IP Index for each domain.

**Note** You can configure the Root Password, Admin Password and Audit Password, and select the **Use above credentials for all the password fields** to use the same password for all the appliances.

| Field | Description |
|---|---|
| Appliance Type | The name of the appliance. It is a non-editable. |
| Appliance Name | The name of the appliance. |
| IP Index | The last octane of the IP address. The first three octanes of the IP address are computed from the IP address of the DNS server. |
| Root Password | Password of the root user of the appliance.<br>**Note** Minimum length of the password is 13 characters and it must include a special character, a capital letter, a lower-case letter, and a number. |
| Admin Password | Password of the administrator of the appliance.<br>**Note** Minimum length of the password is 13 characters and it must include a special character, a capital letter, a lower-case letter, and a number. |

| Field | Description |
|---|---|
| Audit Password | Password of the audit user. Applicable only for NSX Manager, and NSX Edge cluster.<br>**Note** Minimum length of the password is 13 characters and it must include a special character, a capital letter, a lower-case letter, and a number. |
| NSX Manager Configuration | Applicable only for NSX Manager.<br>■ Name: Name of the NSX Manager node.<br>■ IP: The fourth octane of the IP address applicable to the node. |
| NSX Edge Cluster Configuration | Applicable only for NSX Edge Cluster.<br>■ Name: Name of the NSX Edge cluster.<br>■ IP: The fourth octane of the IP address applicable to the node. |
| Node Count | Number of vSAN NFS nodes. Minimum three and a maximum of eight nodes are required. Applicable only for vSAN NFS |
| IP Pool | List of static IP indexes for vSAN NFS nodes. Each vSAN NFS node requires one IP. Applicable only for vSAN NFS. |
| Shares | Size of the NFS share. Applicable only for vSAN NFS. |

### Procedure

1 Click the **Configuration** tab under the **Infrastructure Automation**.

2 Click **Appliance Configuration**.

3 To modify the parameters, click **Edit**.

## Add Images or OVF

Add the URL of the appliance images.

Provide the location where the Infrastructure Automation can locate the install images for all appliances. All the images of application are stored on the web server. Provide the complete link of each appliance image.

To configure the Appliance, follow the steps:

### Procedure

1 Click the **Configuration** tab.

2 Click **Images**.

3 Click **Edit**.

4 Provide complete URL of each appliance image.

> **Note** You can add multiple images for VMware Tanzu Kubernetes Grid and VMware Tanzu Kubernetes Grid - HA Proxy.

# Managing Domains

You can add, delete, and configure various sites to create the infrastructure.

You can add a central site, regional sites, compute clusters, or cell sites in Infrastructure Automation. You can also add a host for each site.

You can modify the details of an already added site and view the appliances related to each site. You can resynchronize the site details after modifying the configurations, to ensure that all the configurations are working correctly.

## Add Central Site

You can add, configure, and manage the central site.

### Prerequisites

Obtain the required licenses and network information required for configuration.

### Procedure

1. Click **Domains** under Infrastructure Automation.
2. Click the **Central Site** icon.
3. Click **Add**. The **Add Site** page appears.
4. On the **Add Site** page, provide the required information.
5. Click the button corresponsing to **Enabled**, to enable the provisioning of the site. If the site is not enabled, no operation can be performed on the site.

| Field | Description |
|---|---|
| **Name** | The name of the site. |
| **Minimum number of hosts** | Minimum number of hosts required for the site. The number of hosts cannot be less than 4 or more than 64. |
| **Location** | The location of the site. Click the button corresponding to the location. |
| **Search** | Type the keyword to search a location. |
| **Latitude** | Latitude of the compute cluster location. The details are automatically added when you select the location. You can also modify the latitude manually. |
| **Longitude** | Longitude of the compute cluster location. The details are automatically added when you select the location. You can also modify the longitude manually. |
| **Settings** | You can modify the service settings and the proxy settings for each site. These configurations override the global configuration available in **Global Configuration** tab on **Configuration** page. For more details on service and proxy parameters, see Configure Global Settings. |

| Field | Description |
|---|---|
| **Licenses** | Licenses of various appliances applicable to the site. These appliances include:<br>■ VMware vSphere (ESXi)<br>■ VMware NSX-T Data Center<br>■ VMware Telco Cloud Automation<br>■ VMware Telco Cloud Automation Control Panel<br>■ VMware vCenter Server<br>■ VMware vRealize Log Insight<br>■ VMware vSAN |
| **Services** | You can enable the networking and storage operations for the specific site. |

6 Add the **Switch Configuration** information. Click **plus** icon to add more switches and uplinks.

| Field | Description |
|---|---|
| **Switch** | Name of the switch. |
| **Uplinks** | Select the network interface card (NIC) for the central site under **Uplinks**.<br>**Note** A central site requires minimum two NICs to communicate. NIC details should match the actual configuration across all ESXi servers. |

7 Add the **Networks** information.

**Note**

- For vMotion and vSAN, the IP pool should equal the total number of ESXi hosts.
- You can click **+** sign under **Networks** to create additional VLAN or Verlay network to connect with additional applications.

| Field | Description |
|---|---|
| **Name** | The name of the network. |
| **Segment Type** | Segment type of the network. Select the value from the list. |
| **Network Type** | The type of the network. |
| **Switch** | The switch details which the sites use for network access. |
| **VLAN** | The VLAN ID for the network. |
| **MTU** | The MTU length (in bytes) for the network. |
| **Prefix Length** | The prefix length for each packet for the network. |
| **Gateway Address** | The gateway address for the network. |

8 (Optional) Add the **Appliance Overrides** information. Ensure that the appliance names match the actual names entered in DNS. If they do not match, you can change the name.

**Note** For NSX-Edge cluster configuration:

- To override the Edge form factor, select the **Size** from the drop-down menu.
- To override the HA, select the **Tier0Mode** from the drop-down menu.

| Field | Description |
|---|---|
| **Override** | Whether to override the current values. |
| **Appliance Type** | The type of the appliance. |
| **Name** | The name of the appliance. |
| **Name Override** | The new name of the appliance to override the previous name of appliance. |
| **IP Index** | The IP index of the appliance. The value is fourth octane of the IP address. The initial three octanes are populated from the network address provided in domain. |
| **Enabled** | Whether the appliance is enabled and available for operations. |

### What to do next

Add Host to a Site.

## Edit a Central Site

You can modify the central site details and add the host for the central site host.

You can modify the configuration of central site, add a host, and view the list of appliances applicable to the central site.

### Procedure

1 Click **Domains** under Infrastructure Automation.

2 Click the **Central Site** icon.

3 Select the central site to edit.

4 Click **Edit**.

5 To modify the configurations, click the **Configuration** tab.

6 To add a new central site host, click the **Host** tab.

7 To view the list of available appliances, click the **Appliance** tab.

## Add Regional Site

You can add, configure, and manage the regional site.

To add a regional site, follow the steps:

### Prerequisites

Obtain the licenses and network information required for configuration.

### Procedure

1. Click **Domains** under Infrastructure Automation.
2. Click the **Regional Site** icon.
3. Click **Add**.

   The **Add Site** page appears.
4. On the **Add Site** page, provide the required information.
5. Click the button corresponsing to **Enabled**, to enable the provisioning of the site. If the site is not enabled, no operation can be performed on the site.

| Field | Description |
|---|---|
| **Name** | The name of the site. |
| **Minimum number of hosts** | Minimum number of hosts required for the site. The number of hosts cannot be less than 4 or more than 64. |
| **Location** | The location of the site. Click to add the location details. |
| **Search** | Type the keyword to search a location. |
| **Latitude** | Latitude of the compute cluster location. The details are automatically added when you select the location. You can also modify the latitude manually. |
| **Longitude** | Longitude of the compute cluster location. The details are automatically added when you select the location. You can also modify the longitude manually. |
| **Settings** | You can modify the service settings and the proxy settings for each site. These configurations override the global configuration available in Global Configuration tab on Configuration page. For more details on service and proxy parameters, see Configure Global Settings. |
| **Licenses** | Licenses of various appliances applicable to the site. These appliances include:<br>■ VMware vSphere (ESXi)<br>■ VMware NSX-T Data Center<br>■ VMware Telco Cloud Automation Control Panel<br>■ VMware vCenter Server<br>■ VMware vRealize Log Insight<br>■ VMware vSAN |
| **Services** | You can enable the networking and storage operations for the specific site. |

6 Add the **Switch Configuration** information. Click **plus** icon to add more switches and uplinks.

| Field | Description |
|---|---|
| **Switch** | The name of the switch. |
| **Uplinks** | Select the network interface card (NIC) for the regional site under **Uplinks**.<br>**Note** A regional site requires minimum two NICs to communicate. NIC details should match the actual configuration across all ESXi servers. |

7 Add the **Networks** information.

**Note**

- For vMotion and vSAN, the IP pool should be equal to the total number of ESXi hosts.
- You can click **+** sign under **Networks** to create additional VLAN or Verlay network to connect with additional applications.

| Field | Description |
|---|---|
| **Name** | The name of the network. |
| **Segment Type** | Segment type of the network. Select the value from the list. |
| **Network Type** | The type of the network. |
| **Switch** | The switch details which the site uses to access network. |
| **VLAN** | The VLAN ID for the network. |
| **MTU** | The MTU length (in bytes) for the network. |
| **Prefix Length** | The Prefix length for each packet for the network. |
| **Gateway Address** | The gateway address for the network. |
| **Network Address** | The network address for the network. |

8 (Optional) Add the **Appliance Overrides** information. Ensure that the appliance names match the actual names entered in DNS. If they do not match, you can change the name.

**Note** For NSX-Edge cluster configuration:

- To override the Edge form factor, select the **Size** from the drop-down menu.
- To override the HA, select the **Tier0Mode** from the drop-down menu.

| Field | Description |
|---|---|
| **Override** | Whether to override the current values. |
| **Appliance Type** | The type of the appliance. |
| **Name** | The name of the appliance. |
| **Name Override** | The new name of the appliance to override the previous name of appliance. |

| Field | Description |
|---|---|
| **IP Index** | IP index of the appliance. The value is fourth octane of the IP address. The initial three octanes are populated from the network address provided in domain. |
| **Enabled** | Whether the appliance is enabled and available for operations. |

What to do next

Add Host to a Site.

## Edit a Regional Site

You can modify the regional site details and add a host.

You can modify the configuration of regional site or add a host, or view the list of appliances applicable to the regional site.

To modify a regional site, follow the steps:

### Procedure

1 Click **Domains** under Infrastructure Automation.

2 Click the **Regional Site** icon.

3 Select the regional site to edit.

4 Click **Edit**.

5 To modify the configurations, click the **Configuration** tab.

6 To add a new regional site or cell site host, click the **Host** tab.

7 To view the list of applicable appliances, click the **Appliance** tab.

## Add Compute Cluster

A compute cluster is a combination of sites managed by a regional or central site.

### Procedure

1 Click **Domains** under Infrastructure Automation.

2 Click the **Compute Cluster** icon.

3 Click **Add**.

The **Add Site** page appears.

4 On the **Add Site** page, provide the required information.

5 Click the button corresponsing to **Enabled**, to enable the provisioning of the site. If the site is not enabled, no operation can be performed on the site.

| Field | Description |
|---|---|
| **Name** | The name of the site. |
| **Minimum number of hosts** | Minimum number of hosts required for the site. The number of hosts cannot be less than 4 or more than 64. |
| **Parent Site** | The regional site or central site that manages the cluster. Select from the list. |
| **Location** | The location of the compute cluster. |
| **Search** | Type the keyword to search a location. |
| **Latitude** | Latitude of the compute cluster location. The details are automatically added when you select the location. You can also modify the latitude manually. |
| **Longitude** | Longitude of the compute cluster location. The details are automatically added when you select the location. You can also modify the longitude manually. |
| **Settings** | You can modify the service settings and the proxy settings for each site. These configurations override the global configuration available in Global Configuration tab on Configuration page. For more details on service and proxy parameters, see Configure Global Settings. |
| **Licenses** | Not applicable. The compute cluster uses the licenses of parent site. |
| **Services** | ■ For a compute cluster, you can enable the NSX services. For certain workloads, if you do not require these services, you can disable these services.<br>■ To use the network services of the parent site, click the **Share Transport Zones With Parent** button.<br>■ You can use the **vSAN** or **localstore**. Select the value from the drop-down menu.<br>Click **Enabled** button to enable or disable the **Networking** or **Storage** services. |

6 Add the **Switch Configuration** information. Click **plus** icon to add more switches and uplinks.

| Field | Description |
|---|---|
| **Switch** | The name of the switch. |
| **Uplinks** | Select the network interface card (NIC) for the compute cluster under **Uplinks**.<br>**Note** A cell site requires minimum two NICs to communicate. This should match the actual configuration across all ESXi servers. |

7 Add the **Networks** information.

**Note**

- For vMotion and vSAN, the IP pool should be equal to the total number of ESXi hosts. If you do not provision the appliances, vSAN, nsxHostOverlay, nsxEdgeOverlay, uplinks are optional.
- You can click **+** sign under **Networks** to create additional VLAN or Verlay network to connect with additional applications.

| Field | Description |
|---|---|
| **Name** | The name of the network. |
| **Segment Type** | Segment type of the network. Select the value from the list. |
| **Network Type** | The type of the network. |
| **Switch** | The switch details which the sites use for network access. |
| **VLAN** | VLAN ID for the network. |
| **MTU** | MTU length (in bytes) for the network. |
| **Prefix Length** | Prefix length for each packet for the network. |
| **Gateway Address** | The gateway address for the network. |

8 (Optional) Add the **Appliance Overrides** information. Ensure that the appliance names match the actual names entered in DNS. If they do not match, you can change the name.

**Note** For NSX-Edge cluster configuration:

- To override the Edge form factor, select the **Size** from the drop-down menu.
- To override the HA, select the **Tier0Mode** from the drop-down menu.

| Field | Description |
|---|---|
| **Override** | Whether to override the current values. |
| **Appliance Type** | The type of the appliance. |
| **Name** | The name of the appliance. |
| **Name Override** | The new name of the appliance to override the previous name of appliance. |
| **IP Index** | IP index of the appliance. The value is fourth octane of the IP address. The initial three octanes are populated from the network address provided in domain. |
| **Enabled** | Whether the appliance is enabled and available for operations. |

## Edit a Compute Cluster

Modify the compute cluster details.

You can modify the configuration of a compute cluster and add a host.

### Procedure

1 Click **Domains** under Infrastructure Automation.

2 Click the **Compute Cluster** icon.

3 Select the compute cluster to edit.

4 Click **Edit**.

5 To modify the configurations, click the **Configuration** tab.

6 To add a new host, click the **Host** tab.

## Add a Cell Site Group

You can add, manage or delete a cell site group.

To add a regional site, follow the steps:

### Prerequisites

Obtain the network information required for configuration.

### Procedure

1 Click **Domains** under Infrastructure Automation.

2 Click the **Cell Site Group** icon.

3 Click **Add**.

   The **Add Site** page appears.

4 On the **Add Site** page, provide the required information.

5 Click the button corresponsing to **Enabled**, to enable the provisioning of the site. If the site is not enabled, no operation can be performed on the site.

| Field | Description |
|---|---|
| **Name** | The name of the site. |
| **Parent Site** | Select the parent site from the list. The parent site manages all the sites within the cell site group. |
| **Settings** | You can modify the service settings and the proxy settings for each site. These configurations override the global configuration available in Global Configuration tab on Configuration page. For more details on service and proxy parameters, see Configure Global Settings. |

6 Add the **Switch Configuration** information. Click **plus** icon to add more switches and uplinks.

| Field | Description |
|---|---|
| **Switch** | The name of the switch. |
| **Uplinks** | Select the network interface card (NIC) for the site under **Uplinks**.<br>**Note** A site requires minimum two NICs to communicate. NIC details should match the actual configuration across all ESXi servers. |

7 Add the **Networks** information.

**Note** System defines the Management network for a cell site group. User can create custom VLAN based application networks. All cell sites in a cell site group connect with same management network.

| Field | Description |
|---|---|
| **Name** | The name of the network. |
| **Segment Type** | Segment type of the network. Select the value from the list. |
| **Network Type** | The type of the network. |
| **Switch** | The switch details which the sites use for network access. |
| **VLAN** | The VLAN ID for the network. |
| **MTU** | The MTU length (in bytes) for the network. |
| **Prefix Length** | The Prefix length for each packet for the network. |
| **Gateway Address** | The gateway address for the network. |

## Edit a Cell Site Group

You can modify the cell site group details.

You can modify the configuration of cell site group, add a host, and modify the network configurations related to cell site group.

To modify a cell site group, follow the steps:

### Prerequisites

A cell site group is configured.

### Procedure

1 Click **Domains** under Infrastructure Automation.

2 Click the **Cell Site Group** icon.

3 Select the cell site group to edit.

4 Click **Edit**.

## Add Host to a Site

A minimum number of hosts are required for each site to start the automated deployment for each site.

You can add a host to any site or site cluster. A minimum number of hosts are required for each site type to function. You can define the minimum number of hosts for each site when adding the site.

### Prerequisites

A site type for which you want to add a host is already added in **Domains**.

### Procedure

1 Click the **Domains** tab under Infrastructure Automation.

2 Select the site type for which you want to add a host.

3 Select the site for which you want to add a host.

4 Click **Edit** to modify the site details.

5 On the **Host** tab, click **Add Host**.

6 Configure the network for the host.

| Fields | Description |
|---|---|
| **Domain Type** | Name of the domain for which you want to add a host. |
| **Host Address (IP/FQDN)** | IP address or the associated FQDN of the host. |
| **User Name** | User name to access the host. |
| **Password** | Password corresponding to the user name to access the host. |

7 Click **Save**.

## Edit a Host

Modify an already created host.

To modify the configurations of an already created host or delete an unprovisioned host, perform the following:

### Prerequisites

A host is already added to a site.

### Procedure

1 Click **Domains** under Infrastructure Automation.

2 Click the site type under which you want to modify the host.

3 Select the site to edit.

4 Click **Edit**.

5 To modify a host, click the **Host** tab.

6 Select the host to edit. You can perform the following operations on the selected host:

   - Click **Edit Host** to edit the host configuration.
   - Click **Delete Host** to delete an unprovisioned host.
   - Click **Refresh** to enforce the changes made to a host.

## Viewing Tasks

You can view the status of the current and the past tasks executed in Infrastructure Automation.

You can view the status of all the tasks. This includes the current task and the older tasks. You can view the progress, status, and the start and end time of the task.

### Procedure

1 Click **Tasks** tab under Infrastructure Automation.

   A list of task appears.

2 Click the task for which you want to view details.

# Working with Kubernetes Clusters

# 8

A Kubernetes cluster is a set of nodes that run containerized applications.

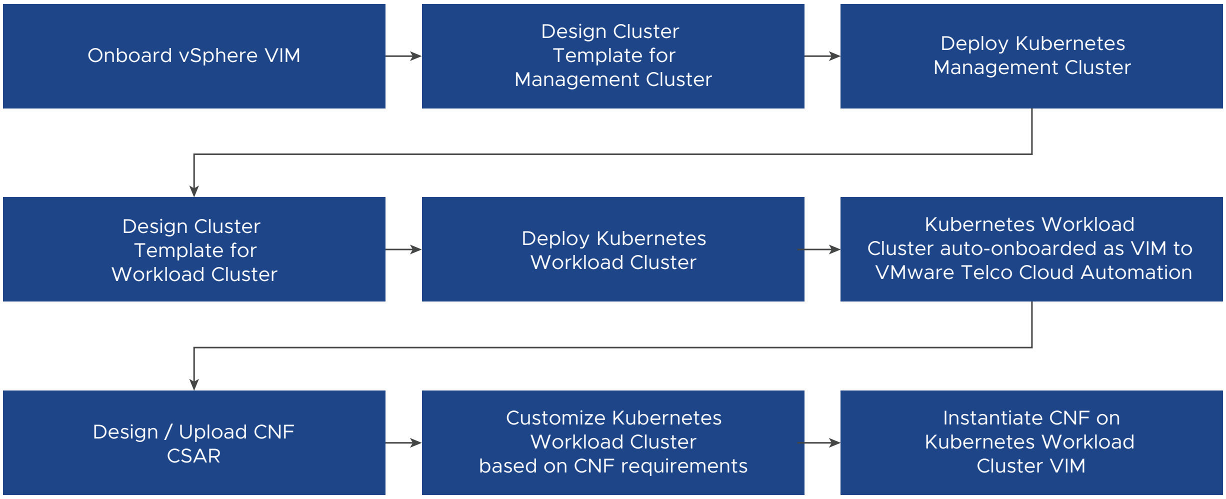
Containerized applications are more lightweight and flexible than virtual machines, and they share the operating system. In this way, Kubernetes clusters allow for applications to be more easily developed, moved, and managed. Kubernetes clusters allow containers to run across multiple machines and environments: Virtual, physical, cloud-based, and on-premises. For more information about Kubernetes clusters and its components, see the Kubernetes documentation at https://kubernetes.io/docs/concepts/overview/.

There must be a minimum of one controller node and one worker node for a Kubernetes cluster to be operational. For production and staging, the cluster is distributed across multiple worker nodes. For testing, the components can all run on the same physical or virtual node.

Network functions require special customizations such as *Real-Time Kernel* and *HugePages* on Kubernetes Worker nodes. The advantage of deploying Kubernetes clusters through VMware Telco Cloud Automation is that, it customizes the Kubernetes clusters according to its network function requirement before deploying the CNFs.

**Note** The Node Customization feature is applicable only when you deploy Kubernetes clusters through VMware Telco Cloud Automation.

# Kubernetes Cluster Deployment Process


Onboard vSphere VIM
Design Cluster Template for Management Cluster
Deploy Kubernetes Management Cluster
Design Cluster Template for Workload Cluster
Deploy Kubernetes Workload Cluster
Kubernetes Workload Cluster auto-onboarded as VIM to VMware Telco Cloud Automation
Design / Upload CNF CSAR
Customize Kubernetes Workload Cluster based on CNF requirements
Instantiate CNF on Kubernetes Workload Cluster VIM

This chapter includes the following topics:

- Kubernetes Cluster Upgrade Flow
- Working with Kubernetes Cluster Templates
- Deploying a Kubernetes Cluster
- Viewing Cluster Details
- Editing Kubernetes Clusters
- Upgrading the Node Pool

# Kubernetes Cluster Upgrade Flow

Upgrade sequence for post VMware Telco Cloud Automation upgrade.

VMware Telco Cloud Automation 1.8 is integrated with the VMware Tanzu Kubernetes Grid (TKG) 1.2. The TKG 1.2 requires you to upgrade the magement cluster. After upgrading the VMware Telco Cloud Automation version, perform the following in the sequence provided.

1 Upgrade the management cluster. For details, see Upgrade Kubernetes Version. For implications of not upgrading the management cluster, see Implications of Not Upgrading Management Cluster.

2 Upgrade the Workload cluster node pools. For details, see Upgrading the Node Pool. For implications of not upgrading the nodepool, see Implications of Not Upgrading Node Pool.

**Note** The step is mandatory when upgrading from VMware Telco Cloud Automation 1.7 to VMware Telco Cloud Automation 1.8. For other version, this step is not required.

3 Upgrade the workload cluster. For details, see Upgrade Kubernetes Version. For implications of not upgrading the workload cluster, see Implications of Not Upgrading Workload Cluster

**Note** Upgrade is mandatory for 1.17.3 and 1.18.2 workload clusters. It is optional for 1.18.6 workload cluster.

## Implications of Not Upgrading Management Cluster

Not upgrading a Management cluster can impact various operations.

Not upgrading the management node can impact:

- Ability to edit the management cluster.
- Ability to create, upgrade, and modify the workload cluster managed through the management cluster.
- Ability to create, upgrade, and modify the workload node pool cluster managed through the management cluster.
- Ability to upgrade and initiate the CNF in the workload cluster managed through the management cluster.

## Implications of Not Upgrading Node Pool

Not upgrading a node pool can impact various operations.

Not upgrading the node pool can impact:

- Ability to edit the node pool.
- Ability to upgrade and initiate CNF on the node pool cluster.

## Implications of Not Upgrading Workload Cluster

Not upgrading a Workload cluster can impact various operations.

Not upgrading the workload cluster running on Kubernetes 1.17.3 and 1.18.2 can impact:

- Ability to edit the workload cluster.
- Ability to create, upgrade, and modify the node pool cluster.
- Ability to upgrade and initiate the CNF.

# Working with Kubernetes Cluster Templates

A Kubernetes cluster template is a blueprint of the Kubernetes cluster and contains the required configuration. Before creating a Kubernetes cluster, you must create a Kubernetes cluster template to deploy the cluster. Using VMware Telco Cloud Automation, you can create a Kubernetes cluster template, upload, download, edit, and, use it to deploy multiple clusters.

When you define the Kubernetes cluster template, select whether it is a Management cluster type or a Workload cluster type.

- Management cluster - A Management cluster is a Kubernetes cluster that performs the role of the primary management and operational center. You use the Management cluster to manage multiple Workload clusters.
- Workload cluster - The clusters where the actual application resides. Network Functions are deployed on the Workload clusters.

When creating a Kubernetes cluster template for a Management cluster or a Workload cluster, you must provide two types of configuration information:

- Cluster Configuration - Specify the details about the Container Storage Interfaces (CSI) such as vSphere-CSI and NFS Client, Container Network Interface (CNI) such as *Antrea*, *Calico*, and *Multus*, version of Kubernetes, and tools such as Helm Charts.
- Master Node and Worker Node Configuration - Here, you specify the details about the master and worker node VMs. Specify details such as the storage, CPU, memory size, number of networks, labels, number of replicas for the master nodes, and worker nodes, and so on.

## Create a Management Cluster Template

Create a Management cluster template and use it to deploy your Kubernetes Management cluster.

### Prerequisites

To perform this operation, you require a role with **Infrastructure Design** privilege.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **Caas Infrastructure** > **Cluster Templates** and click **Add**.

   The **Add Kubernetes Template** wizard is displayed.
3. In the **Template Detail** tab, provide the following details:
   - **Name** - Enter the name of the template.
   - **Cluster Type** - Select **Management Cluster**.
   - **Description** (Optional) - Enter a description for the template.
   - **Tags** (Optional) - Add appropriate tags to the template.
   - **Kubernetes Version** - By default, the latest version of Kubernetes is selected.

   **Note** The supported Container Network Interface (CNI) for a Management cluster is *Antrea*.
4. Click **Next**.

5 In the **Master Node Configuration** tab, enter the following details:

- **Name** - Name of the profile
- **CPU** - Number of vCPUs
- **Memory** - Memory in GB
- **Storage** - Storage size in GB
- **Replica** - Number of controller node VMs to be created. The ideal number of replicas for production or staging deployment is 3.
- **Networks** - Enter the labels to group the networks. The minimum number of labels required to connect to the management network is 1. Network labels are used for providing networks inputs when deploying a cluster. Meaningful network labels such as N1, N2, N3, and so on, help the deployment users provide the correct network preferences. To add more labels, click **Add**.

  **Note** For the Management network, master node supports only one label.

- **Labels** (Optional) - Enter the appropriate labels for this profile. These labels are applied to the Kubernetes node. To add more labels, click **Add**.

6 To use the vSphere Linked Clone feature for creating linked clones for the Kubernetes nodes, click **Advanced Configuration** and select **Use Linked Cloning for Cloning the VMs**.

7 Click **Next**.

8 In the **Worker Node Configuration** tab, add a node pool. A node pool is a set of nodes that have similar properties. Pooling is useful when you want to group the VMs based on the number of CPUs, storage capacity, memory capacity, and so on. You can add one node pool to a Management cluster and multiple node pools to a Workload cluster, with different groups of VMs. To add a node pool, enter the following details:

- **Name** - Name of the profile
- **CPU** - Number of vCPUs
- **Memory** - Memory in GB
- **Storage** - Storage size in GB
- **Replica** - Number of controller node VMs to be created.
- **Networks** - Enter the labels to group the networks. Network labels provide networks inputs when deploying a cluster. To add more labels, click **Add**.
- **Labels** - Enter the appropriate labels for this profile. These labels are added to the Kubernetes node. To add more labels, click **Add**.

9 To use the vSphere Linked Clone feature for creating linked clones for the Kubernetes nodes, click **Advanced Configuration** and select **Use Linked Cloning for Cloning the VMs**.

10 Click **Next** and review the configuration.

11 Click **Add Template**.

### Results

The template is created.

### What to do next

Create a Workload cluster template.

## Create a Workload Cluster Template

Create a Workload cluster template and use it to deploy your workload clusters.

### Prerequisites

To perform this operation, you require a role with **Infrastructure Design** privileges.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Go to **Caas Infrastructure** > **Cluster Templates** and click **Add**.

3 In the **Template Details** tab, provide the following details:

- **Name** - Enter the name of the Workload cluster template.
- **Cluster Type** - Select **Workload Cluster**.
- **Description** (Optional) - Enter a description for the template.
- **Tags** (Optional) - Add appropriate tags to the template.

4 Click **Next**.

5 In the **Cluster Configuration** step, provide the following details:

- **Kubernetes Version** - Select the Kubernetes version from the drop-down menu. The supported versions are 1.17.9, 1.17.11, 1.18.8, and 1.19.1.

- **CNI** - Click **Add** and select a Container Network Interface (CNI). The supported CNIs are **Multus**, **Calico**, and **Antrea**. To add additional CNIs, click **Add** under **CNI**.

  **Note**

  - Either **Calico** or **Antrea** and only one of them must be present. **Multus** is mandatory when the network functions require any CNI plugins such as **SRIOV** or **Host-Device**.
  - You can add CNI plugins such as SRIOV as a part of Node Customization when instantiating, upgrading, or updating a CNF.
  - The following CNIs or CNI plugins are available by default:

    ```
    bandwidth
    dhcp
    flannel
    host-local
    loopback
    ptp
    static
    vlan
    bridge
    firewall
    host-device
    ipvlan
    macvlan
    portmap
    sbr
    tuning
    ```

- **CSI** - Click **Add** and select a Container Storage Interface (CSI) such as vSphere CSI or NFS Client. For more information, see https://vsphere-csi-driver.sigs.k8s.io/ and https://github.com/kubernetes-sigs/nfs-subdir-external-provisioner.
  - **Timeout** (Optional) (For vSphere CSI) - Enter the CSI driver call timeout in seconds. The default timeout is 300 seconds.
  - **Storage Class** - Enter the storage class name. This storage class is used to provision Persistent Volumes dynamically. A storage class with this name is created in the Kubernetes cluster. The storage class name defaults to `vsphere-sc` for the vSphere CSI type and `nfs-client` for the NFS Client type.
  - **Default Storage Class** - To set this storage class as default, enable the **Default Storage Class** option. The storage class defaults to **True** for the vSphere CSI type. It defaults to **False** for the NFS Client type. Only one of these types can be the default storage class.

    **Note** Only one vSphere CSI type and one NFS Client type storage class can be present. You cannot add more than one storage class of the same type.

  - To add additional CSIs, click **Add** under **CSI**.

- **Tools** - The current supported tool is Helm. Helm helps in troubleshooting the deployment or upgrade of a network function.
  - Helm version 3, for example, 3.3.1, is mandatory when the NFS Client CSI is added.
  - Helm version 2, for example, 2.15.2, is mandatory when the network functions deployed on this cluster depend on Helm v2.

    **Note** Only one Helm version 2 and one Helm version 3 can be installed.

  Click **Add** and select **Helm** from the drop-down menu. Enter the Helm version.

6 Click **Next**.

7 In the **Master Node Configuration** tab, enter the following details:

- **Name** - Name of the pool
- **CPU** - Number of vCPUs
- **Memory** - Memory in GB
- **Storage** - Storage size in GB
- **Replica** - Number of controller node VMs to be created. The ideal number of replicas for production or staging deployment is 3.
- **Networks** - Enter the labels to group the networks. The minimum number of labels required to connect to the management network is 1. Network labels are used for providing networks inputs when deploying a cluster. Meaningful network labels such as N1, N2, N3, and so on, help the deployment users provide the correct network preferences. To add more labels, click **Add**.
- **Labels** (Optional) - Enter the appropriate labels for this profile. These labels are applied to the Kubernetes node. To add more labels, click **Add**.

  **Note** For the Management network, master node supports only one label.

8 To use the vSphere Linked Clone feature for creating linked clones for the Kubernetes nodes, click **Advanced Configuration** and select **Use Linked Cloning for Cloning the VMs**.

9 In the **Worker Node Configuration** tab, add a node pool. A node pool is a set of nodes that have similar VMs. Pooling is useful when you want to group the VMs based on the number of CPUs, storage capacity, memory capacity, and so on. You can add multiple node pools with different groups of VMs. Each node pool can be deployed on a different cluster or a resource pool.

**Note** All Worker nodes in a node pool contain the same *Kubelet* and operating system configuration. Deploy one network function with infrastructure requirements on one node pool.

You can create multiple node pools for the following scenarios:

- When you require the Kubernetes cluster to be spanned across multiple vSphere clusters.

- When the cluster is used for multiple network functions that require node customizations.

To add a node pool, enter the following details:

- **Name** - Name of the node pool
- **CPU** - Number of vCPUs
- **Memory** - Memory in MB
- **Storage** - Storage size in GB
- **Replica** - Number of controller node VMs to be created.
- **Networks** - Enter the labels to group the networks. Networks use these labels to provide network inputs during a cluster deployment. Add additional labels for network types such as IPvlan, MacVLAN, and Host-Device. Meaningful network labels such as N1, N2, N3, and so on, help users provide the correct network preferences during deployment. It is mandatory to include a management interface label. SR-IOV interfaces are added to the Worker nodes when deploying the network functions.

  **Note** A label length must not exceed 15 characters.

  Apart from the management network, which is always the first network, the other labels are used as interface names inside the Worker nodes. For example, when you deploy a cluster using the template with the labels **MANAGEMENT**, **N1**, and **N2**, the Worker nodes interface names are **eth0**, **N1**, **N2**. To add more labels, click **Add**.
- **Labels** - Enter the appropriate labels for this profile. These labels are applied to the Kubernetes node and you can use them as node selectors when instantiating a network function. To add more labels, click **Add**.

10 Under **CPU Manager Policy**, set CPU reservations on the Worker nodes as **Static** or **Default**. For information about controlling CPU Management Policies on the nodes, see the Kubernetes documentation at https://kubernetes.io/docs/tasks/administer-cluster/cpu-management-policies/.

  **Note** For CPU-intensive workloads, use **Static** as the CPU Manager Policy.

11 To use the vSphere Linked Clone feature for creating linked clones for the Kubernetes nodes, click **Advanced Configuration** and select **Use Linked Cloning for Cloning the VMs**.

12 Click **Next** and review the configuration.

13 Click **Add Template**.

### Results

The template is created.

### What to do next

Deploy a Management or Workload cluster.

## Edit a Kubernetes Cluster Template

You can edit a cluster template to update its description, cluster configuration, tags, Kubernetes version, master node configuration details, and worker node configuration details.

**Note**

- If you have created the Kubernetes cluster template using VMware Telco Cloud Automation version 1.7 and have upgraded to version 1.8, you must update its Kubernetes version.
- Ensure that the network label length does not exceed 15 characters.
- Editing the Kubernetes cluster template does not change the cluster instances that are already deployed.

To perform this operation, you require a role with **Infrastructure Design** privileges.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **CaaS Infrastructure** and click the **Cluster Templates** tab.
3. Select the Kubernetes cluster template that you want to edit.
4. Click **Edit**.
5. In the Edit Kubernetes Template wizard, make the required updates to the template details, master node configuration, and worker node configuration fields.
6. Review the updates and click **Update Template**.

### Results

You have successfully updated the cluster template.

## Download and Upload a Kubernetes Cluster Template

You can download a Kubernetes cluster template as a JSON file and upload it to another environment. This option is useful when you want to share a validated cluster template across multiple environments.

**Note** To make sure that all the features are available, download or upload a Kubernetes cluster template of the same VMware Telco Cloud Automation version.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **CaaS Infrastructure** > **Cluster Templates**.
3. To download, select the cluster template and click **Download**.

   The cluster template downloads as a JSON file.

4 To upload the JSON file to a different environment, navigate to the environment and log in to the VMware Telco Cloud Automation web interface.

5 Go to **CaaS Infrastructure** > **Cluster Templates**.

6 Click **Upload** and select the JSON file.

7 Click **Upload**.

The cluster template uploads to your environment and is available in the **CaaS Infrastructure** > **Cluster Templates** tab.

## Delete a Kubernetes Cluster Template

Delete a Kubernetes cluster template from VMware Telco Cloud Automation.

**Note** You cannot delete a Kubernetes template when it is being used for deploying a cluster.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Go to **CaaS Infrastructure** and click the **Cluster Templates** tab.

3 Select the Kubernetes cluster template that you want to delete.

4 Click **Delete**.

5 Confirm the delete operation.

### Results

The cluster template is deleted from VMware Telco Cloud Automation.

# Deploying a Kubernetes Cluster

Deploy your CNF on a Kubernetes cluster.

VMware Telco Cloud Automation uses VMware Tanzu Kubernetes Grid to create VMware Tanzu Kubernetes clusters. VMware Tanzu Kubernetes Grid has concepts such as Management and Workload clusters. The Management cluster manages the Workload clusters and both these clusters can be deployed on different vCenter Servers.

For more information about the VMware Tanzu Kubernetes Grid concepts, see *Tanzu Kubernetes Grid Concepts* at https://docs.vmware.com/en/VMware-Tanzu-Kubernetes-Grid/index.html.

## Deploy a Management Cluster

Deploy a Management cluster using the Kubernetes cluster template. You can deploy Management clusters in parallel.

### Prerequisites

- You require a role with **Infrastructure Lifecycle Management** privileges.

- You must have uploaded the Virtual Machine template to VMware Telco Cloud Automation.
- You must have onboarded a vSphere VIM.
- You must have created or uploaded a Management cluster template.
- A network must be present with the DHCP range and the static IP address of the same subnet.

Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Go to **CaaS Infrastructure** and click **Deploy Kubernetes Cluster**.

   - If you have saved a validated Management cluster configuration that you want to replicate on this cluster, click **Upload** on the top-right corner and upload the JSON file. The fields are then auto-populated with this configuration information and you can edit them as required.
   - If you want to create a Management cluster configuration from the beginning, perform the next steps.

3 Select a cloud on which you want to deploy the Kubernetes cluster.

4 Click **Next**.

5 The **Select Cluster Template** tab displays the available Kubernetes cluster templates. Select the Management Kubernetes cluster template that you have created.

   **Note** If the template displays as **Not Compatible**, edit the template and try again.

6 Click **Next**.

7 In the **Kubernetes Cluster Details** tab, provide the following details:

   - **Name** - Enter the cluster name. The cluster name must be compliant with DNS hostname requirements as outlined in RFC-952 and amended in RFC-1123.
   - **Description** (Optional) - Enter an optional description of the cluster.
   - **Password** - Create a password to log in to the Master and Worker nodes. The default user name is *capv*.

     **Note** Ensure that the password meets the minimum requirements displayed in the UI.

   - **Confirm Password** - Confirm the password that you have entered.
   - **OS Image With Kubernetes** - The pop-up menu displays the OS image templates in your vSphere instance that meet the criteria to be used as a Tanzu Kubernetes Grid base OS image with the selected Kubernetes version. If there are no templates, ensure that you upload the them to your vSphere environment.

- **Virtual IP Address** - VMware Tanzu Kubernetes Grid 1.2 deploys a `kube-vip` pod that provides load-balancing services to the cluster API server. This `kube-vip` pod uses a static virtual IP address to load-balance API requests across multiple nodes. Assign an IP address that is not within your DHCP range, but in the same subnet as your DHCP range.
- **Syslog Servers** - Add the syslog server IP address/FQDN for capturing the infrastructure logs of all the nodes in the cluster.
- **vSphere Cluster** - Select the default vSphere cluster on which the Master and Worker nodes are deployed.
- **Resource Pool** - Select the default resource pool on which the Master and Worker nodes are deployed.
- **VM Folder** - Select the virtual machine folder on which the Master and Worker nodes are placed.
- **Datastore** - Select the default datastore for the Master and Worker nodes to use.
- **Domain Name Servers** - Enter a valid DNS IP address. These DNS servers are configured in the guest operating system of each node in the cluster. You can override this option on the Master node and each node pool of the Worker node. To add a DNS, click **Add**.

8 Click **Next**.

9 In the **Master Node Configuration** tab, provide the following details:

- **vSphere Cluster** (Optional) - If you want to use a different vSphere Cluster for the Master node, select the vSphere cluster from here.
- **Resource Pool** (Optional) - If you want to use a different resource pool for the master node, select the resource pool from here.
- **Datastore** (Optional) - If you want to use a different datastore for the master node, select the datastore from here.
- **Network** - Associate a management or a private network. Ensure that the management network connects to a network where DHCP is enabled, and can access the VMware Photon repository.
- **Domain Name Servers** - You can override the DNS. To add a DNS, click **Add**.

10 Click **Next**.

11 In the **Worker Node Configuration** tab, provide the following details:

- **vSphere Cluster** (Optional) - If you want to use a different vSphere Cluster for the worker node, select the vSphere cluster from here.
- **Resource Pool** (Optional) - If you want to use a different resource pool for the worker node, select the resource pool from here.
- **Datastore** (Optional) - If you want to use a different datastore for the worker node, select the datastore from here.

- **Network** - Associate a management or a private network. Ensure that the management network can access the VMware Photon repository.
- **Domain Name Servers** - You can override the DNS. To add a DNS, click **Add**.

12 Click **Next** and review the configuration. You can download the configuration and reuse it for deploying a cluster with a similar configuration.

13 Click **Deploy**.

If the operation is successful, the cluster is created and its status changes to **Active**. If the operation fails, the cluster status changes to **Not Active**. If the cluster fails to create, delete the cluster, upload the previously downloaded configuration, and recreate it.

### Results

The Management cluster is deployed and VMware Telco Cloud Automation automatically pairs it with the cluster's site.

### What to do next

- You can view the Kubernetes clusters deployed through VMware Telco Cloud Automation from the **Kubernetes Cluster** tab.
- To view more details of the Kubernetes cluster that you have deployed, change the password, or to add syslog servers, go to **CaaS Infrastructure** > **Cluster Instances** and click the cluster.

## Deploy a Workload Cluster

Deploy a Workload cluster using the Kubernetes cluster template. You can deploy Workload clusters in parallel.

### Prerequisites

- You require a role with **Infrastructure Lifecycle Management** privileges.
- You must have uploaded the Virtual Machine template to VMware Telco Cloud Automation.
- You must have onboarded a vSphere VIM.
- You must have created a Management cluster or uploaded a Workload cluster template.
- A network must be present with a DHCP range and static IP of the same subnet.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Go to **CaaS Infrastructure** and click **Deploy Kubernetes Cluster**.

- If you have saved a validated Workload cluster configuration that you want to replicate on this cluster, click **Upload** on the top-right corner and upload the JSON file. The fields are then auto-populated with this configuration information and you can edit them as required.

- If you want to create a Workload cluster configuration from the beginning, perform the next steps.

3 Select a cloud on which you want to deploy the Kubernetes cluster.

4 Click **Next**.

5 The **Select Cluster Template** tab displays the available Kubernetes clusters. Select the Workload Kubernetes cluster template that you have created.

**Note** If the template displays as **Not Compatible**, edit the template and try again.

6 Click **Next**.

7 In the **Kubernetes Cluster Details** tab, provide the following details:

- **Name** - Enter the cluster name. The cluster name must be compliant with DNS hostname requirements as outlined in RFC-952 and amended in RFC-1123.
- **Description** (Optional) - Enter an optional description of the cluster.
- **Management Cluster** - Select the Management cluster from the drop-down menu. You can also select a Management cluster that is deployed in a different vCenter.
- **Password** - Create a password to log in to the Master and Worker nodes. The default user name is *capv*.

  **Note** Ensure that the password meets the minimum requirements displayed in the UI.
- **Confirm Password** - Confirm the password that you have entered.
- **OS Image With Kubernetes** - The pop-up menu displays the OS image templates in your vSphere instance that meet the criteria to be used as a Tanzu Kubernetes Grid base OS image with the selected Kubernetes version. If there are no templates, ensure that you upload the them to your vSphere environment.
- **Virtual IP Address** - VMware Tanzu Kubernetes Grid 1.2 deploys a `kube-vip` pod that provides load-balancing services to the cluster API server. This `kube-vip` pod uses a static virtual IP address to load-balance API requests across multiple nodes. Assign an IP address that is not within your DHCP range, but in the same subnet as your DHCP range.
- **Syslog Servers** - Add the syslog server IP address/FQDN for capturing the infrastructure logs of all the nodes in the cluster.
- **vSphere Cluster** - Select the default vSphere cluster on which the Master and Worker nodes are deployed.
- **Resource Pool** - Select the default resource pool on which the Master and Worker nodes are deployed.
- **VM Folder** - Select the virtual machine folder on which the Master and Worker nodes are placed.
- **Datastore** - Select the default datastore for the Master and Worker nodes to use.

- **Domain Name Servers** - Enter a valid DNS IP address. These DNS servers are configured in the guest operating system of each node in the cluster. You can override this option on the Master node and each node pool of the Worker node. To add a DNS, click **Add**.
- **Harbor Repository** - If you have defined a Harbor repository as a part of your Partner system, select the Harbor repository. The Harbor repository details are configured on all Master and Worker nodes.
- **NFS Client** - Enter the server IP address and the mount path of the NFS client. Ensure that the NFS server is reachable from the cluster. The mount path must also be accessible to read and write.
- **vSphere CSI Datastore** (Optional) - Select the vSphere CSI datastore. This datastore must be accessible from all the nodes in the cluster. If you do not select a datastore, the datastore used by the Master node is selected.

8. Click **Next**.
9. In the **Master Node Configuration** tab, provide the following details:
   - **vSphere Cluster** (Optional) - If you want to use a different vSphere Cluster for the master node, select the vSphere cluster from here.
   - **Resource Pool** (Optional) - If you want to use a different resource pool for the master node, select the resource pool from here.
   - **Datastore** (Optional) - If you want to use a different datastore for the master node, select the datastore from here.
   - **Network** - Associate a management or a private network. Ensure that the management network connects to a network where DHCP is enabled, and can access the VMware Photon repository.
   - **Domain Name Servers** - You can override the DNS. To add a DNS, click **Add**.
10. Click **Next**.
11. In the **Worker Node Configuration** tab, provide the following details for each node pool defined in the template:
    - **vSphere Cluster** (Optional) - If you want to use a different vSphere Cluster for the worker node, select the vSphere cluster from here.
    - **Resource Pool** (Optional) - If you want to use a different resource pool for the worker node, select the resource pool from here.
    - **Datastore** (Optional) - If you want to use a different datastore for the worker node, select the datastore from here.
    - **Network** - Associate a management or a private network. Ensure that the management network connects to a network where DHCP is enabled, and can access the VMware Photon repository.

12 Click **Next** and review the configuration. You can download the configuration and reuse it for deploying a cluster with a similar configuration.

13 Click **Deploy**.

    If the operation is successful, the cluster is created and its status changes to **Active**. If the operation fails, the cluster status changes to **Not Active**. If the cluster fails to create, delete the cluster, upload the previously downloaded configuration, and recreate it.

#### Results

The Workload cluster is deployed and VMware Telco Cloud Automation automatically pairs it with the cluster's site.

#### What to do next

- You can view the Kubernetes clusters deployed through VMware Telco Cloud Automation from the **Kubernetes Cluster** tab.
- To view more details of the Kubernetes cluster that you have deployed, go to **CaaS Infrastructure** > **Cluster Instances** and click the cluster.

# Viewing Cluster Details

After a Kubernetes cluster is deployed, it is listed under **CaaS Infrastructure** > **Cluster Instances**. To view more information about the Kubernetes cluster, click it.

## Cluster Details

The **Cluster Details** tab provides the following information:

- **Cluster Type** - Management or a Workload cluster.
- **Cluster URL** - The URL of the cluster API server.
- **Cluster Username** - User name to access the cluster.
- **vSphere Cluster Name** - The name of the selected vSphere cluster.
- **Management Cluster** - The backing Management cluster name.
- **Cluster Template** - The backing cluster template name.

## Cluster Configuration

The **Cluster Configuration** tab displays information about the Kubernetes version of the cluster, upgrade history, its CNI and CSI configurations, any tools such as Helm associated with the cluster, syslog server details, and Harbor repository details. To edit any of the configuration information, click **Edit**.

### Master Nodes

The **Master Nodes** tab displays the details about the Master node, labels attached to the node, and network labels. It also displays the IP addresses and names of the VMs that are deployed for the Master node. To increase or decrease the replica count and to add labels, click **Edit**.

### Worker Nodes

The **Worker Nodes** tab displays the existing node pools of a Kubernetes cluster. To view more details of the node pool such its name, CPU size, memory size, storage size, number of replicas, node customization details, and its status, click the **>** icon against the node pool. You can also add a node pool to the cluster, edit the number of replicas on a node pool, and delete a node pool from here.

### Tasks

The **Tasks** tab displays the progress of the cluster-level tasks and their status.

- **Management Cluster** - Displays the progress of Management cluster tasks along with the progress of all the Workload cluster tasks that are managed by this cluster. It also displays the node pool tasks of all the Workload clusters.
- **Workload Cluster** - Displays the progress of the Workload cluster tasks along with the progress of its node pool tasks.

You can apply filters to view the progress of specific operations and specific clusters.

# Editing Kubernetes Clusters

After you deploy a Kubernetes cluster, you can edit its cluster configuration, edit its Master and Worker node configurations, upgrade the Kubernetes version, and change the Kubernetes password.

## Edit a Kubernetes Cluster Configuration

You can add a Container Network Interface (CNI) such as *Multus* to the existing list of CNIs, update the Container Storage Interface (CSI) timeout duration, and toggle the default storage class type. You can add a tool such as Helm to manage your Kubernetes applications and update its version. You can also add or update the syslog servers to redirect the infrastructure logs of the Master and Worker nodes, and update the Harbor repository details.

#### Prerequisites

Ensure that the cluster is not running any operations.

#### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **CaaS Infrastructure** > **Cluster Instances** tab.

3 Click the **Options** ( ⁝ ) symbol against the Kubernetes cluster that you want to edit.

4 Click **Edit Cluster Configuration**.

5 In the **Cluster Configuration** tab, add a CNI, CSI, tool, or syslog server, and click **Save**.

   **Note** In a Workload cluster:

   - You cannot edit the Storage Class name in the vSphere-CSI (NFS Client is also not supported).
   - You can add CNI, CSI, or Tools, but cannot remove them.

Results

You have successfully edited the cluster configuration.

## Edit a Kubernetes Cluster Master Node Configuration

You can scale up or scale down the number of Master node replicas and add or remove labels.

Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Go to **CaaS Infrastructure** > **Cluster Instance** tab.

3 Click the **Options** ( ⁝ ) symbol against the Kubernetes cluster that you want to edit.

4 Click **Edit Master Node Configuration**.

5 In the **Master Nodes** tab, scale down or scale up the Worker nodes, add or remove labels, and click **Save**.

Results

You have successfully edited the Master node configuration of your Kubernetes cluster instance.

## Edit a Kubernetes Cluster Node Pool

You can scale up or scale down the number of Worker nodes in each node pool and add labels. If a network function with infrastructure requirements is running in this node pool, the scaling up operation automatically applies all the node customizations on the new nodes.

Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Go to **CaaS Infrastructure** > **Cluster Instance** tab.

3 Click the **Options** ( ⁝ ) symbol against the Kubernetes cluster that you want to edit.

4 Click **Edit Worker Node Configuration**.

5 In the **Worker Nodes** tab, select the node pool that you want to edit, and click **Edit**.

6 Scale down or scale up the Worker nodes and add labels.

7. Click **Update**.

### Results

You have successfully edited the Worker node configuration of a Kubernetes cluster instance in your node pool.

## Add a Node Pool

You can add a node pool to your Kubernetes Workload cluster.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **CaaS Infrastructure** > **Cluster Instance** tab.
3. Click the **Options** ( ⋮ ) symbol against the Kubernetes cluster where you want to add the node pool.
4. Click **Edit Worker Node Configuration** and click **Add**.

   In the Add Node Pool window, enter the following information:

   - **Name** - Enter the name of the node pool.
   - **CPU** - Select the number of vCPUs to have in the node pool.
   - **Memory** - Select the amount of memory for the node pool.
   - **Storage** - Select the storage size.
   - **Replica** - Select the number of controller node VMs to be created.
   - **vSphere Cluster** (Optional) - If you want to use a different vSphere Cluster, select the vSphere cluster from here.
   - **Resource Pool** (Optional) - If you want to use a different resource pool, select the resource pool from here.
   - **Datastore** (Optional) - If you want to use a different datastore, select the datastore from here.
   - **Labels** - Add key-value pair labels to your nodes. You can use these labels as node selectors when instantiating a network function.
   - **Networks** - Enter the labels to group the networks. Networks use these labels to provide network inputs during a cluster deployment. Add additional labels for network types such as IPvlan, MacVLAN, and Host-Device. Meaningful network labels such as N1, N2, N3, and so on, help users provide the correct network preferences during deployment. It is mandatory to include a management interface label. SR-IOV interfaces are added to the Worker nodes when deploying the network functions.

     **Note** A label length must not exceed 15 characters.

    Apart from the management network, which is always the first network, the other labels are used as interface names inside the Worker nodes. For example, when you deploy a cluster using the template with the labels **MANAGEMENT**, **N1**, and **N2**, the Worker nodes interface names are **eth0**, **N1**, **N2**. To add more labels, click **Add**.

    - **Domain Name Servers** - Enter a valid DNS IP address. These DNS servers are configured in the guest operating system of each node in the cluster. You can override this option on the Master node and each node pool of the Worker node. To add a DNS, click **Add**.
    - **CPU Manager Policy** - Set CPU reservations on the Worker nodes as **Static** or **Default**. For information about controlling CPU Management Policies on the nodes, see the Kubernetes documentation at https://kubernetes.io/docs/tasks/administer-cluster/cpu-management-policies/.

      **Note** For CPU-intensive workloads, use **Static** as the CPU Manager Policy.

    - **Advanced Configuration** - To use the vSphere Linked Clone feature for creating linked clones for the Kubernetes nodes, select **Use Linked Cloning for Cloning the VMs**.

5. Click **Add**.

### Results

You have successfully added the node pool to your Workload cluster.

## Delete a Node Pool

Delete a node pool from the Kubernetes Workload cluster.

### Prerequisites

Ensure that the node pool is not running any applications.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **CaaS Infrastructure** > **Cluster Instance** tab.
3. Click the **Options** ( ⋮ ) symbol against the Kubernetes cluster and select **Edit Worker Node Configuration**.
4. Select the node pool, click **Delete**, and confirm the operation.

### Results

You have successfully deleted the node pool.

## Upgrade Kubernetes Version

You can upgrade the existing Kubernetes version to the latest versions of Kubernetes supported in the current version of the VMware Telco Cloud Automation.

You can upgrade the Kubernetes cluster through VMware Telco Cloud Automation.

The following table lists the Kubernetes upgrade compatibility for the Workload cluster.

| Existing Kubernetes Versions | 1.17.9 | 1.17.11 | 1.18.8 | 1.19.1 |
|---|---|---|---|---|
| 1.17.3 | Yes | No | Yes | No |
| 1.17.9 | No | No | Yes | No |
| 1.17.11 | No | No | Yes | No |
| 1.18.2 | No | No | Yes | Yes |
| 1.18.6 | No | No | Yes | Yes |
| 1.18.8 | No | No | No | Yes |

Before upgrading Kubernetes to the latest version, consider the following constraints and prepare for the upgrade plan:

- Upgrade the node pools created through VMware Telco Cloud Automation 1.7 before upgrading the Kubernetes version for Workload clusters.
- Ensure that all management clusters and Workload clusters nodes are up and reachable through TCA-CP.

  **Note** You can run `kubectl get nodes` on each Management cluster and Workload cluster. The result shows the IP address each node under Management and Workload cluster.

- VMware Telco Cloud Automation preserves the customization performed through previous CNF instantiate / upgrade on the nodepools of the cluster. Any manual changes performed directly on the nodes are not preserved.
- Applications may face downtime during kubernetes upgrade and may take some time to be available for operations.
- Clusters may take some time to be available for operations.
- Check and upgrade the required node pools in the Workload cluster.
- The IP address of master nodes and the worker nodes changes after upgrade.
- All the customization added to the cluster through VMware Telco Cloud Automation are applied after upgrade.
- If the upgrade fails, you can correct the configuration and perform the upgrade again.

## Upgrade the Kubernetes Version for Cluster Template

You can upgrade the existing version of the Kubernetes version to the latest Kubernetes version.

See Edit a Kubernetes Cluster Template to upgrade the Kubernetes version of the cluster template.

## Upgrade the Kubernetes Version for a Cluster Instance

You can upgrade the existing version of Kubernetes to the latest Kubernetes.

### Prerequisites

- Create an upgrade plan for the upgrading the cluster instance, considering the impact of cluster downtime.
- Take backup of any manual customization added to the clusters. You must take the backup manually.

  **Note** You need to note down all the manual customization added to the clusters.

- Upgrade the node pool required for Workload cluster. For details, see Upgrading the Node Pool.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **Caas Infrastructure**.

   The **CaaS Infrastructure** page is displayed.
3. Select the cluster instance for upgrade.
4. Click the **Options** ( ⋮ ) symbol against the Kubernetes cluster that you want to upgrade.
5. Select **Upgrade Kubernetes**.

   The **Upgrade Kubernetes** window is displayed.
6. In the **Select Version** field, select the Kubernetes version to upgrade from the list.
7. In the **Virtual Machine Template**, click the option to select the VM template applicable for the new version of Kubernetes.
8. Click **Upgrade**.

   The upgrade process starts.
9. Click **>** to view the progress of the update.

### What to do next

To get the latest IP address details of the node, view the **Cluster Instances** page.

## Change the Kubernetes Password

You can change the password that you had set when deploying your Kubernetes cluster.

### Prerequisites

Ensure that your password meets the minimum security requirements listed in the interface.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **CaaS Infrastructure** > **Cluster Instance** tab.

3 Click the **Options** ( ⋮ ) symbol against the Kubernetes cluster that you want to change the password for.

4 Click **Change Password**.

5 In the **Change Password** pop-up window, enter your new password and confirm it.

6 Click **Change Password**.

### Results

You have successfully changed the password of your Kubernetes cluster.

**Note** It might take some time for the system to reflect the changed password.

## Upgrading the Node Pool

You must upgrade the node pool under the Worker nodes of the Workload clusters created in VMware Telco Cloud Automation 1.7.

Upgrade all the node pools under workload clusters before upgrading the Kubernetes cluster. If not upgraded, you cannot edit the node pools or instantiate CNF. Consider the following points before upgrading the node pool:

- Node pool upgrade is a one time operation required when upgrading from VMware Telco Cloud Automation 1.7 to VMware Telco Cloud Automation 1.8.
- There could be an application down-time during the upgrade.
- You can retry the failed node pool upgrade operations after making the required changes.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Go to **Caas Infrastructure**.

The **CaaS Infrastructure** page is displayed.

3 Select the Workload cluster you must upgrade.

4 Click the **Options** ( ⋮ ) symbol against the Workload cluster that you want to upgrade.

5 Select **Edit Worker Node Configuration**.

The **Worker Node Configuration** window is displayed.

6 Select the node pool to upgrade.

7 Click **Upgrade**.

The upgrade process starts.

**Note** You can also upgrade multiple node pools. Click **Upgrade** and select the node pools to upgrade.

8 Click **>** to view the progress of the upgrade.

# Managing Network Function Catalogs

9

A network function, as defined by ETSI Industry Specification Group (ISG), is a functional building block within a network infrastructure. It has well-defined external interfaces and a well-defined functional behavior.

This chapter includes the following topics:

## Onboarding a Network Function

A network function descriptor describes the instantiation parameters and operational behaviors of the VNFs. It contains key requirements for onboarding and managing the life cycle of a VNF. Onboarding a network function includes uploading a network function package to the catalog, and creating or editing a network function descriptor draft.

### Upload a Network Function Package

Using VMware Telco Cloud Automation, you can upload a SOL001/SOL004 compliant Virtual Network Function Descriptor (VNFD) and Cloud Service Archive (CSAR) package. The system parses and validates the configuration, and presents the topology in a visual viewer. It then persists the entry into the Network Function Catalog.

**Prerequisites**

- Verify that your VNFD complies with the following standards:
  - Must be in the CSAR format.
  - Must comply with the SOL001 or SOL004 standard.
  - Must comply with TOSCA Simple Profile in YAML version 1.2 or TOSCA Simple Profile for NFV version 1.0.

**Procedure**

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Catalog** and click **Onboard**.

   The Onboard Network Function page is displayed.

3 Select **Upload Network Function Package**.

4 Enter a name for your network function.

5 Add the associated tags for your Network Function Package. You can add more than one tag.

6 Click **Browse** and select the network function descriptor (CSAR) file.

7 Click **Upload**.

#### Results

The specified network function is added to the catalog. You can now instantiate the function or use it to create a network service.

#### What to do next

- To instantiate the network function, see Instantiate a Virtual Network Function.
- To create a network service that includes the network function, see Design a Network Service Descriptor.
- To obtain the CSAR file corresponding to a network function, select the function in the catalog and click **Download**.
- To add or remove tags for your network function, select the desired network function and click the **Edit** icon.
- To remove a network function from the catalog, stop and delete all instances using the network function. Then select the function in the catalog and click **Delete**.

## Designing a Network Function Descriptor

You can create ETSI-compliant network functions using VMware Telco Cloud Automation. The Network Function Designer is a visual design tool within VMware Telco Cloud Automation that generates SOL001-compliant TOSCA descriptors based on your design.

### Design a Virtual Network Function Descriptor

Design a VNF descriptor.

#### Prerequisites

Add a cloud to your virtual infrastructure.

#### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Catalog** and click **Onboard**.

   The Onboard Network Function page is displayed.

3 Select **Design Network Function Descriptor**.

4 **Name** - Enter a unique name for your VNF descriptor.

5 **Tags (Optional)**- Enter the tags to associate your VNF descriptor with.

6 **Type** - Select the network function type as **Virtual Network Function**.

7 Click **Design**.

   The Network Function Designer is displayed.

8 In the **Network Function Properties** pane, enter the following information:

   - **Descriptor ID** - The descriptor ID is system generated.
   - **Descriptor Version** - Enter the descriptor version.
   - **Provider** - Enter the company name of the provider.
   - **Vendor** - Enter the company name of the vendor.
   - **Product Name** - Enter the product name of the descriptor.
   - **Version** - Enter the product version.
   - **Software Version** - Enter the software version.

9 In the **Available Operations** pane, select the life-cycle management operations to be made available for your VNF. Your users can run only those operations that are enabled here.

10 (Optional) Add one or more workflows to your network function.

   You can add custom workflows using vRealize Orchestrator. For information about adding custom workflows, see Chapter 14 Running Workflows with vRealize Orchestrator.

   a Click **Add Workflow** and select the desired workflow from the drop-down menu:

      - **Instantiate Start**
      - **Instantiate End**
      - **Heal Start**
      - **Heal End**
      - **Scale Start**
      - **Scale End**
      - **Scale To Level Start**
      - **Scale To Level End**
      - **Terminate Start**
      - **Terminate End**

    - **Custom**

  b. Click **Browse** and upload an instantiation script in the JSON format.
  c. Select **Manual Execution** if you want your users to run the workflows manually.
  d. Provide an optional description for your workflow.
  e. Enter any input and output variables specified in your script and select whether they are required.

11. Click **Update**.

    You can modify these settings later by clicking **Edit Network Function Catalog Properties** in the Network Function Designer.

12. Add internal networks (**Virtual Link**) to your VNF by dragging the icon from the toolbar into the design area. During Instantiation, Telco Cloud Automation creates networks for these virtual links. You can override them and select the existing networks if necessary.

13. To configure additional settings for your network, click the pencil icon against the network.

    You can configure the following settings:

    - **Description**
    - **Network Name**
    - **CIDR**
    - **DHCP**
    - (Optional) **Gateway IP**
    - (Optional) **IP Allocation Pools**
      - **Start IP Address**
      - **End IP Address**

    When you finish configuring the settings, click **Update**.

14. Add virtual machines (**VDU**) by dragging the icon from the Toolbar into the design area.

15. In the Configure VDU pane, specify the following settings for each VDU:

    - **Name** - Name of the VDU.
    - **Description** - Description about the VDU.
    - **Minimum Instances** - The minimum number of VDU instances.
    - **Maximum Instances** - The maximum number of VDU instances.

- **Image Name** - The name of the VM template that is on the backing vCenter Server of your cloud.

  **Note**

  - The image name you enter must match the virtual machine template name on the vCenter Server.
  - The image must be saved as a VM template.
- **Virtual CPU** - Number of virtual CPUs.
- **Virtual Memory** - Virtual memory size.
- **Virtual Storage** - Virtual storage size.
- **OVF Properties** (Optional) - OVF properties are the OVF inputs to provide to the VM template. Enter the property, description, type such as string, boolean, or number, and default value. To make this information mandatory, select the **Required** option.
- **Connection Points** - Select an internal or external connection point from the **Add Connection Point** drop-down menu:
  - **Internal Connection Point** - Links the VDU to an existing virtual link that is added to the VNF. At least one virtual link is required for internal connection points.
  - **External Connection Point** - Is a placeholder for an external virtual link that is required during instantiation. You must provide a **Connection Name** that matches with the external virtual link name.
- **Depends On** (Optional) - Specify the VDUs to be deployed before deploying this VDU. In a scenario where you deploy many VDUs, there can be dependencies between VDUs regarding the order in which they are deployed. This option enables you to specify their deployment order.

**Note** To enable the **Depends On** option, you must configure more than one VDU.

**Note** You must add at least one virtual link before configuring the internal connection points for your VDUs.

You can modify VDU settings at a later stage by clicking the pencil icon on the desired VDU.

16 To save your descriptor as a draft and work on it later, click **Save As Draft**. For information about working with different draft versions, see Edit Network Function Descriptor Drafts.

17 After designing your network function descriptor, click **Upload**.

### Results

The specified network function is added to the catalog. You can now instantiate the function or use it to create a network service.

### What to do next

- To instantiate the network function, see Instantiate a Virtual Network Function.
- To create a network service that includes the network function, see Design a Network Service Descriptor.
- To obtain the CSAR file corresponding to a network function, select the function in the catalog and click **Download**.
- To add or remove tags, go to **Network Functions** > **Catalog** and click the desired network function. Then click **Edit**.
- To remove a network function from the catalog, stop and delete all instances using the network function. Then select the function in the catalog and click **Delete**.

## Design a Cloud Native Network Function Descriptor

Design a CNF descriptor.

### Prerequisites

Add a cloud to your virtual infrastructure.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Catalog** and click **Onboard**.

   The Onboard Network Function page is displayed.
3. Select **Design Network Function Descriptor**.
4. **Name** - Enter a unique name for your VNF descriptor.
5. **Tags (Optional)**- Enter the tags to associate your VNF descriptor with.
6. **Type** - Select the network function type as **Cloud Native Network Function**.
7. Click **Design**.

   The Network Function Designer is displayed.
8. In the **Network Function Catalog Properties** pane, enter the following information:

   - **Descriptor Version** - Enter the descriptor version.
   - **Provider** - Enter the company name of the provider.
   - **Vendor** - Enter the company name of the vendor.
   - **Product Name** - Enter the product name of the descriptor.
   - **Version** - Enter the product version.
   - **Software Version** - Enter the software version.

9 In the **Available Operations** pane, select the life-cycle management operations to be made available for your CNF. Your users can run only those operations that are enabled here.

10 (Optional) Add one or more workflows to your network function.

You can add custom workflows using vRealize Orchestrator. For information about adding custom workflows, see Chapter 14 Running Workflows with vRealize Orchestrator.

a Click **Add Workflow** and select the desired workflow from the drop-down menu:

- **Instantiate Start**
- **Instantiate End**
- **Terminate Start**
- **Terminate End**
- **Custom**

b Click **Browse** and upload an instantiation script in the JSON format.

c Select **Manual Execution** if you want your users to run the workflows manually.

d Enter any input and output variables specified in your script and select whether they are required.

11 Click **Update**.

You can modify these settings later by clicking **Edit Network Function Catalog Properties** in the Network Function Designer.

12 From the **Components** toolbar, drag a Helm Chart into the design area. Helm is a Kubernetes application manager used for deploying CNFs. Helm Charts contain a collection of files that describe a set of Kubernetes resources. Helm uses the resources from Helm Charts to orchestrate the deployment of CNFs on a Kubernetes cluster.

13 In the **Configure Helm** window, enter the following details:

- **Name** - Name of the Helm.
- **Description** - A brief description about the Helm.
- **Chart Name** - Name of the chart from the Helm repository.
- **Chart Version** - Version number of the chart from the Helm repository.
- **Helm Version** - Select the version of the Helm from the drop-down menu.
- **ID** - Enter the Helm ID.
- **Helm Property Overrides** (Optional) - Add additional instantiation properties to override or add a YAML file that contains a list of properties to override. To upload a YAML file, enter the filename in the **Property** text box and select the **Type** as **File**. You must upload the YAML file during instantiation.

   - **Depends On** (Optional) - Specify the Helm to be deployed before deploying this Helm. In a scenario where you deploy many Helms, there can be dependencies between the Helms regarding the order in which they are deployed. This option enables you to specify their deployment order.

14. Click **Update**.
15. To save your descriptor as a draft and work on it later, click **Save As Draft**. For information about working with different draft versions, see Edit Network Function Descriptor Drafts.
16. After designing your network function descriptor, click **Upload**.

#### Results

The specified network function is added to the catalog. You can now instantiate the function or use it to create a network service.

#### What to do next

- To instantiate the network function, see Instantiate a Virtual Network Function.
- To create a network service that includes the network function, see Design a Network Service Descriptor.
- To obtain the CSAR file corresponding to a network function, select the function in the catalog and click **Download**.
- To add or remove tags, go to **Network Functions** > **Catalog** and click the desired network function. Then click **Edit**.
- To remove a network function from the catalog, stop and delete all instances using the network function. Then select the function in the catalog and click **Delete**.

## Working with Affinity Rules

Affinity Rules govern the hosting of the VMs on a host.

A VM-VM affinity rule specifies whether selected individual virtual machines run on the same host or kept on separate hosts. This type of rule is used to create affinity or anti-affinity between individual virtual machines that you select.

An affinity rule ensures that the specified virtual machines are placed together on the same host. An anti-affinity rule ensures that the specified virtual machines do not share the host. You can create an anti-affinity rule to guarantee that certain virtual machines are always on different physical hosts. This way, not all virtual machines are at risk when one of the hosts encounters an issue.

### Create Affinity Rules

You can create VM-VM affinity rules to specify whether selected individual virtual machines run on the same host or kept on separate hosts.

### Prerequisites

Create the topology.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Catalog** and click **Onboard**.

   The Network Function Catalog page is displayed.
3. Click the network function on which you want to create affinity rules and select the **Policies** tab.
4. To add an affinity rule, click the **Add** under **Affinity Rules**.
   a. Add the name of the affinity rule in text box corresponding to **Rule Name**.
   b. To create affinity among the VDUs, select the VDU from the list.
5. To add an anti-affinity rule, click the **Add** under **Anti-Affinity Rules**.
   a. Add the name of the anti-affinity rule in text box corresponding to **Rule Name**.
   b. To create an anti-affinity rule among the VDUs, select the VDU from the list.

### Results

The affinity and anti-affinity rules are added.

### Example

Table 9-1. Affinity and Anti Affinity Rules

| VDU | Affinity Rules | Anti-Affinity Rules |
|---|---|---|
| VDU 1, VDU 2 | The deployed VDUs are always kept together on the same ESXi host even for scaled-out instances. | The deployed VDUs are always kept apart on different ESXi hosts. for scaled-out instances, an anti-affinity rule is created for every permutation and combination. |
| VDU 1 | All the scaled VDU instances of VDU 1 are kept together on the same ESXi host. | All the scaled VDU instances of VDU 1 are kept apart on different ESXi hosts and only one anti-affinity rule is created. |

## Edit Network Function Descriptor Drafts

If you have saved a draft in the Network Function Designer, you can modify or delete it at a later stage. The draft is saved as a new version every time you save. When you want to edit, select the version of draft to edit.

### Prerequisites

Use the Network Function Designer to create a network function descriptor and save the design as a draft.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Catalog** and click **Onboard**.

The Onboard Network Function page is displayed.

3 Select **Edit Network Function Descriptor Drafts**.

4 From the table, locate the desired network descriptor draft to edit.

The Network Function Designer page is displayed.

5 To select the draft version, click the page icon on the right side of the Network Function Designer page. You can restore a previous version from here.

6 To modify the draft, click the **Edit** (pencil) icon. To remove the draft, click the **Delete** icon.

## Edit the Network Function Catalog Source Files

You can edit the source files of a Network function catalog and update it, or save the catalog as a new version.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Catalog**.

3 To edit the source files, click the desired Network Function catalog and select the **Source** tab.

4 To save the changes and work on the source files later, click **Save as Draft**.

5 To apply the changes to the current version, click **Update Catalog**.

6 To save the catalog as a new version, click **Save As New Catalog**.

### Results

Changes to the Network Function catalog are saved appropriately.

## Edit a CSAR File Manually

You can manually edit a CSAR file and upload it to VMware Telco Cloud Automation.

### Prerequisites

**Note** The following steps are valid only on macOS and Linux operating systems. On a Windows operating system, use the relevant commands to edit the CSAR file.

### Procedure

1. Download the CSAR file that you want to edit. For more information, see Download a Network Function Package.
2. Unzip the CSAR file.
3. Go to the **Definitions** folder and open the `NFD.yaml` file.
4. Update the `descriptor_id` field with the new descriptor ID. You can also update the `NFD.yaml` with any other changes, as appropriate.
5. Save the `NFD.yaml` file.
6. You can also add any other supporting files to their respective folders or edit the existing files.

   For example, you can add a script to the **Artifacts** > **scripts** folder.
7. Recreate the CSAR file. Run the following command:

   ```
   zip -r <new_name>.csar TOSCA-Metadata/ Definitions/ Artifacts/ NFD.mf
   ```
8. Upload the CSAR file VMware Telco Cloud Automation. For more information, see Upload a Network Function Package.

## Delete a Network Function

You can delete a network function from the catalog.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Catalog**.
3. Select the desired network function and click **Delete**.
4. Confirm the action by clicking **OK**.

### Results

The network function is removed from the catalog.

# Customizing Network Function Infrastructure Requirements

You can customize a CNF's infrastructure according to its unique requirements. Customizing the infrastructure requirements enables you to create a cluster, instantiate, and deploy the network functions without any manual user inputs.

Network functions from different vendors have their own unique set of infrastructure requirements. Defining these requirements in the network functions ensure that they are instantiated and deployed in a cluster without you having to log in to their master or worker nodes.

You can customize a network function's infrastructure requirements from the **Network Functions** > **Catalogs** tab. Click the network function that you want to customize and select the **Source** tab.

A new keyword called `infra_requirements` is introduced. Here, you can define the node, Containers as a Service (CaaS), and Platform as a Service (PaaS) components:

1. Under `node_components`, define the Linux kernel type and kernel arguments for each worker node. You can also define any custom packages to be installed on your nodes here.
2. Under `caas_components`, define the CaaS components such as CNIs to be installed on each worker node. A default version of *Multus* (version 1.16) is installed on all the worker nodes in the cluster.
3. Under `network`, define the network name and the allocatable network resource name for the CaaS components.

After you define the components of `infra_requirements` in the CNF catalog, the cluster is customized according to the differences detected between the CNF catalog and the actual configuration present in the cluster during instantiation.

## Node Customization

You can customize nodepools of the clusters using network function catalog defined in a TOSCA (Topology and Orchestration Specification for Cloud Applications) file.

VMware Telco Cloud Automation uses Network Function TOSCA (Topology and Orchestration Specification for Cloud Applications) extensions to determine the requirements for different VIMs.

Features enabled through TOSCA extensions include:

- SRIOV Interface addition and configuration
- NUMA alignment of vCPUs and VF/PFs
- Latency sensitivity
- Tuned profile
- DPDK binding for SRIOV interfaces
- Kernel Update
- Kernel Modules
- Custom package installations (pciutils, lxcfs.)
- GRUB config (all configurations used for the CPU isolation, hugepages config.)
- Passthrough devices for PTP

### TOSCA Components

You can modify the node components and CaaS components in TOSCA for different Kubernetes VIMs.

To support various network functions, the Worker nodes may require a customization in the TOSCA. These customizations include the kernel-related changes, custom packages installations, network adapter, SRIOV, DPDK configurations, and CPU Pinning of the Worker nodes on which you deploy the network functions.

### Node Components

- Kernel: The Kernel definition uses multiple arguments that require a customization.
  - `kernel_type`: Kernel type for the worker nodes. The kernel types are:
    - Linux RealTime (linux-rt)
    - Linux Non-RealTime (linux)

    The kernel type depends on the network function workload requirement. The required Linux version is downloaded from `TDNF repo[VMware Photon Linux]` during customization.
    **kernel type**

    ```
    infra_requirements:
      node_components:
        kernel:
          kernel_type:
            name: linux-rt
            version: 4.19.132-1.ph3
    ```
  - `kernel_args`: Kernel boot parameters for tuning values that you can adjust when the system is running. These parameters configure the behavior of the kernel such as isolating CPUs. These parameters are free form strings. They are defined as 'key' → name of the parameter and optionally 'value' → if any arguments are provided.
    **kernel_args**

    ```
    infra_requirements:
      node_components:
        kernel:
          kernel_args:
            - key: nosoftlockup
            - key: noswap
            - key: softlockup_panic
              value: 0
            - key: pcie_aspm.policy
              value: performance
            - key: intel_idle.max_cstate
              value: 1
            - key: mce
              value: ignore_ce
            - key: fsck.mode
              value: force
    ```

### Huge Pages

```
infra_requirements:
  node_components:
    kernel:
      kernel_args:
        - key: default_hugepagesz
          value: 1G
        - key: hugepagesz
          value: 1G
        - key: hugepages
          value: 17
Note:
i.   This order should be maintained.
ii.  Nodes will be restarted to set these values
iii. supported hugepagesz are 2M | 1G
```

### isolcpus

```
infra_requirements:
  node_components:
    kernel:
      kernel_args:
        - key: isolcpus
          value: 2-{{tca.node.vmNumCPUs}}
Note: TCA will replace the {{tca.node.vmNumCPUs}} with vCPUs configured on the worker node.
```

- `kernel_modules`: To install any kernel modules on Worker nodes. For example, `dpdk`, `sctp`, and `vrf`.

  **Note** When configuring `dpdk`, ensure that the corresponding `pciutils` package is specified under `custom_packages`.

  ### dpdk

  ```
  infra_requirements:
    node_components:
      kernel:
        kernel_modules:
          - name: dpdk
            version: 19.11.1
  ```

  For a details on supported DPDK versions, see Supported DPDK and Kernel Versions.

- `custom_packages`: Custom packages include the `lxcfs`, `tuned`, `pci-utils`, `ptp`. The required packages are downloaded from `TDNF repo[VMware Photon Linux]` during customization.
  **custom_packages**

  ```
  infra_requirements:
    node_components:
      custom_packages:
         - name: pciutils
           version: 3.6.2-1.ph3
  ```

```
       - name: tuned
         version: 2.13.0-3.ph3
       - name: linuxptp
         version: 2.0-1.ph3
Note: Make sure these packages are available on VMWARE TDNF Repository
```

- `additional_config`: Helps in the additional customization on node. For example, `tuned`.

  **Note** While configuring `tuned`, ensure that the corresponding tuned package is specified under `custom_packages`

  **tuned**

  ```
  infra_requirements:
    node_components:
      additional_config:
        - name: tuned   # <--- for setting tuned
          value: '[{"name":"custom-profile"}]'    # <--- list of profile names to activate.
  ```

- `file_injection`: Inject the configuration files inside the nodes.
  **file_injection**

  ```
  infra_requirements:
    node_components:
      file_injection:

        - source: file
          content: ../Artifacts/scripts/custom-tuned-profile.txt
   #<-- File path location which is embedded in CSAR
          path: /etc/tuned/custom-profile/tuned.conf  #<-- Target location of the configuration
  file. Location should align with name of the profile.
        - source: file

          content: ../Artifacts/scripts/cpu-partitioning-variables.txt
   #<-- File path location which is embedded in CSAR
          path: /etc/tuned/cpu-partitioning-variables.conf #<-- Supporting files for the main
  configuration file.
  ```

- `isNumaConfigNeeded`: This feature tries to find a host and a NUMA node that can fit the VM with the given requirements and assign it. It is useful for high-performance profile Network Functions such as `DU`, which require a high throughput. This sets CPU and Memory reservations to maximum on the Worker node. It sets the affinity for the Worker node `cpus` to the ESXi `cpus`.
  **isNumaConfigNeeded**

  ```
  infra_requirements:
     node_components:
       isNumaConfigNeeded:  [true | false]
  ```

- `latency_sensitivity`: For Network Functions that require a high-performance profile with low-latency such as `DU`, `CU-CP`, `CU-UP`, and `UPF`. These functions require the node latency sensitivity set on vSphere.

  **Note** Node restarts after customization.

  **latency_sensitivity**

  ```
  infra_requirements:
    node_components:
      latency_sensitivity:
  [high | low]
  ```

- `passthrough_devices`: For adding PCI devices. For example, `ptp`.

  **Note** While specifying passthrough device configurations, ensure that the corresponding `linuxptp` package is specified under `custom_packages`.

  **passthrough_devices**

  ```
   infra_requirements:
     node_components:
       passthrough_devices:
         - device_type: NIC
           pf_group: ptp
  Note: For now the values are hardcoded
  ```

- `network`: Creates network adapters on the nodes. For SRIOV, the given resource name will be allocatable resource on the node.
  **Network**

  ```
   infra_requirements:
     node_components:
       network:
          devices:
            - deviceType:    # <-- Network Adapter type [sriov]
              networkName:    # <-- Input label for User Input to provide Network while NF 
  Instantiation. Refer below section how to define these input
              resourceName:  # <-- This is the label the device will be exposed in K8s node.
              dpdkBinding:   # <-- The driver this device should used. If not mentioned, then 
  default OS driver will be used.
              count: 3       # <- Number of adapters required.
  Note:
      1. vmxnet3 not supported in TCA
      2. for 'networkName' refer below section
      3. dpdkBinding
         - igb_uio
         - vfio-pci
      4. Make sure to have 'pciutils' custom packages and 'dpdk' kernel modules.
  ```

For SRIOV network adapters, when initiating, add the following:

### VnfAdditionalConfigurableProperties

```
tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.lmn:
  derived_from: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties
  properties:
    F1U: # <--- label that is provided infra_requirements.node_components.network.devices.networkName
      required: true
      propertyName: F1U # <--- label that is provided 
infra_requirements.node_components.network.devices.networkName
      description: ''
      default: ''
      type: string
      format: network        #<- to show the network drop-down menu

 helm-abc:
    type: tosca.nodes.nfv.Vdu.Compute.Helm.helm-abc
    properties:
      :
      configurable_properties:
        additional_vnfc_configurable_properties:
          type: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.lmn
          :
          :
          F1U: '' # <-- Same label provided above
```

### caas_components

You can configure CaaS components, such as `CNI`, `CSI`, `Helm` for the Kubernetes. You can install `CNI` plugins on Worker nodes during CNF instantiation. Provide CNIs such as `SRINOV` in Cluster Configuration in the CaaS Infrastructure.

```
infra_requirements:
  caas_components:
    - name: sriov
      type: cni
```

## TOSCA Definition Extension

VMware Telco Cloud Automation uses modified TOSCA, which is an extension of the standard TOSCA, to determine prerequisites for different VIMs.

The root node `tosca.nodes.nfv.VMware.VNF` defines the VNF definition like CaaS and NodeConfig related requirements in the TOSCA.

The `infra_requirements` property at the root node defines these infrastructure requirements for the Network Function.

The following sample shows the definition of a TOSCA file used by VMware Telco Cloud Automation.

### tosca.nodes.nfv.VNF

```
Definition of infra_requirements

tosca.nodes.nfv.VNF:
    derived_from: tosca.nodes.Root
    description: The generic abstract type from which all VNF specific abstract node types shall be
derived to form, together with other node types, the TOSCA service template(s) representing the VNFD
    properties:
      descriptor_id: # instead of nfd_id
        type: string # GUID
        description: Globally unique identifier of the VNFD
        required: true
      provider: # instead of nf_provider
        type: string
        description: Provider of the VNF and of the VNFD
        required: true
      product_name: # instead of nf_product_name
        type: string
        description: Human readable name for the VNF Product
        required: true
      software_version: # instead of nf_software_version
        type: string
        description: Software version of the NF
        required: true
      vnfm_info:
        type: list
        required: true
        description: Identifies VNFM(s) compatible with the NF
        entry_schema:
          type: string
      infra_requirements:
        type: InfraRequirements
        required: false
        description: These are the infra requirements for the Network Function
```

The sample shows customized TOSCA with the infrastructure requirements definition.

### infra_requirements definition

```
tosca.datatypes.nfv.InfraRequirements:
    derived_from: tosca.datatypes.Root
    properties:
      node_components:
        type: tosca.datatypes.nfv.NodeComponents
        required: false
      caas_components:
        type: list
        required: false
        entry_schema:
          type: tosca.datatypes.nfv.CaasComponentsData

  tosca.datatypes.nfv.NodeComponents:
    derived_from: tosca.datatypes.Root
    properties:
      isNumaConfigNeeded:
        type: boolean
        required: false
      kernel:
        type: tosca.datatypes.nfv.Kernel
        required: false
      network:
        type: tosca.datatypes.nfv.Network
        required: false
      passthrough_devices:
        type: list
        required: false
        entity_schema: tosca.datatypes.nfv.PassthroughDevices
      latency_sensitivity:
        type: string
        required: false
      additional_config:
        type: list
        required: false
        entity_schema: tosca.datatypes.nfv.AdditionalConfig
      file_injection:
        type: list
        required: false
        entity_schema: tosca.datatypes.nfv.FileInjection
 tosca.datatypes.nfv.Kernel:
    derived_from: tosca.datatypes.Root
    properties:
      kernel_type:
        type: tosca.datatypes.nfv.KernelType
        required: false
      kernel_args:
        type: list
        required: false
        entry_schema:
          type: tosca.datatypes.nfv.KernelArgsData
      kernel_modules:
        type: list
        required: false
        entry_schema:
          type: tosca.datatypes.nfv.KernelModulesData
      custom_packages:
        type: list
        required: false
        entry_schema:
          type: tosca.datatypes.nfv.CustomPackagesData

  tosca.datatypes.nfv.KernelType:
    derived_from: tosca.datatypes.Root
    properties:
      name:
        type: string
        required: true
      version:
        type: string
        required: true
      image:
        type: string
        required: false

  tosca.datatypes.nfv.KernelArgsData:
    derived_from: tosca.datatypes.Root
    properties:
      key:
        type: string
        required: true
      value:
        type: string
        required: false

  tosca.datatypes.nfv.KernelModulesData:
    derived_from: tosca.datatypes.Root
    properties:
      name:
        type: string
        required: true
      version:
        type: string
        required: true

  tosca.datatypes.nfv.CustomPackagesData:
    derived_from: tosca.datatypes.Root
    properties:
      name:
        type: string
        required: true
      version:
        type: string
        required: true

  tosca.datatypes.nfv.Network:
    derived_from: tosca.datatypes.Root
    properties:
      devices:
        type: list
        required: true
        entry_schema:
          type: tosca.datatypes.nfv.NetworkDevices

  tosca.datatypes.nfv.NetworkDevices:
    derived_from: tosca.datatypes.Root
    properties:
      deviceType:
        type: string
        required: true
      networkName:
        type: string
        required: true
      resourceName:
        type: string
        required: true
```

```
       dpdkBinding:
         type: string
         required: false
       interfaceName:
         type: string
         required: false
       count:
         type: integer
         required: false

  tosca.datatypes.nfv.PassthroughDevices:
     derived_from: tosca.datatypes.Root
     properties:
       deviceType:
         type: string
         required: true
       pf_group:
         type: string
         required: true
  tosca.datatypes.nfv.AdditionalConfig:
     derived_from: tosca.datatypes.Root
     properties:
       name:
         type: string
         required: true
       value:
         type: string
         required: true
  tosca.datatypes.nfv.FileInjection:
     derived_from: tosca.datatypes.Root
     properties:
       source:
         type: string
         required: true
       content:
         type: string
         required: true
       path:
         type: string
         required: true
  tosca.datatypes.nfv.CaasComponentsData:
     derived_from: tosca.datatypes.Root
     properties:
       name:
         type: string
         required: true
       type:
         type: string
         required: true
       version:
         type: string
         required: false
       properties:
         type: tosca.datatypes.nfv.CaasProperties
```

```
        required: false

   tosca.datatypes.nfv.CaasProperties:
     derived_from: tosca.datatypes.Root
```

## Supported DPDK and Kernel Versions

List of compatible Data Plane Development Kit (DPDK) and Photon OS kernel versions.

The table lists DPDK version and compatible Photon OS kernel versions.

| Photon OS Kernel Version | DPDK Version | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 17.11 | 17.11.10 | 18.11 | 18.11.7 | 19.08.2 | 19.11 | 19.11.1 | 20.11 |
| Linux-4.19.104-3.ph3 | | ✓ | | ✓ | ✓ | | ✓ | |
| Linux-rt-4.19.98-rt40-4.ph3 | | ✓ | | ✓ | ✓ | | ✓ | |
| Linux-4.19.97-2.ph3 | | ✓ | | ✓ | ✓ | | ✓ | |
| Linux-4.19.124-1.ph3 | | ✓ | | ✓ | ✓ | | ✓ | |
| Linux-rt-4.19.132-1.ph3 | | ✓ | | ✓ | ✓ | ✓ | ✓ | |
| Linux-4.19.132-1.ph3 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | |
| Linux-4.19.115-3.ph3 | | ✓ | | ✓ | ✓ | ✓ | ✓ | |
| Linux-4.19.145-2.ph3 | | ✓ | | ✓ | ✓ | ✓ | ✓ | |
| Linux-4.19.154-1.ph3 | | ✓ | | ✓ | ✓ | | ✓ | ✓ |
| Linux-rt-4.19.154-1.ph3 | | ✓ | | ✓ | ✓ | | ✓ | ✓ |
| Linux-4.19.154-11.ph3 | | ✓ | | ✓ | ✓ | | ✓ | ✓ |

# CNF with Customizations Example

Here are some CNF customization examples.

## Example 1

```
tosca_definitions_version: tosca_simple_profile_for_nfv_1_0_0
description: Network Function description
imports:
  - vmware_etsi_nfv_sol001_vnfd_2_5_1_types.yaml
node_types:
  tosca.nodes.nfv.VMware.CNF.cu-up-1.8:
    derived_from: tosca.nodes.nfv.VMware.CNF
    interfaces:
      Vnflcm:
        type: tosca.interfaces.nfv.Vnflcm
  tosca.nodes.nfv.Vdu.Compute.Helm.cuup-helm-chart:
    derived_from: tosca.nodes.nfv.Vdu.Compute.Helm
    properties:
      configurable_properties:
        type: tosca.datatypes.nfv.VnfcConfigurableProperties.cuup-helm-chart
        required: true
data_types:
  tosca.datatypes.nfv.VnfcConfigurableProperties.cuup-helm-chart:
    derived_from: tosca.datatypes.nfv.VnfcConfigurableProperties
    properties:
      additional_vnfc_configurable_properties:
        type: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.cuup-helm-chart
        description: Describes additional configuration for VNFC that can be configured
        required: true
  tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.cuup-helm-chart:
    derived_from: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties
    properties:
      values:
        required: true
        propertyName: values
        description: Overrides for chart values
        default: ''
        type: string
        format: file
      BHU:
        required: true
        propertyName: BHU
        description: ''
        default: ''
        type: string
        format: network
      F1U:
        required: true
        propertyName: F1U
        description: ''
        default: ''
        type: string
        format: network
      E1C:
        required: true
        propertyName: E1C
        description: ''
        default: ''
        type: string
        format: network
      MGMT:
        required: true
        propertyName: MGMT
        description: ''
        default: ''
        type: string
        format: network
  tosca.datatypes.nfv.VMware.Interface.InstantiateStartInputParameters:
    derived_from: tosca.datatypes.nfv.VnfOperationAdditionalParameters
    properties:
      USERNAME:
        name: USERNAME
        type: string
        description: K8s master username
        required: true
        default: capv
        format: string
      PASSWORD:
        name: PASSWORD
        type: password
        description: K8s master password
        required: true
        default:
        format: password
      IP:
        name: IP
        type: string
        description: K8s master ip address
        required: true
        default:
        format: string
      K8S_NAMESPACE:
        name: K8S_NAMESPACE
        type: string
        description: K8S namespace for CU-UP
        required: true
        default:
        format: string
      NAD_FILE:
        name: NAD_FILE
        type: string
        description: NAD Config File
        required: true
        default: ''
        format: file
  tosca.datatypes.nfv.VMware.Interface.InstantiateStartOutputParameters:
    derived_from: tosca.datatypes.nfv.VnfOperationAdditionalParameters
    properties:
      nsCreateResult:
        name: nsCreateResult
        type: string
        description: ''
topology_template:
  substitution_mappings:
    node_type: tosca.nodes.nfv.VMware.CNF.cu-up-1.8
  node_templates:
    cu-up-1.8:
      node_type: tosca.nodes.nfv.VMware.CNF.cu-up-1.8
      properties:
        descriptor_id: nfd_4e7599b5-9a44-4000-850c-7ec65d2f2423
        provider: Vendor01
        product_name: CU-UP
        version: '1.0'
        id: id
        software_version: '1.3.4761'
        descriptor_version: '1.8'
        flavour_id: default
        flavour_description: default
        vnfm_info:
          - gvnfmdriver
        infra_requirements:
          node_components:
            isNumaConfigNeeded: false
            kernel:
              kernel_type:
                name: linux
                version: 4.19.132-1.ph3
              kernel_modules:
                - name: dpdk
                  version: 19.11.1
              kernel_args:
                - key: default_hugepagesz
                  value: 1G
                - key: hugepagesz
                  value: 1G
                - key: hugepages
                  value: 10
                - key: transparent_hugepage
                  value: never
                - key: intel_idle.max_cstate
                  value: 1
                - key: iommu
                  value: pt
                - key: intel_iommu
                  value: 'on'
                - key: tsc
                  value: reliable
                - key: idle
                  value: pool
                - key: intel_pstate
                  value: disable
                - key: rcu_nocb_poll
                  value: 1
                - key: clocksource
                  value: tsc
                - key: pcie_aspm.policy
                  value: performance
                - key: skew_tick
```

```
                  value: 1
                - key: isolcpus
                  value: 11-17
                - key: nosoftlockup
                - key: nohz
                  value: 'on'
                - key: nohz_full
                  value: 11-17
                - key: rcu_nocbs
                  value: 11-17
              custom_packages:
                - name: pciutils
                  version: 3.6.2-1.ph3
                - name: tuned
                  version: 2.13.0-1.ph3
            network:
              devices:
                - deviceType: sriov
                  networkName: F1U
                  resourceName: ani_netdevice_cuup_f1u
                  dpdkBinding: igb_uio
                - deviceType: sriov
                  networkName: BHU
                  resourceName: ani_netdevice_cuup_bhu
                  dpdkBinding: igb_uio
                - deviceType: sriov
                  networkName: E1C
                  resourceName: ani_netdevice_cuup_e1c
                - deviceType: sriov
                  networkName: MGMT
                  resourceName: ani_netdevice_cuup_mgmt
                  count: 5
            additional_config:
              - name: tuned
                value: '[{"name":"vendor01-cu"}]'
            file_injection:
              - source: file
                content: ../Artifacts/scripts/tuned.conf
                path: /etc/tuned/cu/tuned.conf
              - source: file
                content: ../Artifacts/scripts/cpu-partitioning-variables.conf
                path: /etc/tuned/cpu-partitioning-variables.conf
          caas_components:
            - name: sriov
              type: cni
        description: Network Function description
      interfaces:
        Vnflcm:
          instantiate_start:
            implementation: ../Artifacts/workflows/CUUP_PreInstantiation_Steps.json
            description: Configure Vendor01 CU-UP
            inputs:
              type: >-
                tosca.datatypes.nfv.VMware.Interface.InstantiateStartInputParameters
              USERNAME: capv
              PASSWORD:
              IP:
              K8S_NAMESPACE:
              NAD_FILE: ''
            outputs:
              type: >-
                tosca.datatypes.nfv.VMware.Interface.InstantiateStartOutputParameters
              nsCreateResult: ''
    cuup-helm-chart:
      type: tosca.nodes.nfv.Vdu.Compute.Helm.cuup-helm-chart
      properties:
        name: cuup-helm-chart
        description: cu-up
        chartName: cuup-helm-chart
        chartVersion: 1.3.4760
        helmVersion: v3
        id: cuup-helm-chart
        configurable_properties:
          additional_vnfc_configurable_properties:
            type: >-
              tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.cuup-helm-chart
            values: ''
            BHU: ''
            F1U: ''
            E1C: ''
            MGMT: ''
  policies:
    - policy_scale:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: scale
          interface_type: operation
          isEnabled: true
    - policy_workflow:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: workflow
          interface_type: operation
          isEnabled: true
    - policy_reconfigure:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: reconfigure
          interface_type: operation
          isEnabled: true
    - policy_update:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: update
          interface_type: operation
          isEnabled: true
    - policy_upgrade:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: upgrade
          interface_type: operation
          isEnabled: true
    - policy_upgrade_package:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: upgrade_package
          interface_type: operation
          isEnabled: true
    - policy_instantiate_start:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: instantiate_start
          interface_type: workflow
          isEnabled: true
    - policy_instantiate_start:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: instantiate_start
          interface_type: workflow
          isEnabled: true
```

## Example 2

```
tosca_definitions_version: tosca_simple_profile_for_nfv_1_0_0
description: Network Function description
imports:
  - vmware_etsi_nfv_sol001_vnfd_2_5_1_types.yaml
node_types:
  tosca.nodes.nfv.VMware.CNF.du-1.8:
    derived_from: tosca.nodes.nfv.VMware.CNF
    interfaces:
      Vnflcm:
        type: tosca.interfaces.nfv.Vnflcm
  tosca.nodes.nfv.Vdu.Compute.Helm.du-helm-chart:
    derived_from: tosca.nodes.nfv.Vdu.Compute.Helm
    properties:
      configurable_properties:
        type: tosca.datatypes.nfv.VnfcConfigurableProperties.du-helm-chart
        required: true
data_types:
  tosca.datatypes.nfv.VnfcConfigurableProperties.du-helm-chart:
    derived_from: tosca.datatypes.nfv.VnfcConfigurableProperties
    properties:
      additional_vnfc_configurable_properties:
        type: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.du-helm-chart
        description: Describes additional configuration for VNFC that can be configured
        required: true
  tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.du-helm-chart:
    derived_from: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties
    properties:
      Input Yaml:
        required: true
        propertyName: Input Yaml
        description: ''
        default: ''
        type: string
        format: file
      F1U:
        required: true
        propertyName: F1U
        description: ''
        default: ''
        type: string
        format: network
      F1C:
        required: true
        propertyName: F1C
        description: ''
        default: ''
        type: string
        format: network
      MGMT:
        required: true
        propertyName: MGMT
        description: ''
        default: ''
        type: string
        format: network
      FH:
        required: true
        propertyName: FH
        description: ''
        default: ''
        type: string
        format: network
  tosca.datatypes.nfv.VMware.Interface.InstantiateStartInputParameters:
    derived_from: tosca.datatypes.nfv.VnfOperationAdditionalParameters
    properties:
      USERNAME:
        name: USERNAME
        type: string
        description: K8s master username
        required: true
        default: ''
        format: string
      PASSWORD:
        name: PASSWORD
        type: password
        description: K8s master password
        required: true
        default: ''
        format: password
      IP:
        name: IP
        type: string
        description: K8s master ip address
        required: true
        default: ''
        format: string
      NAD_FILE:
        name: NAD_FILE
        type: string
        description: The NAD Config file
        required: true
        default: ''
        format: file
      K8S_NAMESPACE:
        name: K8S_NAMESPACE
        type: string
        description: K8S namespace for DU
        required: true
        default: ''
        format: string
  tosca.datatypes.nfv.VMware.Interface.InstantiateStartOutputParameters:
    derived_from: tosca.datatypes.nfv.VnfOperationAdditionalParameters
    properties:
      nsCreateResult:
        name: nsCreateResult
        type: string
        description: ''
      copyNADResult:
        name: copyNADResult
        type: string
        description: ''
      nadCreateResult:
        name: nadCreateResult
        type: string
        description: ''
topology_template:
  substitution_mappings:
    node_type: tosca.nodes.nfv.VMware.CNF.du-1.8
  node_templates:
    du-1.8:
      node_type: tosca.nodes.nfv.VMware.CNF.du-1.8
      properties:
        descriptor_id: nfd_4e7599b5-9a44-4000-850c-7ec65d2f2422
        provider: Vendor01
        product_name: DU
        version: '1.0'
        id: id
        software_version: '1.3.4761'
        descriptor_version: '1.8'
        flavour_id: default
        flavour_description: default
        vnfm_info:
          - gvnfmdriver
        infra_requirements:
          node_components:
            isNumaConfigNeeded: true
            kernel:
              kernel_type:
                name: linux-rt
                version: 4.19.132-1.ph3
              kernel_modules:
                - name: dpdk
                  version: 19.11.1
              kernel_args:
                - key: nosoftlockup
                - key: noswap
                - key: softlockup_panic
                  value: 0
                - key: pcie_aspm.policy
                  value: performance
                - key: intel_idle.max_cstate
                  value: 1
                - key: mce
                  value: ignore_ce
                - key: fsck.mode
                  value: force
                - key: fsck.repair
                  value: yes
                - key: nowatchdog
                - key: cpuidle.off
                  value: 1
                - key: nmi_watchdog
                  value: 0
                - key: audit
                  value: 0
                - key: processor.max_cstate
                  value: 1
                - key: intel_pstate
                  value: disable
                - key: isolcpus
                  value: 8-{{tca.node.vmNumCPUs}}
                - key: skew_tick
                  value: 1
                - key: irqaffinity
                  value: 0-7
                - key: selinux
                  value: 0
                - key: enforcing
                  value: 0
                - key: nohz
                  value: 'on'
                - key: nohz_full
                  value: 8-{{tca.node.vmNumCPUs}}
                - key: rcu_nocb_poll
                  value: 1
                - key: rcu_nocbs
                  value: 8-{{tca.node.vmNumCPUs}}
                - key: idle
                  value: poll
                - key: default_hugepagesz
                  value: 1G
                - key: hugepagesz
                  value: 1G
                - key: hugepages
                  value: 17
                - key: intel_iommu
                  value: 'on'
                - key: iommu
                  value: pt
                - key: kthreads_cpu
                  value: 0-7
                - key: clock
                  value: tsc
                - key: clocksource
                  value: tsc
                - key: tsc
                  value: reliable
              custom_packages:
                - name: pciutils
                  version: 3.6.2-1.ph3
                - name: tuned
                  version: 2.13.0-3.ph3
                - name: linuxptp
                  version: 2.0-1.ph3
            additional_config:
              - name: tuned
                value: '[{"name":"vendor01-du"}]'
            file_injection:
              - source: file
                content: ../Artifacts/scripts/tuned.conf
                path: /etc/tuned/du/tuned.conf
              - source: file
                content: ../Artifacts/scripts/cpu-partitioning-variables.conf
                path: /etc/tuned/cpu-partitioning-variables.conf
              - source: file
                content: ../Artifacts/scripts/realtime-variables.conf
                path: /etc/tuned/realtime-variables.conf
            network:
              devices:
                - deviceType: sriov
                  networkName: F1U
                  resourceName: ani_netdevice_du_f1u
                  dpdkBinding: igb_uio
                - deviceType: sriov
                  networkName: F1C
                  resourceName: ani_netdevice_du_f1c
                - deviceType: sriov
                  networkName: FH
                  resourceName: ani_netdevice_du_fh
                  dpdkBinding: vfio-pci
                - deviceType: sriov
                  networkName: MGMT
                  resourceName: ani_netdevice_du_mgmt
                  count: 6
            passthrough_devices:
              - device_type: NIC
                pf_group: ptp
          caas_components:
            - name: sriov
              type: cni
      interfaces:
        Vnflcm:
          instantiate_start:
            implementation: ../Artifacts/workflows/DU-Preinstantion-WF.json
            description: Configure DU using a configmap
            inputs:
              type: >-
                tosca.datatypes.nfv.VMware.Interface.InstantiateStartInputParameters
              USERNAME: ''
              PASSWORD: ''
              IP: ''
              NAD_FILE: ''
              K8S_NAMESPACE: ''
            outputs:
              type: >-
                tosca.datatypes.nfv.VMware.Interface.InstantiateStartOutputParameters
              nsCreateResult: ''
              copyNADResult: ''
              nadCreateResult: ''
    du-helm-chart:
      type: tosca.nodes.nfv.Vdu.Compute.Helm.du-helm-chart
      properties:
        name: du-helm-chart
        description: Chart for DU
        chartName: du-helm-chart
        chartVersion: 1.3.4761
        helmVersion: v3
        id: du-helm-chart-1.0
        configurable_properties:
          additional_vnfc_configurable_properties:
            type: >-
              tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.du-helm-chart
            Input Yaml: ''
            F1U: 'cellsite-F1U'
            F1C: 'cellsite-F1C'
            MGMT: 'cellsite-mgmt'
            FH: 'cellsite-FH'
  policies:
    - policy_scale:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: scale
          interface_type: operation
          isEnabled: true
    - policy_workflow:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: workflow
          interface_type: operation
          isEnabled: true
    - policy_reconfigure:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: reconfigure
          interface_type: operation
          isEnabled: true
    - policy_update:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: update
          interface_type: operation
          isEnabled: true
    - policy_upgrade:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: upgrade
          interface_type: operation
          isEnabled: true
    - policy_upgrade_package:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: upgrade_package
          interface_type: operation
          isEnabled: true
    - policy_instantiate_start:
        type: tosca.policies.nfv.SupportedVnfInterface
        properties:
          interface_name: instantiate_start
          interface_type: workflow
          isEnabled: true
```

# Download a Network Function Package

You can download a network function package in the CSAR format to your local drive.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Catalog**.
3. Select the desired network function and click **Download**.
4. Select a location in your local drive and save the CSAR package.

# Managing Network Function Lifecycle Operations

# 10

Using VMware Telco Cloud Automation, you can instantiate, heal, scale in or out, run a workflow, and *terminate* a network function.

This chapter includes the following topics:

## Instantiating a Network Function

After you upload or create a network function, you can instantiate it in your virtual infrastructure.

### Instantiate a Virtual Network Function

To instantiate a VNF, follow the steps listed in this section.

#### Prerequisites

- Upload or create a network function.
- Upload all required images and templates to your vCenter Server instance.

#### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Catalog**.

3 Select the desired VNF and click **Instantiate**.

The Create Network Function Instance page is displayed.

4. In the **Inventory Detail** tab, enter the following information:

   - **Name** - Enter a name for your network function instance.
   - **Description** - Provide a description.
   - **Select Cloud** - Select a cloud from your network on which to instantiate the network function.
   - **Select Compute Profile** - Select a compute profile from the drop-down menu.
   - **Prefix (Optional)** - Enter a prefix. All entities that are created for this VNF are prefixed with this text. Prefixes help in personalizing and identifying the entities of a VNF.
   - **Instantiation Level** - Select the level of instances to create. The default level is 1.
   - **Grant Validation** - When you deploy a VNF, Grant validates whether the required images are available on the target system. It also validates whether the required resources such as CPU, memory, and storage capacity are available for a successful deployment. To configure Grant, go to **Advanced Settings** > **Grant Validation** and select one of the options:

     - **Enable**: Run validation and feasibility checks for the target cloud. Fail fast if the validation fails.
     - **Disable**: Do not run validation or feasibility checks for the target cloud.

5. Click **Next**.

6. In the **Network Function Properties** tab,

   - By default, VMware Telco Cloud Automation creates an internal network if you do not create any network. If you want to use an existing internal network, click **Edit Internal Network** and select the internal network to use.
   - For a network function with an external network, click **Select External Network** and select a network from the list.

7. Click **Next**.

8. The **Inputs** tab displays the following types of inputs to be provided:

   - The required OVF properties for each VDU within the VNF. Depending on the instantiation level that you have selected, there can be multiple instances deployed for each VDU. Ensure that you enter the correct information for each VDU.
   - The Helm inputs for each Helm chart within a CNF.
   - Any pre-workflows or post workflows that are defined as a part of the Network Function.

   Provide the appropriate information and click **Next**.

9. In the **Review** tab, review the configuration.

10. Click **Instantiate**.

### Results

VMware Telco Cloud Automation creates the virtual machines and networks required by your network function on the cloud that you specified. To view a list of all instantiated functions, select **Network Functions** > **Inventory**. To track and monitor the progress of the instantiation process, click the **Expand** icon on the network function and navigate further. When **Instantiated** is displayed in the **State** column for a network function, it indicates that the instantiation process is completed successfully and the function is ready to use.

If you no longer want to use an instantiated network function, click the **Options** (three dots) icon and select **Terminate**. Then select the network function and click **Delete**.

## Instantiate a Cloud Native Network Function

To instantiate a CNF, follow the steps listed in this section.

### Prerequisites

- Upload or create a network function.
- Upload all required images and templates to your vCenter Server instance.

**Note**

- Ensure that all Harbor repository URLs contain the appropriate port numbers such as 80, 443, 8080, and so on.
- Ensure that all the image repository URLs within the `values.yaml` file contain the appropriate Harbor port numbers.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Catalog**.
3. Select the desired CNF and click **Instantiate**.

   The Create Network Function Instance page is displayed.
4. In the **Inventory Detail** tab, enter the following information:
   - **Name** - Enter a name for your network function instance.
   - **Description** - Provide a description.
   - **Select Cloud** - Select a cloud from your network on which to instantiate the network function. If you have created the Kubernetes cluster instance using VMware Telco Cloud Automation, select the node pool.

   Under **Default Repository**, provide the Helm repository details:
   - **Namespace** - Enter the Kubernetes Cluster namespace.

- **Repo URL**
  - **Select Repo URL** - If you have added Harbor as the third-party repository provider, select the Harbor repository URL from the drop-down menu.
  - **Specify Repo URL** - Specify the repository URL. Optionally, enter the user name and password to access the repository.
- **Grant Validation** - When you deploy a CNF, Grant validates whether the required images are available on the target system. It also validates whether the required resources such as CPU, memory, and storage capacity are available for a successful deployment. Specific to CNFs, it downloads the Helm chart and performs a dry run of the operations on the cluster. If Grant encounters errors, it provides detailed error messages. To configure Grant, go to **Advanced Settings** > **Grant Validation** and select one of the options:
  - **Enable**: Run validation and feasibility checks for the target Kubernetes cluster. Fail fast if the validation fails.
  - **Enable and Ignore**: Run validation and feasibility checks for the target Kubernetes cluster. Ignore failures.
  - **Disable**: Do not run validation or feasibility checks for the target Kubernetes cluster.
- **Auto Roleback** - During a failure, the **Auto Roleback** option allows you to retain the Helm release and Kubernetes resources. To configure **Auto Roleback**, go to **Advanced Settings** > **Auto Roleback** and select one of the options:
  - **Enable**: During failure, do not retain Helm release and Kubernetes resources.
  - **Disable**: During failure, retain Helm release and Kubernetes resources for debugging.

5. Click **Next**.
6. In the **Network Function Properties** tab, click **Next**.
7. The **Inputs** tab displays any instantiation properties. Provide the appropriate inputs and click **Next**.
8. In the **Review** tab, review the configuration.
9. Click **Instantiate**.

### Results

VMware Telco Cloud Automation creates the virtual machines and networks required by your network function on the cloud that you specified. To view a list of all instantiated functions, select **Network Functions** > **Inventory**. To track and monitor the progress of the instantiation process, click the **Expand** icon on the network function and navigate further. When **Instantiated** is displayed in the **State** column for a network function, it indicates that the instantiation process is completed successfully and the function is ready to use.

If you no longer want to use an instantiated network function, click the **Options** (three dots) icon and select **Terminate**. Then select the network function and click **Delete**.

# Heal an Instantiated Network Function

If a network function instance does not operate as expected, you can heal it by either rebooting or recreating the network function.

### Prerequisites

Instantiate the network function.

**Note** This action is not supported on a Cloud Native Network Function (CNF).

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Inventory**.

3 Click the **Options** (three dots) icon for the desired network function and select **Heal**.

4 In the Heal page, enter a reason for healing the network function.

5 Select whether to restart or recreate the network function and click **Next**.

6 In the **Inputs** tab, enter the input variables required for starting and stopping the heal function. Provide any required inputs appropriately. Click **Next**.

7 Review the configuration and click **Finish**.

### Results

The instantiated network function is restarted or recreated.

To view relevant information and recent tasks, click the **Expand** (>) icon on the network function.

# Scale an Instantiated VNF

You can scale your network function in or out by aspect or instantiation level.

### Prerequisites

**Note**

- For this release, you cannot define a network function's scale aspects using the Network Function Designer. Instead, you can manually add the scale aspects to your descriptor YAML file as a part of the **Policies** section.
- Scale aspects and minimum and maximum values cannot be identified for network functions that are imported from a partner system. For these network functions, you must enter the valid values manually.
- The scale to level feature is not supported for network functions that are imported from a partner system.
- You can set the instantiation scale when instantiating a Virtual Network Function (VNF).

Verify that the network function descriptor for the instantiated network function includes scaling aspects. Network functions without scaling aspects cannot be scaled.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory**.
3. To scale a network function by aspect, perform the following steps:

   a. Click the **Options** (three dots) icon for the desired network function and select **Scale**.

   b. In the **Scale** tab, select the aspect to scale.

   c. Drag the scroll bar to select the number of steps to scale to be performed. The default number of steps is 1.

   d. Click **Next**.

   e. In the **Inputs** tab, enter the input variables required for starting and ending the scale. These credentials are required for running a workflow.

   f. Click **Next**.

   g. In the **Review** tab, review your configuration and click **Finish**.

4. To scale a network function by instantiation level, perform the following steps:

   a. Click the **Options** (three dots) icon for the desired network function and select **Scale To Level**.

   b. Select whether to scale the entire network function or only certain aspects.

   c. Select the desired scale level and click **Next**.

   d. In the **Inputs** tab, enter the input variables required for starting and ending the scale to level. Provide any required inputs appropriately.

   e. Click **Next**.

   f. Review the configuration and click **Finish**.

### What to do next

To view relevant information and recent tasks, click the **Expand** (>) icon on the network function.

## Scale an Instantiated CNF

You can scale an instantiated CNF by uploading a descriptor YAML file with the new Helm Chart values.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory**.

3 Click the ⋮ icon against the CNF you want to scale, and select **Scale**.

4 In the **Scale** tab, click **Browse** and upload the YAML file that contains the Helm Chart values.

5 Click **Next**.

6 In the **Inputs** tab, enter the appropriate properties.

7 Click **Next**.

8 In the **Review** tab, review the YAML file and click **Finish**.

### Results

The CNF uses the new Helm values from the YAML file to scale accordingly.

## Operate an Instantiated Network Function

To change the power state of a network function, use the **Operate** life-cycle operation. This operation powers on or powers off the VDUs belonging to a network function. For the stop operation, you can either perform a forceful stop or a graceful shutdown.

### Prerequisites

Instantiate the network function.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Inventory**.

3 Click the **Options** (three dots) icon for the desired network function and select **Operate**. You can also click the network function and select **Actions** > **Operate**.

4 In the **Operate** dialog box, change the power state to **Started** or **Stopped**.

5 If you select **Stopped**, select one of the following options:

   - **Forceful Stop** - Powers off the VDUs.
   - **Graceful Stop** - Shuts down the guest operating systems of the VDUs. Optionally, enter the **Graceful Stop Timeout** time in seconds.

6 Click **OK**.

### Results

The VDUs in the instantiated network function powers on or powers off according to your selection.

## Run a Workflow on an Instantiated Network Function

You can run a workflow on a network function instance that contains one or more interfaces.

### Prerequisites

For information about workflows and interfaces, see the Chapter 14 Running Workflows with vRealize Orchestrator.

- Instantiate your network function that contains one or more interfaces.
- To run a vRealize Orchestrator workflow, you must register vRealize Orchestrator with VMware Telco Cloud Automation Control Plane (TCA-CP). For more information, see the *VMware Telco Cloud Automation Deployment Guide*.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory**.
3. Click the **Options** (three dots) icon for the desired network function and select **Run a Workflow**.
4. Select the desired workflow and click **Next**.
5. Enter the required parameters for the workflow.
6. Review the configuration and click **Run**.

### What to do next

To view relevant information and recent tasks of a network function, click the **Expand** (>) icon on the network function.

## Terminate a Network Function

When you select **Terminate** on a network function, the underlying workloads are deleted from VMware Telco Cloud Automation.

### Prerequisites

The network function must be instantiated.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory**.
3. Click the **Options** (three dots) icon for the desired network function and select **Terminate**.

   VMware Telco Cloud Automation checks for inputs based on the workflows that you added for the catalog. If there are any inputs, you can update them here.
4. Click **Finish** after adding the inputs, if any.

### Results

The network function is *terminated*.

To view relevant information and recent tasks, click the **Expand** (>) icon on the network function.

# Managing Network Service Catalogs

# 11

A network service is a combination of network functions that run together. After configuring your network functions, you can upload network service descriptors or design new network service descriptors. You can then perform network service life-cycle operations such as instantiate, heal, monitor, and *terminate*.

This chapter includes the following topics:

## Onboarding a Network Service

Onboarding a network service includes uploading a network service package to the catalog, and creating or editing a network service descriptor draft.

### Upload a Network Service Package

Using VMware Telco Cloud Automation, you can upload a SOL001/SOL004 compliant network service descriptor and cloud service archive (CSAR) package. The system parses and validates the configuration, and presents the topology in a visual viewer. It then persists the entry into the network services catalog.

#### Prerequisites

- Add a cloud to your virtual infrastructure.
- Add any required network functions to your cloud.
- Verify that your network service descriptor complies with the following standards:
  - Must be in the CSAR format.
  - Must comply with the SOL001 or SOL004 standard.
  - Must comply with TOSCA Simple Profile in YAML version 1.2 or TOSCA Simple Profile for NFV version 1.0.

#### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Services** > **Catalog** and click **Onboard**.

   The Onboard Network Service page is displayed.

3 Select **Upload Network Service Package**.

4 Enter a name for your network service.

5 Click **Browse** and select the network service descriptor (CSAR) file.

6 Click **Upload**.

### Results

The specified network service is added to the catalog. You can now instantiate the network service.

### What to do next

- To instantiate the network service, see Instantiate a Network Service.
- To obtain the CSAR file corresponding to a network service, select the function in the catalog and click **Download**.
- To remove a network service from the catalog, first *terminate* and delete all instances using the network service. Then select the service in the catalog and click **Delete**.

## Design a Network Service Descriptor

Using the Network Service Designer, you can compose a compliant network service template. A network service descriptor is a deployment template that describes a network service's deployment and operational requirement. It is used to create a network service where life-cycle management operations are performed.

### Prerequisites

- Add a cloud to your virtual infrastructure.
- Add network functions to your cloud.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Services** > **Catalog** and click **Onboard**.

   The Onboard Network Function page is displayed.

3 Select **Design Network Service Descriptor**.

4 Enter a unique name for your network function and click **Design**.

   The Network Service Designer page is displayed.

5 In the Network Service Catalog Properties pane, enter the following information:

   - **Descriptor ID** - Enter the descriptor ID.

- **Designer** - Enter the company name of the designer.
- **Version** - Enter the product version.
- **Name** - Enter the name of the descriptor.
- **Invariant ID** - Enter the invariant ID that is unique to the descriptor.
- **Flavor ID** - Enter the unique ID for the new flavor.

6 (Optional) Add one or more workflows to your network service.

   You can add custom workflows using vRealize Orchestrator. For information about adding custom workflows, see Chapter 14 Running Workflows with vRealize Orchestrator.

   a Click **Add Workflow** and select the desired workflow from the drop-down menu:

   - **Instantiate Start**
   - **Instantiate End**
   - **Heal Start**
   - **Heal End**
   - **Scale Start**
   - **Scale End**
   - **Scale To Level Start**
   - **Scale To Level End**
   - **Terminate Start**
   - **Terminate End**
   - **Custom**

   b Click **Browse** and upload a Workflow Engine in the JSON format.

   c Enter any input and output variables specified in your script and select whether they are required.

7 Click **Update**.

   You can modify these settings later by clicking **Edit Network Service Catalog Properties** in the Network Service Designer.

8 You can drag Virtual Network Functions (VNFs), Cloud-Native Network Functions (CNFs), VNFs that are part of a Specialized Virtual Network Function Manager (SVNFM), and networks (NS Virtual Link) to the design area. You can also drag other Network Service catalogs to your Network Service to create a Nested Network Service.

9 On each network function and virtual link, click the **Edit** (pencil) icon to configure additional settings.

**VNF**

- **Name** - Name of the network function.
- **Description** - Description about the network function.
- **External Connection Points** - Virtual link for each external connection point.
- **Depends On** (Optional) - Specify the VNF or CNF to be deployed before deploying this VNF. In a scenario where you deploy many VNFs and CNFs, there can be dependencies between them on the order in which they are deployed. This option enables you to specify their deployment order.

**CNF**

- **Name** - Name of the network function.
- **Description** - Description about the network function.
- **Depends On** (Optional) - Specify the VNF or CNF to be deployed before deploying this CNF. In a scenario where you deploy many VNFs and CNFs, there can be dependencies between them on the order in which they are deployed. This option enables you to specify their deployment order.

**VNFs From SVNFM**

VMware Telco Cloud Automation auto-discovers VNFs that are part of an SVNFM registered as a partner system, and lists them in the catalog. You can use these VNFs for creating a Network Service Catalog.

- Name - Enter the name of the SVNFM.
- Description (Optional) - Description about the SVNFM.
- **Depends On** (Optional) - Specify the VNF, SVNFM, or CNF to be deployed before deploying this SVNFM. In a scenario where you deploy many SVNFMs, VNFs, and CNFs, there can be dependencies between them on the order in which they are deployed. This option enables you to specify their deployment order.

**Nested Network Services**

- **Name** - Name of the nested network service.
- **Description** - Description about the nested network service.

**Virtual Links**

- Network name
- Description
- Protocol

When you have finished modifying the settings of an item, click **Update**.

10 After adding and configuring all the necessary items, click **Upload**.

If you want to save your work and continue later, click **Save as Draft**.

### Results

The specified network service is added to the catalog. You can now instantiate the service.

### What to do next

- To obtain the CSAR file corresponding to a network service, select the service in the catalog and click **Download**.
- To remove a network service from the catalog, select the service in the catalog and click **Delete**.

## Edit Network Service Descriptor Drafts

If you have saved a draft in the Network Service Designer, you can modify or delete the draft later.

### Prerequisites

You must have created and saved a network service descriptor using the Network Service Designer.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Services** > **Catalog** and click **Onboard**.

3 Select **Edit Network Service Descriptor Drafts**.

4 Locate the desired draft in the table.

5 To modify the draft, click the **Edit** (pencil) icon. To remove the draft, click the **Delete** icon.

## Delete a Network Service

You can delete a network service from the catalog.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Services** > **Catalog**.

3 Select the desired network service and click **Delete**.

4 Confirm the action by clicking **OK**.

### Results

The network service is removed from the catalog.

# Download a Network Service Package

You can download a network service package in the CSAR format to your local drive.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Services** > **Catalog**.
3. Select the desired network service and click **Download**.
4. Select a location in your local drive and save the CSAR package.

# Managing Network Service Lifecycle Operations

12

You can instantiate, run a workflow, or *terminate* your network service instance.

This chapter includes the following topics:

## Instantiate a Network Service

After you upload or create a network service catalog, you can instantiate it in your virtual infrastructure.

### Prerequisites

- Upload or create a network service catalog.
- Register any VIMs required by the network service.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Services** > **Catalog**.

3 Select the desired network service and click **Instantiate**.

   - If you have saved a validated configuration that you want to replicate on this network service, click **Upload** on the top-right corner and upload the JSON file. The fields are then auto-populated with this configuration information and you can edit them as required.
   - If you want to create a network service configuration from the beginning, perform the next steps.

4 Enter the following details:

   a **Name** - Enter a name for your Network Service instance.

   b **Description (Optional)** - Enter an optional description for your Network Service.

   c **Prefix (Optional)** - Enter a prefix. All entities that are created for this Network Service are prefixed with this text. Prefixes help in personalizing and identifying the entities of a Network Service.

5 In the **Preview Network Service** tab, enter a name for the service, an optional description, review its design, and click **Next**.

6 In the **Deploy Network Function** tab, select a cloud on which to include each network function in the network service.

7 Click **Next**.

8 In the **Configure Network Functions** tab, click the **Edit** (pencil) icon on each of the network functions or Nested Network Service catalogs.

   a For a Nested Network Service, select a pre-deployed Network Service from the existing list of Network Services. This list is automatically curated based on the deployed instances of the Nested Network Service catalog.

   **Note** You can only select pre-instantiated Network Service instances for a Nested Network Service.

   b To deploy a new Network Function, click **Instantiate New**.

   - Optionally, to select a pre-deployed network function from an existing list of VNFs that are deployed and ready for instantiation, click **Select Existing**. VMware Telco Cloud Automation auto-discovers VNFs that are part of an SVNFM registered as a partner system, and lists them in the catalog. You can use these VNFs for creating a Network Service Catalog.

   - These Network Functions are curated automatically based on the deployed instances and the selected Cloud.

   - Instantiated Network Functions that are connected to other network services are not displayed in this list.

   c In the **Inventory Detail** tab, select the desired compute profile, select the instantiation level, and click **Next**.

   d In the **Network Function Properties** tab, select or edit an internal or external network, and click **Next**.

   e In the **Inputs** tab, provide the required inputs appropriately and click **Next**.

   f In the **Review** tab, review your configuration and click **Finish**.

**Note** You cannot add a deployment profile or select an internal or an external link on a CNF.

9 In the **Instantiate Properties** tab, enter the values for any required properties and click **Next**.

10 In the **Review** tab, review your configuration. You can download this configuration and reuse it for instantiating a network service catalog with a similar configuration. Click **Instantiate**.

### Results

VMware Telco Cloud Automation creates the network functions required by your network service on the clouds that you specified. To view a list of all instantiated functions, select **Network Services** > **Inventory**. To track and monitor the progress of the instantiation process, click the **Expand** icon on the network service and navigate further. When `instantiated` is displayed in the State column for a network service, it indicates that the instantiation process is completed successfully and the service is ready for use.

### What to do next

To view the relevant information and recent tasks, click the **Expand** (>) icon on the desired network service.

If you no longer want an instantiated network service, click the **Options** (three dots) icon and select **Terminate**. Then select the network service and click **Delete**.

## Run a Workflow on a Network Service

You can run a workflow on a network service instance that contains one or more interfaces.

### Prerequisites

- Instantiate your network service that contains one or more interfaces.
- To run a vRealize Orchestrator workflow, you must register vRealize Orchestrator with VMware Telco Cloud Automation Control Plane (TCA-CP). For more information, see the *VMware Telco Cloud Automation Deployment Guide*.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Services** > **Inventory**.

3 Click the **Options** (three dots) icon for the desired network service and select **Run a Workflow**.

4 Select the desired network service or network function workflow and click **Next**.

5 Enter the required parameters for the workflow.

6 Review the configuration and click **Run**.

### What to do next

To view the relevant information and recent tasks, click the **Expand** (>) icon on the desired network service.

# Heal a Network Service

If your Network Service does not work as expected, you can heal it by running a set of workflows. These workflows are designed to perform some pre-defined corrective actions on the Network Service and are pre-packaged when designing the Network Service catalog.

Heal a Network Service.

### Prerequisites

- Upload or create a network service.
- Register any VIMs required by the network service.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Services** > **Inventory**.
3. Click the ⋮ (vertical ellipsis) icon against the Network Service that you want to heal and select **Heal**.

   In the Heal page, you can either select the **Network Service** radio button or the **Network Function** radio button. Selecting **Network Function** displays the associated Network Functions in the Network Service. Select the relevant Network Functions to heal. In this example, we heal a Network Service.
4. Select the **Network Service** radio button.
5. In the **Select a Workflow** tab, select one of the pre-defined types of healing from the **Degree Healing** drop-down menu. This option is required for auditing purposes.
6. Select the pre-packaged workflow that is used for healing the Network Service and click **Next**.
7. In the **Inputs** tab, enter the properties of the workflow such as user name, password, host name, Network Service command, and VIM location.
8. Click **Next**.
9. In the **Review** tab, review the changes and click **Heal**.

### Results

The Network Service begins to heal. To view its progress, go to **Network Services** > **Inventory** and expand the Network Service.

# Terminate a Network Service

When you **Terminate** a network service, the underlying workloads are deleted from VMware Telco Cloud Automation.

### Prerequisites

The network service must be instantiated.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Services** > **Inventory**.
3. Click the **Options** (three dots) icon for the desired network service and select **Terminate**.

   VMware Telco Cloud Automation checks for inputs based on the workflows that you added for the catalog. The **Finish** button is then displayed.
4. Click **Finish**.

### Results

The network service is *terminated*.

To view the relevant information and recent tasks, click the **Expand** (>) icon on the desired network service.

# Upgrading Network Functions and Network Services 13

VMware Telco Cloud Automation allows you to make minor software updates and major package and component upgrades to your network functions and network services. You can then map your upgraded VNFs, CNFs, and Network Services to the latest version in the Catalog.

You can perform the following upgrades or updates:

**Package Upgrade**

When you upgrade the package of an existing VNF, CNF, or Network Service, VMware Telco Cloud Automation detects those changes and provides an option to update the software version and description. You can then point your VNF, CNF, or Network Service instance in the catalog to the newer version.

**Component Upgrade**

When you upgrade the components of a CNF, you can map them to a newer version. For example, when you update the Helm charts in a CNF instance and upgrade the instance to a newer CNF catalog and version, the deployed Helm charts is automatically upgraded to the newer version.

**Software Update**

You can perform minor software updates to your VNFs and CNFs and map them to their latest versions in the Catalog.

The only criteria for performing an update or an upgrade is that the software provider and the product name must be invariant across all versions.

The following table lists the type of upgrades and updates you can perform for VNFs, CNFs, and Network Services.

Table 13-1. Type of Upgrades

| Network Function/ Service | Package Upgrade | Software Update | Component Upgrade |
|---|---|---|---|
| VNF | Yes | No | No |
| CNF | Yes | Yes | Yes |
| Network Service | Yes | No | No |

This chapter includes the following topics:

- Upgrade a VNF Package
- Upgrade a CNF Package
- Update a CNF Software
- Upgrade a CNF
- Upgrade Network Service Package

# Upgrade a VNF Package

Upgrade your VNF package and map it to the latest version in the Catalog.

### Prerequisites

You must be a **System Administrator** or a **Network Function Deployer** to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory** and select the VNF to upgrade.
3. Click the ⋮ symbol against the VNF and select **Upgrade Package**.
4. In the **Upgrade Package** screen, select the new VNF catalog to upgrade your VNF to. The descriptor version changes accordingly to the selected catalog.

   **Note** Only those VNF catalogs that have the same software provider and product name are displayed.

5. Click **Upgrade**.

### Results

Your VNF is upgraded to the selected catalog version. The VNF instance now displays the upgraded catalog name in the **Network Functions** > **Inventory** tab.

# Upgrade a CNF Package

Upgrade your CNF package and map it to the latest version in the Catalog.

### Prerequisites

You must be a **System Administrator** or a **Network Function Deployer** to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory** and select the CNF to upgrade.
3. Click the ⋮ symbol against the CNF and select **Upgrade Package**.

4. In the **Upgrade Package** screen, select the new CNF catalog to upgrade to. The descriptor version changes accordingly to the selected catalog.

   **Note** Only those CNF catalogs that have the same software provider and product name are displayed.

5. Click **Upgrade**.

### Results

Your CNF is upgraded to the selected catalog version. The CNF instance now displays the upgraded catalog name in the **Network Functions** > **Inventory** tab.

## Update a CNF Software

When the underlying software of your CNF has a new release, you can update the software of your CNF instance to the latest version.

### Prerequisites

You must be a **System Administrator** or a **Network Function Deployer** to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory** and select the CNF to update.
3. Click the ⋮ symbol against the CNF and select **Update**.
4. In the **Update Revision** tab, select the CNF catalog to update to. The Descriptor version updates automatically based on your selection.
5. Click **Next**.
6. In the **Inventory Detail** tab, select the repository for your CNF.
7. In the **Inputs** tab, update the instantiation properties, if any.
8. In the **Review** tab, review the updates.
9. Click **Update**.

### Results

The new version of the CNF instance is installed in the workload cluster. The CNF instance points to the new catalog version.

## Upgrade a CNF

Upgrade the software version, descriptor version, components, repository details, instantiation properties, and Network Function properties of your CNF and map them to the newer version in the Catalog.

If the existing Helm Chart requires a software upgrade, the system upgrades the software version of the CNF instance. If the existing CNF instance is not present in the new catalog, you can map the current CNF instance to a new Helm Chart. If you do not make a selection, then the existing CNF instance is removed from the Workload Cluster.

### Prerequisites

You must be a **System Administrator** or a **Network Function Deployer** to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory** and select the VNF to upgrade.
3. Click the ⋮ symbol against the VNF and select **Upgrade**.
4. In the **Upgrade Revision** tab, select the software version and Descriptor version to upgrade to.
5. In the **Components** tab, select the upgraded components to be included in your CNF.
6. In the **Inventory** tab, select the repository URL from the drop-down menu, or specify the repository.
7. In the **Inputs** tab, update the instantiation properties, if any.
8. In the **Network Function Properties** tab, review the updated model. You can download or delete Helm Charts from the updated model.
9. In the **Review** tab, review the updates.

### Results

Your CNF is upgraded to the specified properties.

## Upgrade Network Service Package

Upgrade your Network Service package and map it to the latest version in the Catalog.

### Prerequisites

You must be a **System Administrator** or a **Network Service Deployer** to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Services** > **Inventory** and select the Network Service to upgrade.
3. Click the ⋮ symbol against the Network Service and select **Upgrade Package**.
4. In the **Upgrade Package** screen, select the new Network Service catalog and descriptor version to upgrade your Network Service to.
5. Click **Upgrade**.

### Results

Your Network Service is upgraded to the selected catalog version.

# 14 Running Workflows with vRealize Orchestrator

Use vRealize Orchestrator to run operations that are not supported natively on VMware Telco Cloud Automation.

VMware Telco Cloud Automation provides a workflow orchestration engine that is distributed (spans across multiple connected sites), reliable, scalable, consistent, efficient, and easily maintainable. Workflows are a series of steps that must be completed sequentially to get the work done. It is an orchestration of tasks or steps. Every step represents a piece of business logic such that the ordered execution produces a meaningful result.

Using vRealize Orchestrator, you can create custom workflows or use an existing workflow as a template to design a specific workflow to run on your network function or network service. For example, you can create a workflow to start or query the status of certain services within a network function. These workflows can then be uploaded to your catalog in VMware Telco Cloud Automation.

Here is a sample workflow that is used to run pre-instantiation checks on a network function:

```
{
  "id":"sample_workflow",
  "name": "Sample Workflow",
  "description":"Sample Description",
  "version":"1.0",
  "startStep":"step1",
  "variables": [
    {"name":"vnfId", "type": "string"},
    {"name":"stringVar1", "type": "string"}
  ],
  "input": [
    {"name": "USER", "description": "Username", "type": "string"},
    {"name": "PWD", "description": "Password", "type": "password"}
  ],
  "output": [
    {"name":"output", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step1",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"decision",
      "description": "Step 1 – Run SSH Command",
      "inBinding":[
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "sh /opt/vmware/service-start.sh"},
        {"name": "encoding", "type": "string", "default": ""},
        {"name": "hostNameOrIP", "type": "string", "default": "10.112.45.100"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "outputText", "type": "string", "exportName": "stringVar1"}
      ],
      "condition": [
        {
          "name": "stringVar1", "type": "string", "comparator": "equals", "value": "PASS",
          "nextStep": "step2"
        },
        {
          "name": "stringVar1", "type": "string", "comparator": "equals", "value": "FAIL",
          "nextStep": "END"
        }
      ]
    },
    {
      "stepId":"step2",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Step 2 - Run Custom vRO Workflow",
      "inBinding":[
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "vroWorkflowName", "type": "string", "default": "Run SSH command"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "sh /opt/vmware/service-status.sh"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "default": "10.112.45.100"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "outputText", "type": "string", "exportName": "output"}
      ],
      "nextStep":"END"
    }
  ]
}
```

Some of the key parameters/attributes supported by the workflow engine are:

| Parameter | Description |
|---|---|
| `id` | Workflow identification number. |
| `name` | Name of the workflow. |
| `description` | Description of the workflow. |

| Parameter | Description |
|---|---|
| variables | Any variables to be used within the script. |
| input | Any user inputs to be provided from VMware Telco Cloud Automation. |
| output | Any output for VMware Telco Cloud Automation to receive. |
| steps | List of steps in the workflow. |
| stepId | Each step is associated with a step ID. The first step in the workflow is step1. |
| name | The name of the step. |
| namespace | This parameter is a constant called nfv. |
| type | Define whether the step is a task or a decision type.<br>**Note** Decision type steps require a condition. |
| description | Description of the step. |
| inBinding | Enter the workflow inputs. These inputs identify the workflow to run on the network function or service.<br>■ name - Name of the workflow.<br>■ type - String<br>■ exportName - User-provided variable.<br>■ default - A default string. |
| outBinding | The output from vRealize Orchestrator. |
| nextStep | The next step in the workflow. |

After creating a workflow, you can upload it to the network function or network service catalog as a part of the onboarding process. For more information, see Design a Virtual Network Function Descriptor, Design a Network Service Descriptor.

This chapter includes the following topics:

- Key Concepts of Workflows
- Creating a Workflow
- Running Ansible Playbooks with VMware Telco Cloud Automation
- Workflow Examples
- Advanced Workflow Use Cases

# Key Concepts of Workflows

Workflows consist of a schema, attributes, and parameters. The workflow schema is the main component of a workflow as it defines all the workflow elements and the logical connections between them.

The workflow attributes and parameters are the variables that workflows use to transfer data. Orchestrator saves a workflow token every time a workflow runs, recording the details of that specific run of the workflow.

## Workflow Parameters

Workflows receive input parameters and generate output parameters when they run.

### Input Parameters

Input parameters are read-only variables. Most workflows require a certain set of input parameters to run. An input parameter is an argument that the workflow processes when it starts. The user, an application, another workflow, or an action passes input parameters to a workflow for the workflow to process when it starts.

For example, if a workflow resets a virtual machine, the workflow requires as an input parameter the name of the virtual machine.

To modify the value supplied by the workflow caller, or to read the information using an input parameter, copy the input parameter to an attribute.

### Output Parameters

Output parameters are write-only variables. A workflow's output parameters represent the result from the workflow run. Output parameters can change when a workflow or a workflow element runs. While workflows run, they can receive the output parameters of other workflows as input parameters.

For example, if a workflow creates a snapshot of a virtual machine, the output parameter for the workflow is the resulting snapshot.

To read the value of a variable, use an attribute within the workflow. To pass the value of that attribute to the workflow caller, copy the attribute to an output parameter.

## Workflow Attributes and Variables

Use attributes to pass information between the schema elements inside a workflow.

Attributes are read and write variables. It is a common design pattern to copy input parameters to attributes at the beginning of a workflow so that you can modify the value if necessary within the workflow. It is a common design pattern to copy attributes to output parameters at the end of a workflow so that you can read the value if necessary within the workflow.

## Workflow Bindings

Bindings populate elements with data from other elements by binding input and output parameters to workflow attributes.

With parameter bindings, you can explicitly state whether you want each of your workflow variables to be accessible.

### Inward Binding

You can read the value stored *in* the variable.

### Outward Binding

You can change the value stored by a variable. That is, you can write *out* to the variable.

## Creating a Workflow

The process for developing a workflow involves a series of phases. You can follow a different sequence of phases or skip a phase, depending on the type of workflow that you are developing. For example, you can create a workflow without custom scripting.

Generally, you develop a workflow through the following phases:

1. Create a workflow or create a duplicate of an existing workflow from the standard library.
2. Provide general information about the workflow.
3. Define the input parameters of the workflow.
4. Lay out and link the workflow schema to define the logical flow of the workflow.
5. Bind the input and output parameters of each schema element to workflow attributes.
6. Write the necessary scripts for scriptable task elements or custom decision elements.
7. Create the workflow presentation to define the layout of the input parameters dialog box that the users see when they run the workflow.
8. Validate the workflow.

If you are developing custom workflows, you can use the following workflow template:

```
{
  "id":"set_workflow_id",
  "name": "Set the workflow name",
  "description":"Provide a description for the workflow",
  "version":"1.0",
  "startStep":"item0", // Starting step for a workflow
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "INPUT_PARAM1", "description": "Description for the input param", "type": "string"},
    {"name": "INPUT_PARAM2", "description": "Description for the input param", "type": "string"}
  ],
  "output": [
    {"name":"output", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
    }
  ]
}
```

When creating a workflow, you must provide the following details:

- General Information
- Define Workflows and Variables.

## General Information

In the workflow template, provide the following information:

- `id` - Provide a workflow ID.
- Provide a `name` and `description` for the workflow.
- Set the `version` of the workflow.
- Define the workflow variables and parameters.

## Define Workflows and Variables

After providing the general information, you must provide the global variables, input parameters, and output parameters of the workflow. For more information, see Defining Workflow Variables and Parameters.

## Defining Workflow Variables and Parameters

Workflow variables are used to pass data within the workflow steps. Workflow input parameters are the data provided by the caller of the workflow. Workflow output parameters are the data that is in the output of the workflow after the workflow is run.

### Define Workflow Variables

Workflow variables are used as placeholders to share data between the execution steps of the workflow. Input parameters are copied as variables at the beginning of the workflow execution. After a workflow step is run, output parameters can be copied to the workflow variables and used as inputs for other workflow steps.

**Note** A workflow attribute must not have the same name as a workflow parameter.

#### Variable Example

```
{
    "name": "vmName",
    "description": "The description of the variable",
    "type": "string"
}
```

## Variable Schema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "$id": "http://example.com/example.json",
    "type": "object",
    "readOnly": false,
    "writeOnly": false,
    "minProperties": 0,
    "title": "The Root Schema",
    "description": "The root schema comprises the entire JSON document.",
    "additionalProperties": true,
    "required": [
        "name",
        "description",
        "type"
    ],
    "properties": {
        "name": {
            "$id": "#/properties/name",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Name Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "vmName"
            ]
        },
        "description": {
            "$id": "#/properties/description",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Description Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "The description of the variable"
            ]
        },
        "type": {
            "$id": "#/properties/type",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Type Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "string",
                "number",
                "boolean"
            ]
        }
    }
}
```

## Define Workflow Parameters

You can use input and output parameters to pass data in and out of the workflow.

The input parameters are the initial data the workflow requires to run. You provide the values for the input parameters when you run the workflow. The output parameters are the data the workflow returns when it finishes its execution.

### Parameter Example

```
{
    "name": "vmName",
    "description": "The description of the variable",
    "type": "string",
    "default":"vm-32",
    "value":"vm-32"
}
```

### Parameter Schema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "$id": "http://example.com/example.json",
    "type": "object",
    "readOnly": false,
    "writeOnly": false,
    "minProperties": 0,
    "title": "The Root Schema",
    "description": "The root schema comprises the entire JSON document.",
    "additionalProperties": true,
    "required": [
        "name",
        "type"
    ],
    "properties": {
        "name": {
            "$id": "#/properties/name",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Name Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "vmName"
            ]
        },
        "description": {
            "$id": "#/properties/description",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Description Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "The description of the variable"
            ]
        },
        "type": {
            "$id": "#/properties/type",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Type Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "string",
                "boolean",
                "number"
            ]
        },
        "default": {
            "$id": "#/properties/default",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Default Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "vm-32"
            ]
        },
        "value": {
            "$id": "#/properties/value",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Value Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "vm-32"
            ]
        }
    }
}
```

## Add Workflow Steps to Your Workflow

Workflow steps are the sequential set of tasks that your workflow performs. You can add any number of steps to your workflow.

### Workflow Step Example

```
"steps":[
    {
      "stepId":"step1",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"decision",
      "description": "Step 1 - Run SSH Command",
      "inBinding":[
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "sh /opt/service-start.sh"},
        {"name": "encoding", "type": "string", "default": ""},
        {"name": "hostNameOrIP", "type": "string", "exportName": "SYSTEM_IP"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "outputText", "type": "string", "exportName": "stringVar1"}
      ],
      "nextStep":"END"
    }
  ]
```

### Workflow Step Schema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "$id": "http://example.com/example.json",
    "type": "object",
    "readOnly": false,
    "writeOnly": false,
    "minProperties": 0,
    "title": "The Root Schema",
    "description": "The root schema comprises the entire JSON document.",
    "additionalProperties": true,
    "required": [
        "stepId",
        "workflow",
        "namespace",
        "type",
        "inBinding",
        "outBinding",
        "nextStep",
        "stepNumber"
    ],
    "properties": {
        "stepId": {
            "$id": "#/properties/stepId",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Stepid Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "step1"
            ]
        },
        "workflow": {
            "$id": "#/properties/workflow",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Workflow Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "VRO_CUSTOM_WORKFLOW"
            ]
        },
        "namespace": {
            "$id": "#/properties/namespace",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Namespace Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "nfv"
            ]
        },
        "type": {
            "$id": "#/properties/type",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Type Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "task"
            ]
        },
        "description": {
            "$id": "#/properties/description",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Description Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "Basic"
            ]
        },
        "inBinding": {
            "$id": "#/properties/inBinding",
            "type": "array",
            "readOnly": false,
            "writeOnly": false,
            "uniqueItems": false,
            "minItems": 0,
            "minContains": 1,
            "title": "The Inbinding Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": [],
            "additionalItems": true,
            "items": {
                "$id": "#/properties/inBinding/items",
                "type": "object",
                "readOnly": false,
                "writeOnly": false,
                "minProperties": 0,
                "title": "The Items Schema",
                "description": "An explanation about the purpose of this instance.",
                "default": {},
                "examples": [
                    {
                        "name": "username",
                        "type": "string",
                        "exportName": "USER"
                    },
                    {
                        "type": "string",
                        "name": "vroWorkflowName",
                        "default": "Run SSH command"
                    },
                    {
                        "type": "password",
                        "exportName": "PWD",
                        "name": "password"
                    },
                    {
                        "type": "number",
                        "name": "port",
                        "default": "22"
                    },
                    {
                        "name": "cmd",
                        "default": "sh /opt/vmware/return-string-pass.sh",
                        "type": "string"
                    },
                    {
                        "type": "string",
                        "name": "encoding",
                        "default": " "
                    },
                    {
                        "name": "hostNameOrIP",
                        "default": "10.144.164.91",
                        "type": "string"
                    },
                    {
                        "type": "boolean",
                        "name": "passwordAuthentication",
                        "default": "true"
                    }
                ],
                "additionalProperties": true,
                "required": [
                    "name",
                    "type",
                    "exportName"
                ],
                "properties": {
                    "name": {
                        "$id": "#/properties/inBinding/items/properties/name",
                        "type": "string",
                        "readOnly": false,
                        "writeOnly": false,
                        "minLength": 0,
                        "title": "The Name Schema",
                        "description": "An explanation about the purpose of this instance.",
                        "default": "",
                        "examples": [
                            "username"
                        ]
                    },
                    "type": {
                        "$id": "#/properties/inBinding/items/properties/type",
                        "type": "string",
                        "readOnly": false,
                        "writeOnly": false,
                        "minLength": 0,
                        "title": "The Type Schema",
                        "description": "An explanation about the purpose of this instance.",
                        "default": "",
                        "examples": [
                            "string"
                        ]
                    },
                    "exportName": {
                        "$id": "#/properties/inBinding/items/properties/exportName",
                        "type": "string",
                        "readOnly": false,
                        "writeOnly": false,
                        "minLength": 0,
                        "title": "The Exportname Schema",
                        "description": "An explanation about the purpose of this instance.",
                        "default": "",
                        "examples": [
                            "USER"
                        ]
                    }
                }
            }
        },
        "outBinding": {
            "$id": "#/properties/outBinding",
            "type": "array",
            "readOnly": false,
            "writeOnly": false,
            "uniqueItems": false,
            "minItems": 0,
            "minContains": 1,
            "title": "The Outbinding Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": [],
            "additionalItems": true,
            "items": {
                "$id": "#/properties/outBinding/items",
                "type": "object",
                "readOnly": false,
                "writeOnly": false,
                "minProperties": 0,
                "title": "The Items Schema",
                "description": "An explanation about the purpose of this instance.",
                "default": {},
                "examples": [
                    {
                        "type": "string",
                        "exportName": "output",
                        "name": "outputText"
                    }
                ],
                "additionalProperties": true,
                "required": [
                    "name",
                    "type",
                    "exportName"
                ],
                "properties": {
                    "name": {
                        "$id": "#/properties/outBinding/items/properties/name",
                        "type": "string",
                        "readOnly": false,
                        "writeOnly": false,
                        "minLength": 0,
                        "title": "The Name Schema",
                        "description": "An explanation about the purpose of this instance.",
                        "default": "",
                        "examples": [
                            "outputText"
                        ]
                    },
                    "type": {
                        "$id": "#/properties/outBinding/items/properties/type",
                        "type": "string",
                        "readOnly": false,
                        "writeOnly": false,
                        "minLength": 0,
                        "title": "The Type Schema",
                        "description": "An explanation about the purpose of this instance.",
                        "default": "",
                        "examples": [
                            "string"
                        ]
                    },
                    "exportName": {
                        "$id": "#/properties/outBinding/items/properties/exportName",
                        "type": "string",
                        "readOnly": false,
                        "writeOnly": false,
                        "minLength": 0,
                        "title": "The Exportname Schema",
                        "description": "An explanation about the purpose of this instance.",
                        "default": "",
                        "examples": [
                            "output"
                        ]
                    }
                }
            }
        },
        "nextStep": {
            "$id": "#/properties/nextStep",
            "type": "string",
            "readOnly": false,
            "writeOnly": false,
            "minLength": 0,
            "title": "The Nextstep Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": "",
            "examples": [
                "step2"
            ]
        },
        "stepNumber": {
            "$id": "#/properties/stepNumber",
            "type": "integer",
            "readOnly": false,
            "writeOnly": false,
            "title": "The Stepnumber Schema",
            "description": "An explanation about the purpose of this instance.",
            "default": 0,
            "examples": [
                1
            ]
        }
    }
}
```

# Running Ansible Playbooks with VMware Telco Cloud Automation

Starting with VMware Telco Cloud Automation version 1.8, you can upload binary files when designing network functions. Package your Ansible Playbooks as a ZIP file and place it within the CSAR file. You can then run your Ansible Playbooks through VMware Telco Cloud Automation using vRealize Orchestrator (vRO) workflows.

## Bundling Ansible Playbooks in Network Function Catalogs

To upload your Ansible Playbook, perform the following steps:

1. Log in to the VMware Telco Cloud Automation web interface.
2. Design a Network Function descriptor. For more information, see Designing a Network Function Descriptor.
   a. In the **Network Function Catalog Properties** window, add the vRO Workflow that runs your Ansible Playbook to the network function.
   b. In the **Resources** tab, upload your Ansible Playbook scripts as a ZIP file, if necessary.

**Note** The maximum size limit for copying files through vRO is 20 MB.

## Running Ansible Playbooks

The Workflow.JSON file that you upload performs the following steps through vRO Workflows when you instantiate your VNF or CNF:

1. Copies the Ansible Playbook package from the CSAR to the Ansible host.
2. Unzips the package on the Ansible host.
3. Replaces the host file on the Ansible host and performs any environment-related settings.
4. Runs the Ansible Playbooks through vRO SSH Workflows.
5. Returns the Ansible Playbook output to VMware Telco Cloud Automation.

## Sample Ansible Playbook Workflow

For a sample workflow for deploying LAMP through Ansible, see Ansible Workflow.

# Workflow Examples

Here are some workflow examples you can use.

## Ansible Workflow

Workflow for deploying LAMP through Ansible.

```
{
  "id":"ansible-workflow",
  "name": "Workflow for deploying LAMP via Ansible",
  "description":"Sample Workflow for deploying an entire LAMP stack via Ansible",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "ANSIBLE_HOST_IP", "description": "Ansible Host IP Address", "type": "string"},
    {"name": "ANSIBLE_HOST_USER", "description": "Ansible Host Username", "type": "string"},
    {"name": "ANSIBLE_HOST_PWD", "description": "Ansible Host Password", "type": "password"},
    {"name": "WEBSERVER_IP", "description": "Web Server IP Address", "type": "string"},
    {"name": "DBSERVER_IP", "description": "DB Server IP Address", "type": "string"}
  ],
  "output": [
    {"name":"COPY_RESULT", "description": "Copy Output", "type": "string"},
    {"name":"SSH_OUTPUT", "description": "SSH Command Output", "type": "string"},
    {"name":"ANSIBLE_OUTPUT", "description": "Ansible Command Output", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"COPY_FILE_TO_GUEST", "namespace": "nfv",
      "type":"task",
      "description": "Copy Ansible Scripts from CSAR",
      "inBinding":[
         {"name": "username", "type": "string", "exportName": "ANSIBLE_HOST_USER"}
        ,{"name": "password", "type": "password", "exportName": "ANSIBLE_HOST_PWD"}
        ,{"name": "ip", "type": "string", "exportName": "ANSIBLE_HOST_IP"}
        ,{"name": "inFile", "type": "file", "default": [{"name": "ansible-centos7-lamp-master.zip"}]}
        ,{"name": "workingDirectory", "type": "string", "default": "/opt/ansible"}
        ,{"name": "destinationFileName", "type": "string", "default": "ansible-centos7-lamp-
master.zip"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "COPY_RESULT"}
      ],
      "nextStep": "step1"
    },
    {
      "stepId":"step1",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Unzip Ansible Scripts",
      "inBinding":[
        {"name": "vroWorkflowName", "type": "string", "default": "Run SSH Command"},
        {"name": "username", "type": "string", "exportName": "ANSIBLE_HOST_USER"},
        {"name": "password", "type": "password", "exportName": "ANSIBLE_HOST_PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "cd /opt/ansible; unzip ansible-centos7-lamp-
master.zip; "},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "ANSIBLE_HOST_IP"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "SSH_OUTPUT"}
      ],
      "nextStep":"step2"
    },
    {
      "stepId":"step2",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Prepare Ansible Hosts file",
      "inBinding":[
        {"name": "vroWorkflowName", "type": "string", "default": "Run SSH Command"},
        {"name": "username", "type": "string", "exportName": "ANSIBLE_HOST_USER"},
        {"name": "password", "type": "password", "exportName": "ANSIBLE_HOST_PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "cd /opt/ansible/ansible-centos7-lamp-master;
sed -i -e 's/client1.example.com/{{WEBSERVER_IP}}/g' /opt/ansible/ansible-centos7-lamp-master/hosts;
sed -i -e 's/client2.example.com/{{DBSERVER_IP}}/g' /opt/ansible/ansible-centos7-lamp-master/hosts"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "ANSIBLE_HOST_IP"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "SSH_OUTPUT"}
      ],
      "nextStep":"step3"
    },
    {
      "stepId":"step3",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Execute Ansible Playbook",
      "inBinding":[
        {"name": "vroWorkflowName", "type": "string", "default": "Run SSH Command"},
        {"name": "username", "type": "string", "exportName": "ANSIBLE_HOST_USER"},
        {"name": "password", "type": "password", "exportName": "ANSIBLE_HOST_PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "cd /opt/ansible/ansible-centos7-lamp-master;
ansible-playbook -v -i hosts site.yml"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "ANSIBLE_HOST_IP"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "ANSIBLE_OUTPUT"}
      ],
      "nextStep":"END"
    }
  ]
}
```

# SSH Workflows

Here are some sample SSH workflows.

## SSH Workflow

```
{
  "id":"ssh_workflow",
  "name": "SSH Workflow",
  "description":"SSH Workflow",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "USER", "description": "Username", "type": "string"},
    {"name": "PWD", "description": "Password", "type": "password"},
    {"name": "HOSTNAME", "description": "Hostname", "type": "string"},
    {"name": "CMD", "description": "Command", "type": "string"}
  ],
  "output": [
    {"name":"output", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "SSH Command",
      "inBinding":[
       {"name": "username", "type": "string", "exportName": "USER"}
      ,{"name": "password", "type": "password", "exportName": "PWD"}
      ,{"name": "port", "type": "number", "default": "22"}
      ,{"name": "cmd", "type": "string", "exportName": "CMD"}
      ,{"name": "encoding", "type": "string", "default": " "}
      ,{"name": "hostNameOrIP", "type": "string", "exportName": "HOSTNAME"}
      ,{"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "output"}
      ],
      "nextStep":"END"
    }
  ]
}
```

## SSH Workflow with Sleep

```
{
  "id":"ssh_workflow_with_sleep",
  "name": "SSH Workflow with Sleep",
  "description":"SSH Workflow with 60 seconds sleep in step 0",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "USER", "description": "Username", "type": "string"},
    {"name": "PWD", "description": "Password", "type": "password"},
    {"name": "HOSTNAME", "description": "Hostname", "type": "string"},
    {"name": "CMD", "description": "Command", "type": "string"}
  ],
  "output": [
    {"name":"output", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "Step with 60 sec Sleep",
      "inBinding":[
        { "name": "initialDelay", "type": "number", "default": "60" },
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "exportName": "CMD"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "HOSTNAME"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "output"}
      ],
      "nextStep":"END"
    }
  ]
}
```

### Parameterized SSH Workflow

```
{
  "id":"parameterized_ssh_workflow",
  "name": "Parameterized SSH Workflow",
  "description":"Parameterized SSH Workflow",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "USER", "description": "Username", "type": "string"},
    {"name": "PWD", "description": "Password", "type": "password"},
    {"name": "HOSTNAME", "description": "Hostname", "type": "string"},
    {"name": "SCRIPT_ARGUMENT", "description": "Script Argument", "type": "string"}
  ],
  "output": [
    {"name":"output", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "Parameterized SSH Command",
      "inBinding":[
       {"name": "username", "type": "string", "exportName": "USER"}
      ,{"name": "password", "type": "password", "exportName": "PWD"}
      ,{"name": "port", "type": "number", "default": "22"}
      ,{"name": "cmd", "type": "string", "default": "/opt/script1.sh {{SCRIPT_ARGUMENT}}"}
      ,{"name": "encoding", "type": "string", "default": " "}
      ,{"name": "hostNameOrIP", "type": "string", "exportName": "HOSTNAME"}
      ,{"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "output"}
      ],
      "nextStep":"END"
    }
  ]
}
```

## File Workflow Example

Here are some sample File workflows.

### Copy File Packaged Within Catalog to Guest

```
{
  "id":"copy-file-to-guest",
  "name": "Copy File Packaged Within Catalog to Guest",
  "description":"Copy a File bundled within the CSAR under Artifacts/scripts to a remote machine",
  "version":"1.0",
  "startStep":"item0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
     {"name": "USERNAME", "description": "Username", "type": "string"}
    ,{"name": "PASSWORD", "description": "Password", "type": "password"}
    ,{"name": "IP", "description": "IP Address of the Guest", "type": "string"}
  ],
  "output": [
     {"name": "copyResult", "type": "string"}
    ,{"name": "createResult", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"item0",
      "workflow":"COPY_FILE_TO_GUEST", "namespace": "nfv",
      "type":"task",
      "description": "Copy file",
      "inBinding":[
         {"name": "username", "type": "string", "exportName": "USERNAME"}
        ,{"name": "password", "type": "password", "exportName": "PASSWORD"}
        ,{"name": "ip", "type": "string", "exportName": "IP"}
        ,{"name": "inFile", "type": "file", "default": [{"name": "file.txt"}]}
        ,{"name": "workingDirectory", "type": "string", "default": "/tmp"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "copyResult"}
      ],
      "nextStep": "END"
    }
  ]
}
```

## Copy User Provided File to Host

```
{
  "id":"copy-file-to-guest-user-input",
  "name": "Copy user provided file to host",
  "description":"Copy user provided file to host",
  "version":"1.0",
  "startStep":"item0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "USERNAME", "description": "K8s master username", "type": "string"}
    ,{"name": "FILENAME", "description": "Filename", "type": "file"}
    ,{"name": "PASSWORD", "description": "K8s master password", "type": "password"}
    ,{"name": "IP", "description": "K8s master ip address", "type": "string"}
  ],
  "output": [
    {"name": "copyResult", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"item0",
      "workflow":"COPY_FILE_TO_GUEST", "namespace": "nfv",
      "type":"task",
      "description": "Copy file",
      "inBinding":[
         {"name": "username", "type": "string", "exportName": "USERNAME"}
        ,{"name": "password", "type": "password", "exportName": "PASSWORD"}
        ,{"name": "ip", "type": "string", "exportName": "IP"}
        ,{"name": "inFile", "type": "file", "exportName": "FILENAME"}
        ,{"name": "workingDirectory", "type": "string", "default": "/tmp"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "copyResult"}
      ],
      "nextStep": "END"
    }
  ]
}
```

## Custom vRO Workflows

Here are some sample custom vRO workflows.

### Custom vRO Workflow Execution

```
{
  "id":"custom-vro-workflow",
  "name": "Custom vRO Workflow Execution",
  "description":"Execute a Custom defined vRO Workflow which is pre-uploaded / defined in vRO",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "STR_INPUT", "description": "Sample String Input for Workflow", "type": "string"},
    {"name": "NUM_INPUT", "description": "Sample Number Input for Workflow", "type": "number"},
    {"name": "BOOL_INPUT", "description": "Sample Boolean Input for Workflow", "type": "boolean"},
    {"name": "PWD_INPUT", "description": "Sample Password (SecureString) Input for Workflow", "type":
"password"}
  ],
  "output": [
    {"name":"output", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Execute vRO Workflow",
      "inBinding":[
       {"name": "vroWorkflowName", "type": "string", "default": "Custom-vRO-Workflow-Name"}
      ,{"name": "strInput", "type": "string", "exportName": "STR_INPUT"}
      ,{"name": "numInput", "type": "number", "exportName": "NUM_INPUT"}
      ,{"name": "boolInput", "type": "boolean", "exportName": "BOOL_INPUT"}
      ,{"name": "pwdInput", "type": "password", "exportName": "PWD_INPUT"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "output"}
      ],
      "nextStep":"END"
    }
  ]
}
```

## Custom vRO Workflow Execution with File Upload

```
{
  "id":"custom-vro-workflow-with-file",
  "name": "Custom vRO Workflow Execution with File Upload",
  "description":"Execute a Custom defined vRO Workflow with File Input",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "STR_INPUT", "description": "Sample String Input for Workflow", "type": "string"},
    {"name": "NUM_INPUT", "description": "Sample Number Input for Workflow", "type": "number"},
    {"name": "BOOL_INPUT", "description": "Sample Boolean Input for Workflow", "type": "boolean"},
    {"name": "PWD_INPUT", "description": "Sample Password (SecureString) Input for Workflow", "type": 
"password"},
    {"name": "FILE_INPUT", "description": "Sample File Input for Workflow", "type": "file"}
  ],
  "output": [
    {"name":"output", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Execute vRO Workflow with File",
      "inBinding":[
       {"name": "vroWorkflowName", "type": "string", "default": "Custom-vRO-Workflow-Name-with-File-
Input"}
      ,{"name": "strInput", "type": "string", "exportName": "STR_INPUT"}
      ,{"name": "numInput", "type": "number", "exportName": "NUM_INPUT"}
      ,{"name": "boolInput", "type": "boolean", "exportName": "BOOL_INPUT"}
      ,{"name": "pwdInput", "type": "password", "exportName": "PWD_INPUT"}
      ,{"name": "inFile", "type": "file", "exportName": "FILE_INPUT"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "output"}
      ],
      "nextStep":"END"
    }
  ]
}
```

## VMware Tools Script

Run a script using VMware Tools.

```
{
  "id": "run-script-via-vm-tools",
  "name": "Run Script via VMware Tools",
  "description": "Run Script via VMware Tools with Initial Delay",
  "version": "1.0",
  "startStep": "item0",
  "variables": [
    { "name": "vnfId", "type": "string" }
  ],
  "input": [
    { "name": "VDU_USER", "description": "VDU Username", "type": "string" },
    { "name": "VDU_PWD", "description": "VDU Password", "type": "password" },
    { "name": "VDUNAME", "description": "VDU Name from VNF Catalog", "type": "string" }
  ],
  "output": [
    { "name": "VDU_RESULT", "description": "VDU Result", "type": "string" }
  ],
  "steps": [
    {
      "stepId":"item0",
      "workflow":"RUN_PROGRAM_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "VM Tools Script with Delay",
      "inBinding": [
        { "name": "initialDelay", "type": "number", "default": "300" },
        { "name": "username", "type": "string", "exportName": "VDU_USER" },
        { "name": "password", "type": "password", "exportName": "VDU_PWD" },
        { "name": "vduName", "type": "string", "exportName": "VDUNAME" },
        { "name": "scriptType", "type": "string", "value": "bash", "default": "bash" },
        { "name": "script", "type": "string", "default": "uptime" },
        { "name": "scriptTimeout", "type": "number", "default": "12" },
        { "name": "scriptRefreshTime", "type": "number", "default": "5" },
        { "name": "scriptWorkingDirectory", "type": "string", "default": "/bin" },
        { "name": "interactiveSession", "type": "boolean", "value": false, "default": "false" }
      ],
      "outBinding": [
        { "name": "result", "type": "string", "exportName": "VDU_RESULT" }
      ],
      "nextStep": "END"
    }
  ]
}
```

## Multiple Steps

Workflow with multiple steps.

```
{
  "id":"multi-step-workflow",
  "name": "Workflow with Multiple Steps",
  "description":"Sample Workflow with Multiple Steps",
  "version":"1.0",
  "startStep":"step1",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "STEP1_USER", "description": "Step 1 Username", "type": "string"},
    {"name": "STEP1_PWD", "description": "Step 1 Password", "type": "password"},
    {"name": "STEP1_HOSTNAME", "description": "Step 1 Hostname", "type": "string"},
    {"name": "STEP1_CMD", "description": "Step 1 Command", "type": "string"},
    {"name": "STEP2_STR_INPUT", "description": "Step 2 String Input for Workflow", "type": "string"},
    {"name": "STEP2_NUM_INPUT", "description": "Step 2 Number Input for Workflow", "type": "number"},
    {"name": "STEP2_BOOL_INPUT", "description": "Step 2 Boolean Input for Workflow", "type":
"boolean"},
    {"name": "STEP2_PWD_INPUT", "description": "Step 2 Password (SecureString) Input for Workflow",
"type": "password"}
  ],
  "output": [
    {"name":"STEP1_OUTPUT", "description": "Step 1 Output", "type": "string"},
    {"name":"STEP2_OUTPUT", "description": "Step 2 Output", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step1",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "Step 1 - SSH Workflow",
      "inBinding":[
        { "name": "initialDelay", "type": "number", "default": "60" },
        {"name": "username", "type": "string", "exportName": "STEP1_USER"},
        {"name": "password", "type": "password", "exportName": "STEP1_PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "exportName": "STEP1_CMD"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "STEP1_HOSTNAME"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "STEP1_OUTPUT"}
      ],
      "nextStep":"step2"
    },
    {
      "stepId":"step2",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Step 2 - Custom Workflow",
      "inBinding":[
        {"name": "vroWorkflowName", "type": "string", "default": "Custom-vRO-Workflow-Name"},
        {"name": "strInput", "type": "string", "exportName": "STEP2_STR_INPUT"},
        {"name": "numInput", "type": "number", "exportName": "STEP2_NUM_INPUT"},
        {"name": "boolInput", "type": "boolean", "exportName": "STEP2_BOOL_INPUT"},
        {"name": "pwdInput", "type": "password", "exportName": "STEP2_PWD_INPUT"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "STEP2_OUTPUT"}
      ],
      "nextStep":"END"
    }
  ]
}
```

## Variables

Workflow with variables.

```
{
  "id":"using-variables",
  "name": "Example for showcasing Variables",
  "description":"Sample Workflow to show use of Variables - Use Output of Step 1 as Input",
  "version":"1.0",
  "startStep":"step1",
  "variables": [
    {"name":"vnfId", "type": "string"},
    {"name":"VAR_1", "type": "string"}
  ],
  "input": [
    {"name": "STEP1_USER", "description": "Step 1 Username", "type": "string"},
    {"name": "STEP1_PWD", "description": "Step 1 Password", "type": "password"},
    {"name": "STEP1_HOSTNAME", "description": "Step 1 Hostname", "type": "string"},
    {"name": "STEP1_CMD", "description": "Step 1 Command", "type": "string"},
    {"name": "STEP2_NUM_INPUT", "description": "Step 2 Number Input for Workflow", "type": "number"},
    {"name": "STEP2_BOOL_INPUT", "description": "Step 2 Boolean Input for Workflow", "type":
"boolean"},
    {"name": "STEP2_PWD_INPUT", "description": "Step 2 Password (SecureString) Input for Workflow",
"type": "password"}
  ],
  "output": [
    {"name":"FINAL_OUTPUT", "description": "Step 2 Output", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step1",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "Step 1 - SSH Workflow",
      "inBinding":[
        { "name": "initialDelay", "type": "number", "default": "60" },
        {"name": "username", "type": "string", "exportName": "STEP1_USER"},
        {"name": "password", "type": "password", "exportName": "STEP1_PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "exportName": "STEP1_CMD"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "STEP1_HOSTNAME"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "VAR_1"}
      ],
      "nextStep":"step2"
    },
    {
      "stepId":"step2",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Step 2 - Use Output of Step 1 as Input",
      "inBinding":[
        {"name": "vroWorkflowName", "type": "string", "default": "Custom-vRO-Workflow-Name"},
        {"name": "strInput", "type": "string", "exportName": "VAR_1"},
        {"name": "numInput", "type": "number", "exportName": "STEP2_NUM_INPUT"},
        {"name": "boolInput", "type": "boolean", "exportName": "STEP2_BOOL_INPUT"},
        {"name": "pwdInput", "type": "password", "exportName": "STEP2_PWD_INPUT"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "FINAL_OUTPUT"}
      ],
      "nextStep":"END"
    }
  ]
}
```

## Conditions

Workflow with the `if` condition.

```
{
  "id":"using-variables-and-conditions",
  "name": "Example for showcasing Conditions",
  "description":"Sample Workflow to show use of Conditions - If Conditions",
  "version":"1.0",
  "startStep":"step1",
  "variables": [
    {"name":"vnfId", "type": "string"},
    {"name":"CONDITION_VARIABLE", "type": "string"}
  ],
  "input": [
    {"name": "USER", "description": "Username", "type": "string"},
    {"name": "PWD", "description": "Password", "type": "password"},
    {"name": "HOSTNAME", "description": "Hostname", "type": "string"},
    {"name": "SCRIPT_ARG", "description": "Script Argument", "type": "string"}
  ],
  "output": [
    {"name":"FINAL_OUTPUT", "description": "Final Output", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step1",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"decision",
      "description": "Step 1 - Make Decision",
      "inBinding":[
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "/opt/script-1.sh {{SCRIPT_ARG}}"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "HOSTNAME"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "CONDITION_VARIABLE"}
      ],
      "condition": [
        {
          "name": "CONDITION_VARIABLE",
          "type": "string",
          "comparator": "equals",
          "value": "PASS",
          "nextStep": "step2"
        },
        {
          "name": "CONDITION_VARIABLE",
          "type": "string",
          "comparator": "contains",
          "value": "WARN",
          "nextStep": "step3"
        },
        {
          "name": "CONDITION_VARIABLE",
          "type": "string",
          "comparator": "equals",
          "value": "ERROR",
          "nextStep": "END"
        }
      ]
    },
    {
      "stepId":"step2",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "Step 2 - Run if step 1 output is PASS",
      "inBinding":[
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "/opt/scriptPass.sh"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "HOSTNAME"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "FINAL_OUTPUT"}
      ],
      "nextStep":"END"
    },
    {
      "stepId":"step3",
      "workflow":"RUN_SSH_COMMAND_IN_GUEST",
      "namespace": "nfv",
      "type":"task",
      "description": "Step 3 - Run if step 1 output is WARN",
      "inBinding":[
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "22"},
        {"name": "cmd", "type": "string", "default": "/opt/scriptWarn.sh"},
        {"name": "encoding", "type": "string", "default": " "},
        {"name": "hostNameOrIP", "type": "string", "exportName": "HOSTNAME"},
        {"name": "passwordAuthentication", "type": "boolean", "default": "true"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "FINAL_OUTPUT"}
      ],
      "nextStep":"END"
    }
  ]
}
```

# Advanced Workflow Use Cases

You can perform the following advanced operations with vRealize Orchestrator Workflows.

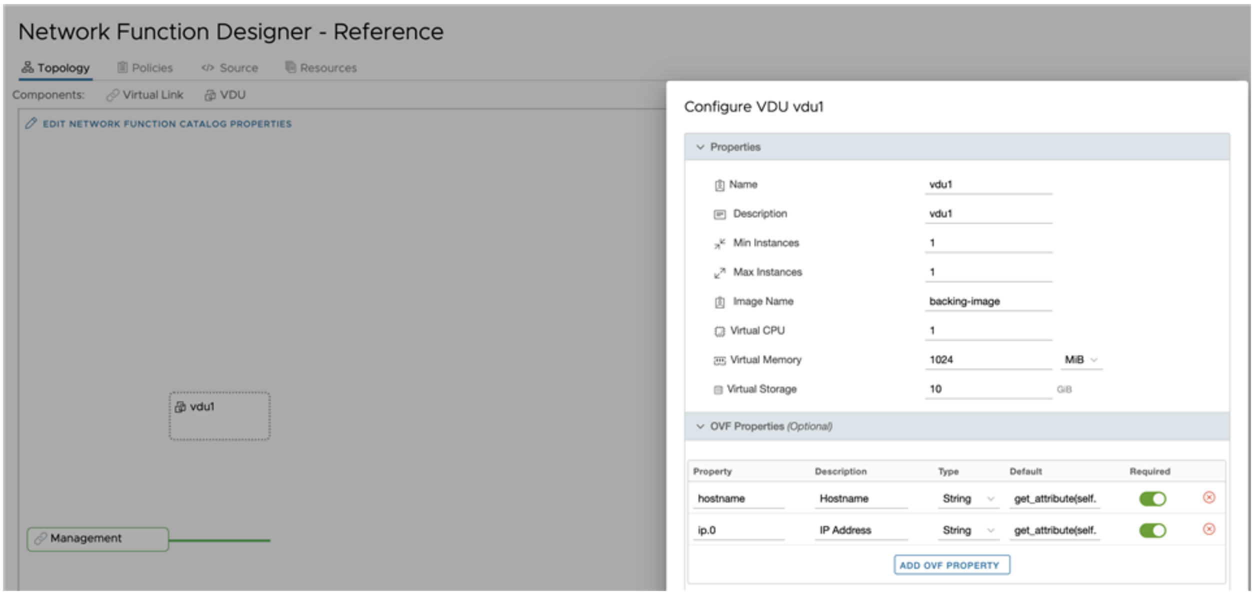
## Referencing Default Values Using the get_attribute Method

You can use `get_attribute` to get the attribute values of each VDU from the JSON file that you upload when designing a Network Function. In a scenario where you create multiple VDUs, each VDU requires a different IP address and a host name. You can parameterize these values in the **OVF Properties** tab using the `get_attribute` method. For example, to get the host name for each VDU, add the following command in the **Default** column when configuring the OVF properties of the VDU.

```
get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU1-HOSTNAME)
```

In this command, VMware Telco Cloud Automation checks for those VNF instances that are in the pre-instantiation state and populates their host names.

You can add the `get_attribute` method in the OVF Properties tab when designing a Network Function Descriptor. For information about designing a Network Function Descriptor, see Designing a Network Function Descriptor.



Configure VDU vdu1

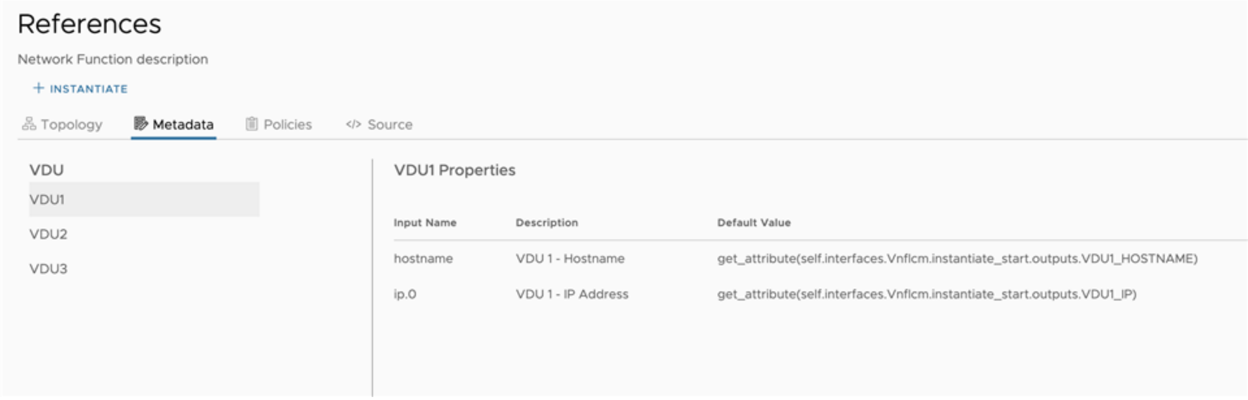
Properties

| | |
|---|---|
| Name | vdu1 |
| Description | vdu1 |
| Min Instances | 1 |
| Max Instances | 1 |
| Image Name | backing-image |
| Virtual CPU | 1 |
| Virtual Memory | 1024 MiB |
| Virtual Storage | 10 GiB |

OVF Properties *(Optional)*

| Property | Description | Type | Default | Required |
|---|---|---|---|---|
| hostname | Hostname | String | get_attribute(self. | ● |
| ip.0 | IP Address | String | get_attribute(self. | ● |

To view the Workflow reference, go to **Network Functions** > **Catalog**, click the network function, and click the **Metadata** tab.



VDU1 Properties

| Input Name | Description | Default Value |
|---|---|---|
| hostname | VDU 1 - Hostname | get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU1_HOSTNAME) |
| ip.0 | VDU 1 - IP Address | get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU1_IP) |

## Examples

The following example is a snippet of a pre-instantiation workflow in the TOSCA YAML file:

```
interfaces:
      Vnflcm:
        instantiate_start:
          implementation: ../Artifacts/workflows/get-env-details.json
          description: Hostname and IP per environment
          inputs:
            type: tosca.datatypes.nfv.VMware.Interface.InstantiateStartInputParameters
            ENVIRONMENT: RDC
          outputs:
            type: tosca.datatypes.nfv.VMware.Interface.InstantiateStartOutputParameters
            VDU1_HOSTNAME: ''
            VDU1_IP: ''
            VDU2_HOSTNAME: ''
            VDU2_IP: ''
            VDU3_HOSTNAME: ''
            VDU3_IP: ''
```

The following example is a snippet of the VMware Telco Cloud Automation - Workflow JSON file:

```
{
  "id":"generate-hostnames-and-ips",
  "name": "Generate Hostnames and IPs",
  "description":"Hostname and IP per environment",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "ENVIRONMENT", "description": "Deployment Environment", "type": "string", "default":
"CDC"}
  ],
  "output": [
    {"name":"VDU1_HOSTNAME", "description": "VDU1 Hostname", "type": "string"},
    {"name":"VDU1_IP", "description": "VDU1 IP Address", "type": "string"},
    {"name":"VDU2_HOSTNAME", "description": "VDU2 Hostname", "type": "string"},
    {"name":"VDU2_IP", "description": "VDU2 IP Address", "type": "string"},
    {"name":"VDU3_HOSTNAME", "description": "VDU3 Hostname", "type": "string"},
    {"name":"VDU3_IP", "description": "VDU3 IP Address", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Get Environment Details",
      "inBinding":[
         {"name": "vroWorkflowName", "type": "string", "default": "Get-Environment-Details"},
         {"name": "envName", "type": "string", "exportName": "ENVIRONMENT"}
      ],
      "outBinding": [
         {"name": "vdu1_hostname", "type": "string", "exportName": "VDU1_HOSTNAME"},
         {"name": "vdu1_ip", "type": "string", "exportName": "VDU1_IP"},
         {"name": "vdu2_hostname", "type": "string", "exportName": "VDU2_HOSTNAME"},
         {"name": "vdu2_ip", "type": "string", "exportName": "VDU2_IP"},
         {"name": "vdu3_hostname", "type": "string", "exportName": "VDU3_HOSTNAME"},
         {"name": "vdu3_ip", "type": "string", "exportName": "VDU3_IP"}
      ],
      "nextStep":"END"
    }
  ]
}
```

The following example is a snippet of how VMware Telco Cloud Automation uses the variables from the `instantiate_start` workflow as reference within the TOSCA YAML file:

```
vdu1:
      type: tosca.nodes.nfv.Vdu.Compute.vdu1
      properties:
        name: vdu1
        ...
        configurable_properties:
          additional_vnfc_configurable_properties:
            type: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.vdu1
            hostname: 
get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU1_HOSTNAME)
            ip.0: get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU1_IP)
      ...
    vdu2:
      type: tosca.nodes.nfv.Vdu.Compute.vdu2
      properties:
        name: vdu2
        ...
        configurable_properties:
          additional_vnfc_configurable_properties:
            type: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.vdu2
            hostname: 
get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU2_HOSTNAME)
            ip.0: get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU2_IP)
      ...
    vdu3:
      type: tosca.nodes.nfv.Vdu.Compute.vdu3
      properties:
        name: vdu3
        ...
        configurable_properties:
          additional_vnfc_configurable_properties:
            type: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.vdu3
            hostname: 
get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU3_HOSTNAME)
            ip.0: get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU3_IP)
```

Referencing default values using the `get_attribute` method is applicable to VNFs and CNFs.

## Scaling VDUs with Instance Count

Take a scenario where you want to scale a VDU to five instances. OVF properties such as Host Name and IP address that are assigned must be different for each of these VDUs. To ensure that unique OVF properties are assigned to each of the VDUs, you can use the *INSTANCE_COUNT* variable in the following ways:

1. *INSTANCE_COUNT* in references or variables - If you have a pre-instantiated workflow that generates OVF properties such as host name for each VDU instance, you can reference it using the following command.

   ```
   get_attribute(self.interfaces.Vnflcm.instantiate_start.outputs.VDU-HOSTNAME-{{INSTANCE_COUNT}})
   ```

   In this command, VMware Telco Cloud Automation replaces *INSTANCE_COUNT* with the corresponding instance count of the VDU.

2. Use *INSTANCE_COUNT* directly in OVF Properties - If you have defined the OVF properties and want the host names and IP addresses to increase by one based on the number of scale instances, use the following command.

   ```
   hostname: centos-{{INSTANCE_COUNT}}
   ipAddress: 192.168.10.{{INSTANCE_COUNT}}
   ```

   When the VDU is deployed or scaled, the actual instance number replaces *INSTANCE_COUNT*. The values of the hostname and IP address in the OVF properties are updated to:

   ```
   Instance 1
   hostname: centos-1
   ipAddress: 192.168.10.1
   Instance 2
   hostname: centos-2
   ipAddress: 192.168.10.2
   Instance 3
   hostname: centos-3
   ipAddress: 192.168.10.3
   Instance 4
   hostname: centos-4
   ipAddress: 192.168.10.4
   Instance 5
   hostname: centos-5
   ipAddress: 192.168.10.5
   ```

**Note** You can add the *INSTANCE_COUNT* reference on OVF properties when performing the following operations:

1. Instantiating a VNF.
2. Scaling or scaling to level a VNF.

Scaling VDUs with Instance Count is applicable only to VNFs.

# Updating NETCONF Protocol Using VMware Telco Cloud Automation

15

Network Configuration Protocol (NETCONF) is a network management protocol developed and standardized by the Internet Engineering Task Force (IETF). By using specific workflows and by uploading the required configuration files to VMware Telco Cloud Automation, you can apply certain configuration changes to your NETCONF environment.

## Supported NETCONF Operations

The following NETCONF operations are supported for VMware Telco Cloud Automation version 1.8:

- `get`
- `getconfig`
- `edit-config`
  - `merge`
  - `replace`

## Prerequisites

Unlike other workflows in VMware Telco Cloud Automation that use VMware vRealize Orchestrator, the NETCONF workflow runs from the NETCONF client that is located within the Telco Cloud Automation Control Plane (TCA-CP) appliance. From a connectivity and firewall perspective, ensure that TCA-CP has access to the NETCONF server IP address and port before running the workflow.

## `getconfig` Workflow Example

You can provide the following possible inputs:

- NETCONF server IP.
- User name for logging in to the NETCONF server.
- Password for logging in to the NETCONF server.
- Port on which the NETCONF server runs.

- If you change the action to `get`, VMware Telco Cloud Automation runs the `get` command on the NETCONF server.

```
{
  "id":"netconf_getconfig_workflow",
  "name": "Netconf Get-Config Workflow",
  "description":"Netconf Get-Config Workflow",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "USER", "description": "Username", "type": "string"},
    {"name": "PWD", "description": "Password", "type": "password"},
    {"name": "HOSTNAME", "description": "Hostname", "type": "string"}
  ],
  "output": [
    {"name":"result", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"NETCONF_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Netconf Get-Config Workflow",
      "inBinding":[
        {"name":"action", "type":"string", "default" : "getconfig"},
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "17830"},
        {"name": "hostname", "type": "string", "exportName": "HOSTNAME"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "result"}
      ],
      "nextStep":"END"
    }
  ]
}
```

## `merge` Workflow Example

You can provide the following possible inputs:

- NETCONF server IP.
- User name for logging in to the NETCONF server.
- Password for logging in to the NETCONF server.
- Port on which the NETCONF server runs.

- XML file containing the configuration to be applied on the NETCONF server.
- If you change the action to `replace`, VMware Telco Cloud Automation runs the `edit-config` command with the `replace` option.

```
{
  "id":"netconf_merge_workflow",
  "name": "Netconf Merge Workflow",
  "description":"Netconf Merge Workflow",
  "version":"1.0",
  "startStep":"step0",
  "variables": [
    {"name":"vnfId", "type": "string"}
  ],
  "input": [
    {"name": "USER", "description": "Username", "type": "string"},
    {"name": "FILENAME", "description": "Filename", "type": "file"},
    {"name": "PWD", "description": "Password", "type": "password"},
    {"name": "HOSTNAME", "description": "Hostname", "type": "string"}
  ],
  "output": [
    {"name":"result", "description": "Output Result", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step0",
      "workflow":"NETCONF_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Netconf Merge Workflow",
      "inBinding":[
        {"name":"action", "type":"string", "default" : "merge"},
        {"name": "inFile", "type": "file", "exportName": "FILENAME"},
        {"name": "username", "type": "string", "exportName": "USER"},
        {"name": "password", "type": "password", "exportName": "PWD"},
        {"name": "port", "type": "number", "default": "17830"},
        {"name": "hostname", "type": "string", "exportName": "HOSTNAME"}
      ],
      "outBinding": [
        {"name": "result", "type": "string", "exportName": "result"}
      ],
      "nextStep":"END"
    }
  ]
}
```

# Monitoring Performance and Managing Faults

16

You can monitor the network functions to track their performance and perform actions based on their CPU utilization and other parameters.

This chapter includes the following topics:

## Managing Alarms

The **Dashboard** tab displays the total number of alarms triggered. It also displays the number of alarms according to their severity.

### VNF Alarms

VNF alarms are triggered when VMware Telco Cloud Automation identifies anomalies in the network connection status or when the power state changes. VMware Telco Cloud Automation also triggers VNF alarms that are predefined and user-defined in VMware vSphere.

### CNF Alarms

CNF triggers alarms for system level and service level anomalies. For example, system level alarms are triggered when an image or resource is not available, or when a pod becomes unavailable. Service level alarms are triggered when the number of replicas that you have specified is not identical to the number of nodes that get created, and so on. Here are some possible anomalies when VMware Telco Cloud Automation displays an error message and triggers an alarm. These alarms are in the **Critical** state:

- `Image pull error` - The URL to the Helm Chart image is incorrect or the image cannot be accessed due to network issues.
- `Crash loop backoff` - The application fails to load.

- `Progress deadline exceeded` - Kubernetes controller exceeds the maximum number of tries to recover a crashed application.
- `Failed create` - Kubernetes controller fails to create or schedule a Kubernetes Pod.
- `Resource failed` - Kubernetes controller fails to create the resources.

## VIM Alarms

VIM alarms are triggered at the VIM level for CNF infrastructure anomalies. For example, when a Kubernetes cluster reaches its memory or CPU resource limit, its corresponding VIM triggers an alarm. Here are some possible CNF infrastructure anomalies for which alarms are triggered. These alarms are in the **Warning** state:

- `Network unavailable` - Worker node is unable to reach the network.
- `PID pressure` - Worker node encounters Process ID (PID) limitations.
- `Disk pressure` - Worker node is running out of disk space.
- `Memory pressure` - Worker node is running out of memory.

## Viewing and Acknowledging Alarms

Alarms are triggered at four levels:

- CNF/VNF level - To view the alarms of individual CNFs and VNF instances, go to the **Inventory** tab, click a VNF or CNF instance, and click **Alarms**.
- Network Service level - VNF and CNF alarms are listed at the corresponding Network Service level.
- VDU level - For a VNF, the alarms are also listed at the corresponding VDU level.
- Global level - You can view the global alarms for all entities and users from the **Administration** > **Alarms** tab.

To view and acknowledge alarms, perform the following steps:

1. Go to **Administration** > **Alarms**. Details of the alarm such as the alarm name, its associated entity, its associated managed object, alarm severity, alarm triggered time, description, and state are displayed.
2. To acknowledge a triggered alarm, select the alarm and click **Acknowledge**. When the acknowledgment is successful, the state of the alarm changes to **Acknowledged**. To acknowledge multiple alarms together, select the alarms that you want to acknowledge and click **Acknowledge**.

By default, the list refreshes every 120 seconds. To get the current state of the alarms, click **Refresh**.

# Performance Management Reports

Performance management reports are useful to monitor the behavior of the network. You can generate performance management reports for a VNF or a CNF instance.

## VNF Reports

You can generate reports for performance metrics such as **Mean CPU Usage** and **Mean Memory Usage** for each VNF. Set the frequency of report collection, end date and time, and the performance metrics that you want to generate reports for.

You can collect the following performance metrics:

- Mean CPU Usage
- Disk Read
- Disk Write
- Mean Memory Usage
- Number of Incoming Bytes
- Number of Outgoing Bytes
- Number of Incoming Packets
- Number of Outgoing Packets

The performance management report includes stats collected at the VNF and VDU levels for a VNF instance.

## CNF Reports

For this release, you can generate only the **Mean CPU Usage** and **Mean Memory Usage** performance metrics reports.

**Note** To generate performance management reports for CNFs, you must install Prometheus Operator on the namespace *vmware-paas* and set the default port to 9090.

## Scheduling Performance Management Reports

Create and schedule a performance management job report for a VNF or a CNF.

### Procedure

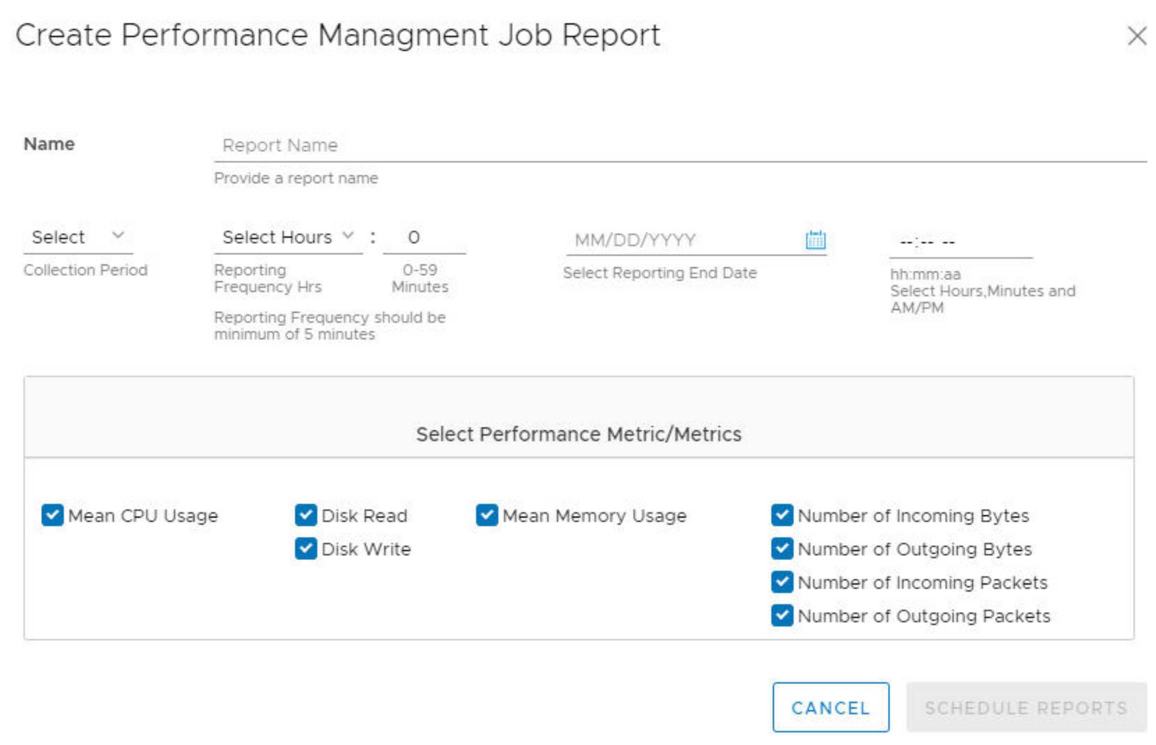
1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Functions** > **Inventory**.
3. Click the desired CNF or VNF, and from the details page click the **PM Reports** tab.
4. Click **Schedule Reports**.

5 In the **Create Performance Management Job Report** window, enter the following details:

- Provide a name for the report.
- Select the collection period time, reporting frequency in hours and minutes, reporting end date and time.

  **Note** The minimum reporting frequency is 5 minutes.

- Select the performance metrics data to collect.



Create Performance Managment Job Report

Name: Report Name
Provide a report name

Select — Collection Period

Select Hours : 0
Reporting Frequency Hrs — 0-59 Minutes
Reporting Frequency should be minimum of 5 minutes

MM/DD/YYYY
Select Reporting End Date

--:-- --
hh:mm:aa
Select Hours,Minutes and AM/PM

Select Performance Metric/Metrics

☑ Mean CPU Usage
☑ Disk Read
☑ Disk Write
☑ Mean Memory Usage
☑ Number of Incoming Bytes
☑ Number of Outgoing Bytes
☑ Number of Incoming Packets
☑ Number of Outgoing Packets

6 Click **Schedule Reports**.

The report is scheduled and is available under **PM Reports** in the details page. It stays active from the current time stamp until the provided end time.

7 To download the generated report, click the **More** (**>**) icon against your report name and click **Download**.

The report is downloaded to your system in the CSV format.

**Note** You can only download those reports that are in the **Available** state. The generated reports are available for download for 7 days.

# Monitor Instantiated Virtual Network Functions and Virtual Deployment Units

After you instantiate a Virtual Network Function (VNF), you can monitor its performance metrics and take corrective actions.

### Prerequisites

**Note** This procedure is not supported for network functions that are imported from partner systems.

### Procedure

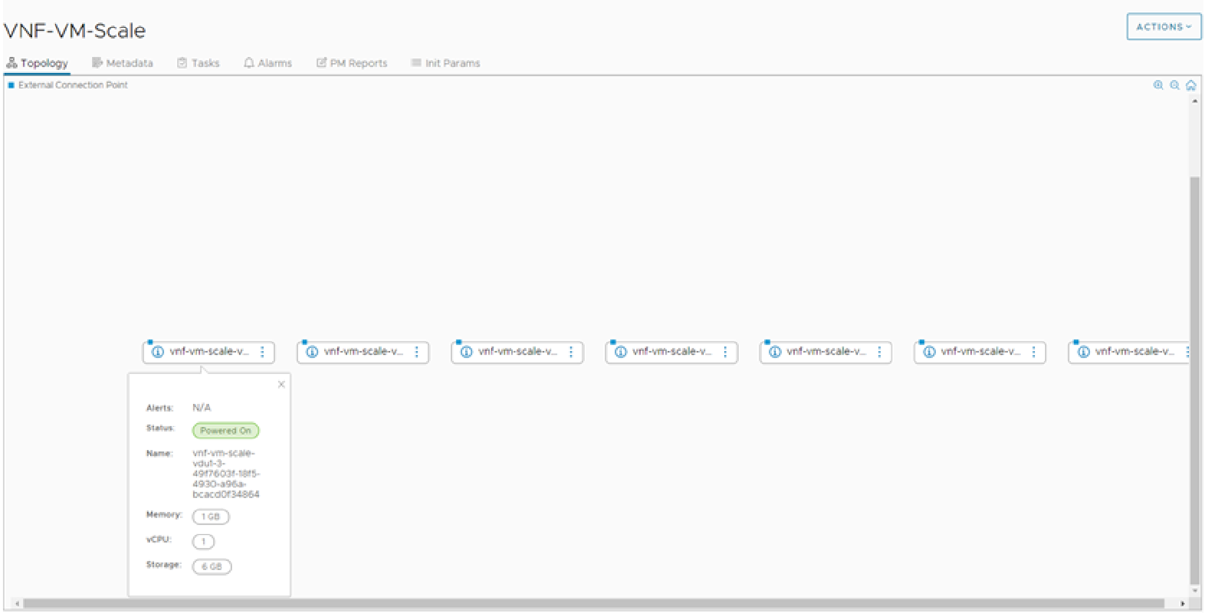
1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Inventory**.

3 Click the desired network function to monitor.

   The network function topology is displayed.

4 Perform the desired monitoring or management actions:



VNF-VM-Scale

Alerts: N/A
Status: Powered On
Name: vnf-vm-scale-vdu1-3-49f7603f-18f5-4930-a96a-bcacd0f34864
Memory: 1 GB
vCPU: 1
Storage: 6 GB

- To view more details of a Virtual Deployment Unit (VDU) such as alerts, status, name, memory, vCPU, and storage, click the **i** icon.
- To view more information about the virtual link, point at the blue square icon on the VDU.
- To view detailed information about the VDU and the VNFs, their performance data, alarms, and reports, click the ⋮ icon on the desired VDU and click **Summary**. The details page provides the following tabs:
  - **Summary** - Provides a detailed summary of the VDU.

  - **Alarms** - Lists the alarms generated for the VDUs of the selected VNFs. You can acknowledge alarms from here.
  - **Performance Monitoring** - Provides a graphical view of the performance metrics for CPU, Network, Memory, and Virtual Disk. For example, to view more information about the CPU performance, click **CPU**. For an overview of all metrics, click **Overview**. The performance metrics captured here are live with an interval of 1 hour.
  - **Reports** - To set parameters for generating performance reports, click **Schedule Reports**. You can generate historic reports for a metric group, set the collection period, reporting period, and the reporting end date.
- To view historical tasks for a desired network function, go to **Network Functions** > **Inventory** and click the desired network function. The **Tasks** tab displays the historical tasks and their status.

# Monitor Instantiated CNF

After you instantiate a Containerized Network Function (CNF), you can monitor its performance metrics and take corrective actions.

### Prerequisites

**Note** This procedure is not supported for network functions that are imported from partner systems.

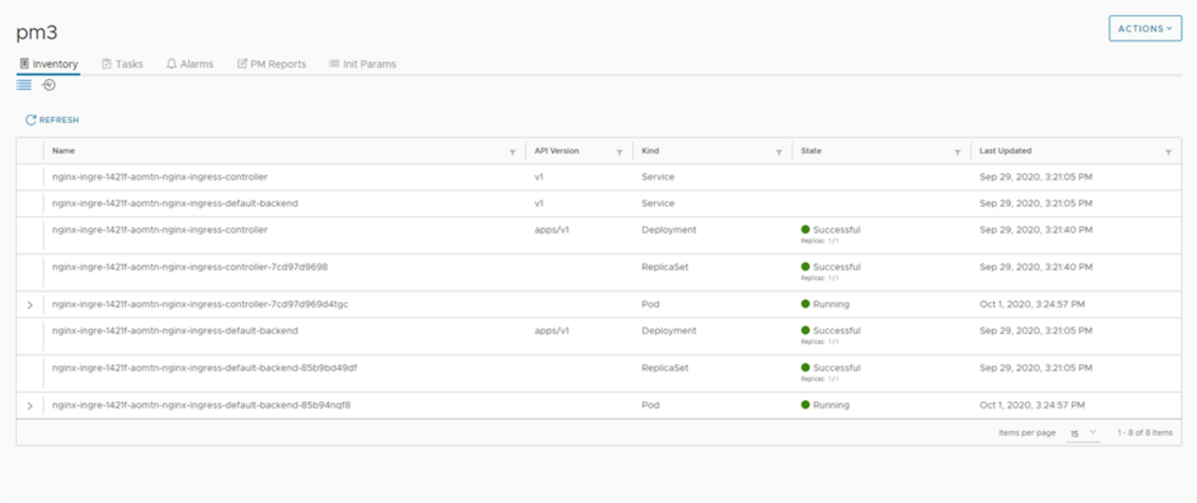
### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Select **Network Functions** > **Inventory**.

3 Click the desired CNF to monitor.

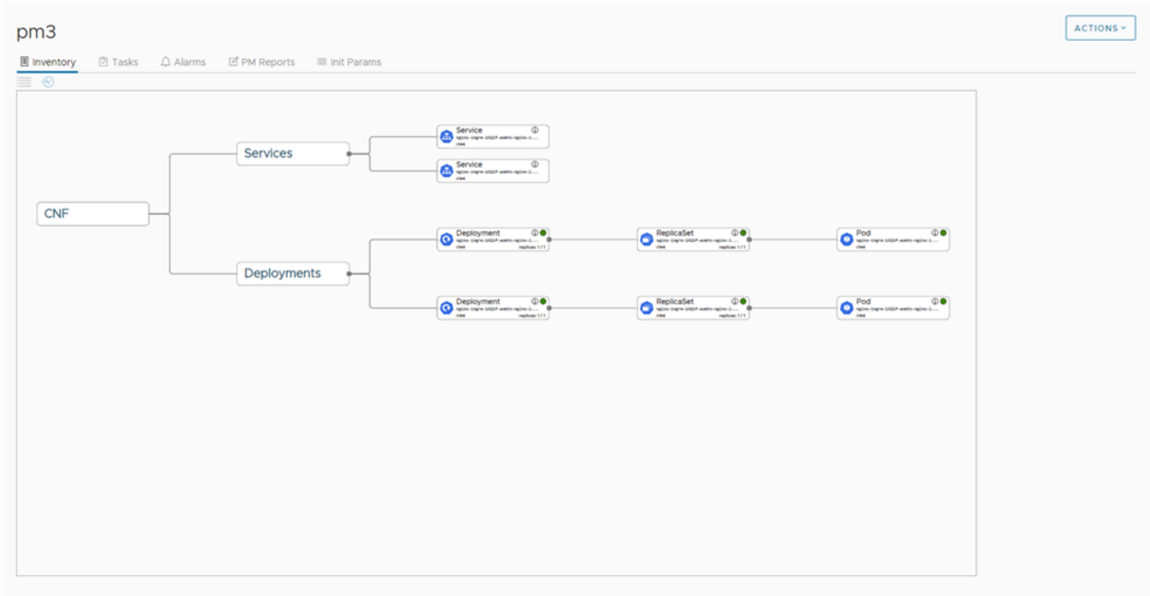
The following tabs are displayed:



| Name | API Version | Kind | State | Last Updated |
|---|---|---|---|---|
| nginx-ingre-1421f-aomtn-nginx-ingress-controller | v1 | Service | | Sep 29, 2020, 3:21:05 PM |
| nginx-ingre-1421f-aomtn-nginx-ingress-default-backend | v1 | Service | | Sep 29, 2020, 3:21:05 PM |
| nginx-ingre-1421f-aomtn-nginx-ingress-controller | apps/v1 | Deployment | ● Successful<br>Replicas: 1/1 | Sep 29, 2020, 3:21:40 PM |
| nginx-ingre-1421f-aomtn-nginx-ingress-controller-7cd97d9698 | | ReplicaSet | ● Successful<br>Replicas: 1/1 | Sep 29, 2020, 3:21:40 PM |
| nginx-ingre-1421f-aomtn-nginx-ingress-controller-7cd97d969d4tgc | | Pod | ● Running | Oct 1, 2020, 3:24:57 PM |
| nginx-ingre-1421f-aomtn-nginx-ingress-default-backend | apps/v1 | Deployment | ● Successful<br>Replicas: 1/1 | Sep 29, 2020, 3:21:05 PM |
| nginx-ingre-1421f-aomtn-nginx-ingress-default-backend-85b9bd49df | | ReplicaSet | ● Successful<br>Replicas: 1/1 | Sep 29, 2020, 3:21:05 PM |
| nginx-ingre-1421f-aomtn-nginx-ingress-default-backend-85b94nqf8 | | Pod | ● Running | Oct 1, 2020, 3:24:57 PM |

Items per page 15 1 - 8 of 8 items

- **Inventory** - Displays the summary of the status of the pods, deployments, and services. To display the tree view, click the tree icon below the **Inventory** tab.



- **Tasks** - Displays historical tasks for the CNF.
- **Alarms** - Lists the alarms generated for the selected CNFs. You can acknowledge alarms from here.
- **PM Reports** - Displays the list of performance reports that are being collected. To set parameters for generating performance reports, click **Generate Reports**. You can generate reports for a metric group, set the collection period, reporting period, and the reporting end date.
- **Init Params** - Displays the input parameters for the CNF.

# Monitor Instantiated Network Services

After you instantiate a network service, you can view its topology and task information from the VMware Telco Cloud Automation web interface.

### Prerequisites

Instantiate the network service.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Select **Network Services** > **Inventory**.
3. Click the desired network service to monitor.

   The network service topology is displayed.
4. Perform the desired monitoring or management actions:
   - To view and acknowledge the consolidated alarms of all the VNFs and CNFs that belong to the network service, click the **Alarms** tab. You can also view the alarms from the **Topology** tab. Click the **More** (...) icon on the desired network service and select **Alarms**.
   - View historical tasks for the selected network service from the **Tasks** tab.

# Administrating VMware Telco Cloud Automation 17

View logs and download them for auditing and troubleshooting.

This chapter includes the following topics:

## Viewing Audit Logs

If an error occurs, you can review the logs and take corrective actions on your deployment. Or, you can download the logs for auditing purposes.

To view or download audit logs, go to **Administration** > **Audit Logs**.

Log entries from the specified time period are displayed in the table. You can click **Download Audit Logs** to download a copy of the displayed logs to your local machine.

## Troubleshooting and Support

If VMware Telco Cloud Automation does not operate as expected, you can create a support bundle that includes logs and database files for analysis.

Go to **Administration** > **Troubleshooting** and click **Request** to generate a support bundle.

If you intend to contact VMware support, go to **Administration** > **Support** and copy the support information to your clipboard. This information is required in addition to the support bundle.

# Upgrading VMware Telco Cloud Automation 18

When a new version of VMware Telco Cloud Automation or VMware Telco Cloud Automation Control Plane (TCA-CP) becomes available, you can update your deployment from the web interface.

This chapter includes the following topics:

## Upgrade VMware Telco Cloud Automation

VMware Telco Cloud Automation notifies you when a newer version is available. You can then go to **Administration** > **System Updates** and upgrade to the newer version.

If you miss an upgrade notification and want to check for a newer version of VMware Telco Cloud Automation, perform the following steps:

### Prerequisites

- Ensure that VMware Telco Cloud Automation is connected to the activation server.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Go to **Administration** > **System Updates**.
3. In the **Available Service Update Versions** column, click **Check For Updates**. This option is useful when you miss a notification from VMware Telco Cloud Automation about a service update and want to upgrade to the latest version.

   **Note** The **Check For Updates** option is available only from VMware Telco Cloud Automation version 1.8 onwards.

4. When an update is available, you can perform one of the following operations:
   - **Download** - To download the upgrade bundle from the public URL and stores it in the VMware Telco Cloud Automation node, click **Download**.

- **Upgrade** - After downloading the upgrade bundle, the **Upgrade** operation is enabled. To upgrade your VMware Telco Cloud Automation version, click **Upgrade**.
- **Download & Upgrade** - This option combines the download and upgrade operations.
- **Readme** - This option opens the release notes for the particular build.

# Upgrade VMware Telco Cloud Automation Control Plane

VMware Telco Cloud Automation notifies you when a new version of VMware Telco Cloud Automation Control Plane (TCA-CP). You can then go to **Administration** > **System Updates** and upgrade to the newer version.

If you miss an upgrade notification and want to check for a newer version of TCA-CP, perform the following steps:

### Prerequisites

- Ensure that VMware Telco Cloud Automation is connected to the activation server.

### Procedure

1. Log in to the VMware Telco Cloud Automation Control Plane web interface.
2. Go to **Administration** > **System Updates**.
3. In the **Available Service Update Versions** column, click **Check For Updates**. This option is useful when you miss a notification from TCA-CP about a service update and want to upgrade to the latest version.

   **Note** The **Check For Updates** option is available only from VMware Telco Cloud Automation version 1.8 onwards.

4. When an update is available, you can perform one of the following operations:
   - **Download** - To download the upgrade bundle from the public URL and stores it in the TCA-CP node, click **Download**.
   - **Upgrade** - After downloading the upgrade bundle, the **Upgrade** operation is enabled. To upgrade your TCA-CP version, click **Upgrade**.
   - **Download & Upgrade** - This option combines the download and upgrade operations.
   - **Readme** - This option opens the release notes for the particular build.

# Scheduling TCA-CP Upgrades 19

VMware Telco Cloud Automation enables you to schedule upgrades for TCA-CP systems installed at Telco Cloud sites through a centralized management interface.

The TCA-CP Upgrade Manager interface allows you to upgrade those TCA-CP appliances that are connected to VMware Telco Cloud Automation. To upgrade VMware Telco Cloud Automation to a newer version, see Upgrade VMware Telco Cloud Automation.

Using the TCA-CP Upgrade Manager interface provided by Telco Cloud Automation, you organize individual TCA-CP systems into logical groups. You then apply an upgrade schedule to each group. The group schedule is synchronized down to the TCA-CP systems associated with that group. When the schedule occurs, the individual TCA-CP systems in that group automatically begin the upgrade process. Each system provides its upgrade status back to the TCA-CP Upgrade Manager.

**Note** Because the TCA-CP systems in a group are distributed across the Telco environment, the upgrade status reported by individual systems can vary depending on things such as network performance.

Through group upgrade scheduling, Telco Cloud Automation can simplify and automate upgrade operations across your Telco environment.

This chapter includes the following topics:

## Creating Upgrade Groups

Managing upgrades is made simpler by creating logical groups for TCA-CP systems.

### Prerequisites

- Group names must be unique.
- The system being added must not be part of another group.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Navigate to **Administration > TCA-CP Upgrade Manager** and click **TCA-CP Systems**.

   The list of TCA-CP systems that are paired with VMware Telco Cloud Automation appears in the window.

3 Select the TCA-CP systems that you want to group for upgrading.

   A TCA-CP system can be part of only one group. Group creation fails if you select TCA-CP systems that are already associated with a group.

   **Note** You can add additional TCA-CP systems to an existing group at any time. To add more systems to a group, navigate to **TCA-CP Systems**, select the specific systems, and click **Add to Group**.

4 Click **Create Group**.

5 Enter a group name and click **Create**.

   The group name is associated with the selected TCA-CP connected systems.

### Results

The systems associated with VMware Telco Cloud Automation are now logically grouped for upgrade operations.

You can view the group name associated with a specific TCA-CP system from **TCA-CP Upgrade Manager > TCA-CP Systems**. Or, you can review the list of systems available in each group at **TCA-CP Upgrade Manager > TCA-CP Groups** and select a group name.

**Note** You can delete a group at any time. To remove a group, first delete the schedule associated with that group. Then navigate to **TCA-CP Groups**, select a group name, and click **Delete**.

### What to do next

Create an upgrade schedule for the group.

## Scheduling Group Upgrades

Scheduling upgrades for groups of TCA-CP systems simplifies maintenance across Telco clouds.

### Prerequisites

- You have created the upgrade groups.
- Upgrade schedule planning has been done for the group. You can have only one schedule per group.

Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Navigate to **Administration > TCA-CP Upgrade Manager** and click **TCA-CP Groups**.

The system displays the list of groups.

3 Click a group name.

The group information window lists the TCA-CP systems belonging to the group and displays the group schedule information. The window includes two charts depicting the overall upgrade and schedule status for all group members.

4 In the **Schedules** section, click **Add Schedule**.

The Add Schedule window appears.

5 Enter the schedule information.

| Parameter | Description |
|---|---|
| **Name** | Enter a logical name for the schedule. |
| **Group ID** | This information is internally generated and set by default. |
| **Maintenance Window** | Set the time frame for the system upgrade. The default setting is 2 hours.<br>If the download for a specific system exceeds the window, the download operation can fail. You can increase the maintenance window to allow more time for the upgrade. The change is applied at the next scheduled upgrade time. Alternatively, you can manually upgrade individual systems by accessing the TCA-CP service interface (https://tca-cp-or-fqdn) and navigating to the **Administration > Upgrade** screen. |
| **Enable** | Use the slider to enable or disable the schedule. Disabling the schedule does not remove it. |
| **Time Zone** | Use the pull-down menu to select your time zone. Settings are in the GMT format only. |

6 Enter the schedule frequency information.

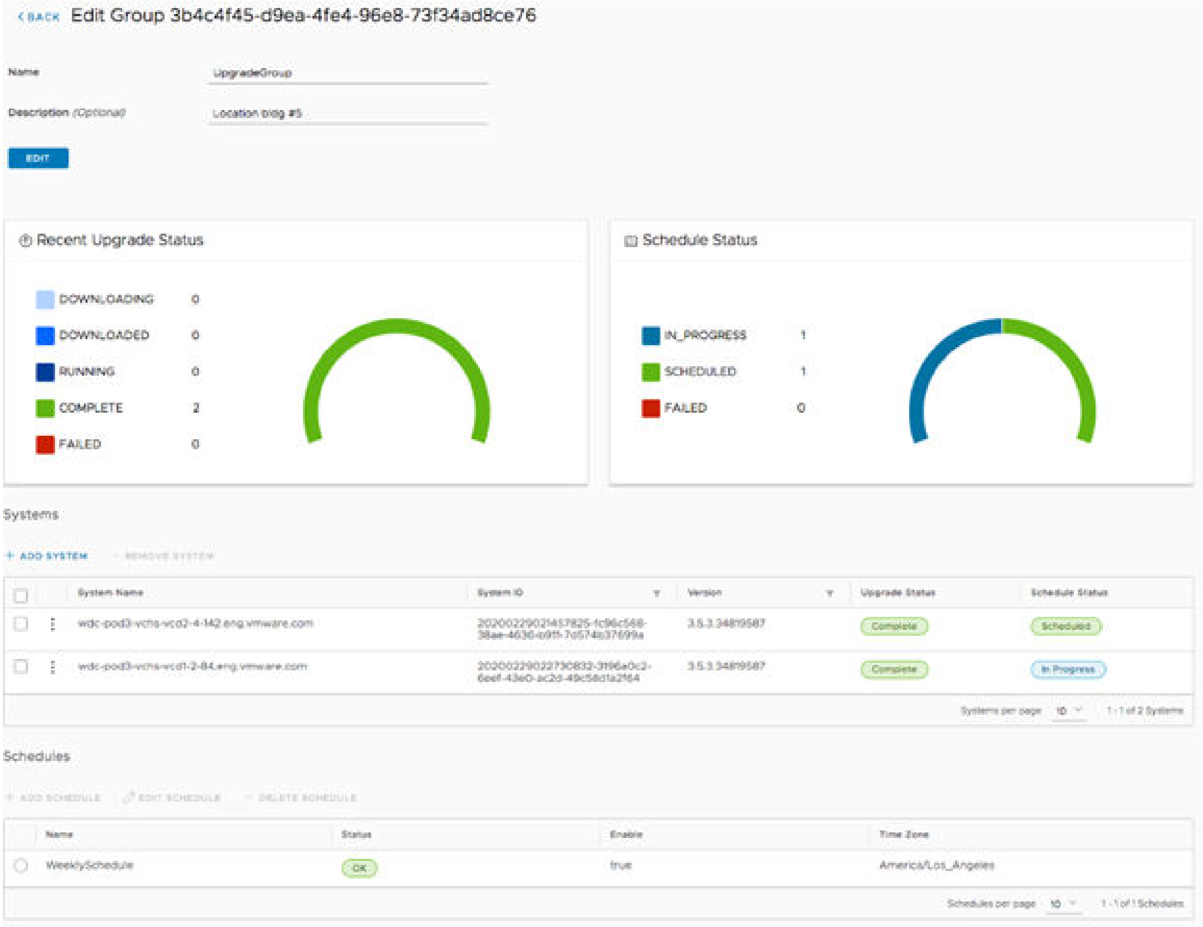
You can schedule group upgrades either by week or month.

| Schedule Selection | Settings |
|---|---|
| Weekly | Enter the day of the week and time that you want the upgrade to occur each week. |
| Monthly | You have two options when selecting a monthly schedule:<br>■ Update on a specific date in a month<br>■ Update on a specific day in a month<br>Enter the date or day, the monthly interval, and the time of day that you want the upgrade to occur.<br>**Note** Create schedule intervals that adhere system support policies. |

The system displays your selections as a CRON expression.

7 Click **Add**.

The TCA-CP Upgrade Manager in Telco Cloud Automation applies the schedule information to each TCA-CP system in the group. The time that this takes can vary depending on your environment. To verify upgrade operations, you can monitor the activity stream for individual TCA-CP systems, or for the overall group.



< BACK Edit Group 3b4c4f45-d9ea-4fe4-96e8-73f34ad8ce76

Name: UpgradeGroup

Description (Optional): Location bldg #5

EDIT

Recent Upgrade Status

| | |
|---|---|
| DOWNLOADING | 0 |
| DOWNLOADED | 0 |
| RUNNING | 0 |
| COMPLETE | 2 |
| FAILED | 0 |

Schedule Status

| | |
|---|---|
| IN_PROGRESS | 1 |
| SCHEDULED | 1 |
| FAILED | 0 |

Systems

+ ADD SYSTEM   – REMOVE SYSTEM

| □ | | System Name | System ID | Version | Upgrade Status | Schedule Status |
|---|---|---|---|---|---|---|
| □ | ⋮ | wdc-pod3-vchs-vcd2-4-142.eng.vmware.com | 20200229021457825-fc96c568-38ae-4636-b911-7d574b37699a | 3.5.3.34819587 | Complete | Scheduled |
| □ | ⋮ | wdc-pod3-vchs-vcd1-2-84.eng.vmware.com | 20200229022730832-3196a0c2-6eef-43e0-ac2d-49c58d1a2f64 | 3.5.3.34819587 | Complete | In Progress |

Systems per page 10   1 - 1 of 2 Systems

Schedules

+ ADD SCHEDULE   EDIT SCHEDULE   – DELETE SCHEDULE

| | Name | Status | Enable | Time Zone |
|---|---|---|---|---|
| ○ | WeeklySchedule | OK | true | America/Los_Angeles |

Schedules per page 10   1 - 1 of 1 Schedules

### Results

The group scheduling is complete.

**Note** You can edit or delete a schedule at any time by selecting **Edit** or **Delete** in the Schedule section of the group information.

### What to do next

Monitor the TCA-CP system upgrades and activities.

## Monitoring Group Upgrades

You can monitor the upgrade status for TCA-CP systems through the TCA-CP Upgrade Manager interface in VMware Telco Cloud Automation.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Navigate to **Administration > TCA-CP Upgrade Manager** and click **TCA-CP Groups**.

   The system displays the list of groups.
3. Click a group name.

   The group information window lists the TCA-CP systems belonging to the group. The system information includes the System Name, System ID, Version, Upgrade Status, and Schedule Status.

4 Review the Recent Upgrade Status and Schedule Status charts for an overview of the upgrade progress.

**Note** The status is empty if the TCA-CP system has no upgrade history.

| Option | Description |
|---|---|
| **Status Chart** | Description |
| **Upgrade Status** | Provides the status of the most recent scheduled upgrade.<br>■ Downloading - Indicates the number of systems that have started the download.<br>■ Downloaded- Indicates the number of systems that have completed the download.<br>■ Running - Specifies the number of systems installing the upgrade.<br>■ Complete - Indicates the number of systems that fully completed the upgrade.<br>■ Failed - Specifies how many systems were not upgraded during the scheduled time. |
| **Schedule Status** | Provides the schedule synchronization status. When you add a group schedule, it is updated on each of the systems belonging to the group.<br>■ In-Progress - Identifies the number of systems receiving the schedule information.<br>■ Scheduled - Specifies the number of systems that have been updated with the latest schedule.<br>■ Failed - Indicates the number of systems the were not updated with the latest schedule. |

5 Review the individual system information for more a detailed look at the TCA-CP system upgrade history and activity.

a Go to the **Systems** section of the page and review the **Upgrade Status** and **Schedule Status** columns for the member system.

b Click the menu (vertical ellipsis) icon preceding the TCA-CP system entry.

A pop-up window appears with selections for **Upgrade History** and **Activity History**.

c Choose from the selections to review TCA-CP system information:

| System History | Description |
|---|---|
| **Upgrade** | Provides the history of software version updates on the TCA-CP system. The Status column lists the current state of the upgrade: Downloading, Downloaded, Complete, Running, Failed.<br>**Note** One cause of a failure might be that the maintenance window was not long enough to complete the upgrade operation. Check the upgrade history start and stop time, and messages for more information. |
| **Activity** | Provides status information about schedule updates applied on the system from TCA-CP Upgrade Manager.<br>■ Type - Identifies the schedule change applied on the TCA-CP system.<br>■ Status - Indicates the state of the change applied on the system: Initiated, Running, Failed, Complete.<br>**Note** A failure can indicate that the system was down or unresponsive. Check the connectivity of the systems using **Link Last Communicated**.<br>■ Message - Provides additional information about the change status.<br>■ Schedule Name - Identifies the name of the schedule in the group that originated the change.<br>■ Timezone and Frequency - Provides information about the schedule. |

# Deleting a Schedule

For a specific Upgrade Group, the upgrade schedule for that group is synchronized to each member system. When a group schedule is removed, it must be removed not only from TCA-CP Upgrade Manager, but also from each member TCA-CP system of the group.

When you delete a schedule, TCA-CP Upgrade Manager immediately marks that schedule for deletion and then advertises the deletion to its members. Each member system works to remove the schedule. This entire process usually takes just a few minutes.

If a system member of the group is not reachable or not responding, deleting the schedule may fail. In this case, the member system retains the schedule information although it is deleted from the TCA-CP Upgrade Manager. You can use the **Force Delete** option to try to force deleting the schedule from a member system.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Navigate to **Administration > TCA-CP Upgrade Manager** and click **TCA-CP Groups**.

   The system displays the list of groups.

3 Click a group name.

   The group information window lists the TCA-CP systems belonging to the group and displays the group schedule information.

4 In the **Schedules** section, select the schedule that you want to delete, and click **Delete Schedule**.

   A pop-up screen appears prompting you to verify the deletion.

5 Click **Delete**.

   The system updates the Schedules status column with the progress of the operation: `Deletion in Progress` or `Failed to Delete`. Deleting a schedule can fail if any one of the member systems fails to delete the schedule or is not reachable. To determine which system failed to remove the schedule, review the Schedule Status column under the Systems information.

6 (Optional) If deleting the schedule is unsuccessful, click **Delete Schedule** again, select the **Force Delete** option, and click **Delete**.

   The Force Delete option is a best effort attempt by TCA-CP Upgrade Manager to delete the schedule, even when a system is not reachable. You can repeat using the Force Delete option until the operation is successful.

# Global Settings for Cluster Automation

# 20

VMware Telco Cloud Automation allows you to configure certain cluster automation settings.

You can configure the behavior for virtual machine placement, update the supported hardware version of `vfio-pci` device drivers, and update the wait time and poll intervals for customization tasks. Configure these settings to change the default behavior only when there is an issue with your existing environment.

This chapter includes the following topics:

## API for Cluster Automation Global Settings

Run the following API on the relevant VMware Telco Cloud Automation Manager or VMware Telco Cloud Automation Control Plane (TCA-CP) instance.

### API

```
PUT: /admin/hybridity/api/global/settings/<namespace>/<property>
{
   "value": <value>
}
```

**Note** The authentication is the same as the other VMware Telco Cloud Automation APIs.

## Configure Cluster Automation Settings

You can configure the following cluster automation settings listed in this section.

### Hardware Version for VFIO PCI Driver

VMware Tanzu Kubernetes Grid deploys a template with the virtual machine hardware version 13 by default. If your network function uses the VFIO PCI driver, it requires the hardware version 14 for an Intel-based setup and version 18 for an AMD-based setup. VMware Telco Cloud Automation updates this information to the VMConfig Operator according to the firmware. Currently, VMware Telco Cloud Automation updates the hardware version as 14 assuming that the setup is Intel-based. If you are using an AMD-based setup, use the following API to update the global settings to send the hardware version as 18.

### Prerequisites

Run this API on VMware Telco Cloud Automation Manager.

### API

```
PUT: /admin/hybridity/api/global/settings/InfraAutomation/vfioPciHardwareVersion
{
    "value": "18"
}
```

**Note**

- Update the appropriate value based on the firmware.

## Enable Virtual Machine Placement in vSphere DRS

During customization, the VMConfig plug-in tries to fit the virtual machines to the correct NUMA nodes of the ESXi server based on its SR-IOV, Passthrough, and Pinning specifications. To disable the virtual machine placement operation by the plug-in and enable the virtual machine placement by vSphere DRS, configure the following settings on the node pool before instantiating the network function.

### Prerequisites

Run this API on VMware Telco Cloud Automation Manager.

### API

```
PUT: /admin/hybridity/api/global/settings/InfraAutomation/<nodePoolId>_enableDRS
  {
      "value": true
   }
```

**Note**

- Ensure that you replace all the hyphens (-) in `<nodePoolId>` with underscores (_).

## Update vCPU and Memory Reservation During Virtual Machine Placement

During customization, VMware Telco Cloud Automation reserves 2 vCPUs and 512 MB of memory for the ESXi host while performing the CPU or memory pinning operation on the virtual machines. You can update this default configuration and update the VMware ESXi host information before instantiating your network function.

### Prerequisites

Run this API on VMware Telco Cloud Automation Manager.

### Update the vCPU Reservation

```
PUT: /admin/hybridity/api/global/settings/InfraAutomation/reservedCoresPerNumaNode
  {
      "value": 3
   }
```

**Note** Enter the new reservation value.

### Update the Memory Reservation

```
PUT: /admin/hybridity/api/global/settings/InfraAutomation/reservedMemoryPerNumaNode
  {
      "value": 1024
  }
```

**Note** Enter the new reservation value in MB.

### Update the VMware ESXi Host Information

After updating the vCPU or memory reservation, run the following API:

```
PUT: /hybridity/api/infra/k8s/clusters/<workloadclusterId>/esxinfo
  {
  }
```

## Update Wait Timeout for Customization Tasks

VMware Telco Cloud Automations posts the customizations to VMConfig and polls for the customization tasks to complete. By default, VMware Telco Cloud Automation waits for 30 minutes before polling at 30-second intervals if the customization is not successful. You can configure this default behavior using the following APIs.

### Prerequisites

Run this API on VMware Telco Cloud Automation Manager.

### Number of Polls

By default, VMware Telco Cloud Automation polls 60 times. If your customization requires a longer time to complete, you can increase the poll count.

```
PUT: /admin/hybridity/api/global/settings/InfraAutomation/nodePolicyStatusRetryCount
  {
      "value": 120
  }
```

## Wait Interval Between Each Poll

By default, VMware Telco Cloud Automation waits for 30 seconds between successful polls. If your customization requires a longer time to complete, you can increase the poll interval.

```
PUT: /admin/hybridity/api/global/settings/InfraAutomation/nodePolicyStatusWaitTime
  {
      "value": 120
  }
```

## Number of Retries on Failure

On failure, VMware Telco Cloud Automation retries the customization task up to 10 times. You can increase the retry count.

```
PUT: /admin/hybridity/api/global/settings/InfraAutomation/nodePolicyStatusFailureRetryCount
  {
      "value": 30
  }
```

# Registering Partner Systems

21

You can register third-party partner systems with VMware Telco Cloud Automation for managing VNFs.

You can also register third-party cloud-native repositories such as Harbor for managing CNFs.

This chapter includes the following topics:

## Add a Partner System to VMware Telco Cloud Automation

Add a partner system such as to VMware Telco Cloud Automation.

To add a partner system, perform the following steps:

### Prerequisites

**Note** You must add at least one vCloud Director cloud to your VMware Telco Cloud Automation environment before adding a partner system.

You must have the **Partner System Admin** privileges to perform this task.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Navigate to **Infrastructure** > **Partner Systems** and click **Register**.

3 In the Register Partner System page, select the partner system page and enter the appropriate information for registering the partner system.

4 Click **Next**.

5 Associate one or more VIMs to your partner system.

6 Click **Finish**.

### Results

The partner system is added to VMware Telco Cloud Automation and is displayed in the Partner Systems page.

### Example

In this example, we list the steps to add Nokia CBAM as a partner system:

1. Log in to the VMware Telco Cloud Automation web interface.
2. Navigate to **Infrastructure** > **Partner Systems** and click **Register**.
3. In the Register Partner System page, Nokia CBAM is preselected. Enter the following information:
   - **Name** - Enter a unique name to identify the partner system in VMware Telco Cloud Automation.
   - **Version** - Select the version of the partner system from the drop-down menu.
   - **URL** - Enter the URL to access the partner system.
   - **Secret** - Enter the secret pass code for the client.

     **Note** You can get the Client ID and Secret from Nokia CBAM.
   - **Trusted Certificate (Optional)** - Paste the contents of the certificate.
4. Click **Next**.
5. Associate one or more VIMs to your partner system.
6. Click **Finish**.

Nokia CBAM is added to VMware Telco Cloud Automation and is displayed in the Partner Systems page.

### What to do next

- You can select the partner system and click **Modify Registration** or **Delete Registration** to edit the configuration or remove the system from VMware Telco Cloud Automation.
- You can add a network function catalog from the partner system to VMware Telco Cloud Automation.

## Edit a Registered Partner System

After registering, you can edit the partner system details and its associated VIMs.

### Prerequisites

You must have the **Partner System Admin** privileges to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Navigate to **Infrastructure** > **Partner Systems** and select the partner system that you want to edit.
3. Click **Modify Registration**.
4. In the **Credentials** tab, edit the partner system details.
5. Click **Next**.
6. Select additional VIMs or deselect the VIMs that you do not want to associate your partner system with.
7. Click **Finish**.

### Results

The partner system details are updated.

### What to do next

To view the updated details of your partner system, go to **Infrastructure** > **Partner Systems**, select your partner system, and click the **>** icon.

## Associate a Partner System Network Function Catalog

VMware Telco Cloud Automation can orchestrate a network function catalog from a partner system. Add the partner system's network function catalog to VMware Telco Cloud Automation.

### Prerequisites

You must have the **Partner System Admin** privileges to perform this task.

### Procedure

1. Log in to the VMware Telco Cloud Automation web interface.
2. Navigate to **Infrastructure** > **Partner Systems** and select the partner system.
3. Click **Add Network Function Catalog**.
4. In the Add Network Function Catalog page, enter the following details:
   - **Descriptor ID** - The descriptor ID of the network function catalog.
   - **Product Name** - The name of the product associated with the network function catalog.
   - **Software Version** - The software version of the partner system.
   - **Descriptor Version** - The version number of the network descriptor.
5. Click **Add**.

### Results

The network function catalog is added to the **Network Functions** > **Catalogs** page.

**Note** You cannot edit the Network Function Description of a network function catalog that is added from a partner system.

## Add a Harbor Repository

Add a Harbor repository to VMware Telco Cloud Automation.

### Prerequisites

You must have the **Partner System Administrator** privileges to perform this task.

**Note** Ensure that all Harbor repository URLs contain the appropriate port numbers such as 80, 443, 8080, and so on.

### Procedure

1 Log in to the VMware Telco Cloud Automation web interface.

2 Navigate to **Infrastructure** > **Partner Systems** and click **Register**.

3 Select **Harbor**.

4 Enter the following details:

   - **Name** - Provide a name for your repository.
   - **Version** - Select the Harbor version from the drop-down menu.
   - **URL** - Enter the URL of your repository. If you use a Harbor repository from a third-party application, ensure that you provide this URL in VMware Telco Cloud Automation Control Plane (TCA-CP).
   - To trust the certificates provided by Harbor, select **Trust Certificate**.
   - **Username** and **Password** - Provide the credentials to access your repository.

5 Click **Next**.

6 Associate one or more VIMs to your Harbor repository.

7 Click **Finish**.

### Results

You have successfully registered your Harbor repository. You can now select this repository for resources when instantiating a CNF.