

VMware Telco Cloud Automation User Guide

VMware Telco Cloud Automation 1.5
VMware Telco Cloud Manager 3.5.3
VMware HCX for Telco Cloud 3.5.3

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

Copyright © 2020 VMware, Inc. All rights reserved. [Copyright and trademark information.](#)

Contents

- 1 Introduction 6**
 - Common Abbreviations 6
 - Deployment Architecture 9

- 2 Getting Started 11**
 - Viewing the Dashboard 11

- 3 Managing Roles and Permissions 13**
 - Enabling Users and User Groups to Access VMware Telco Cloud Automation 13
 - Object Level Access Permissions 14
 - Privileges and Roles 14
 - Creating Roles and Permissions 19
 - Create a Role 19
 - Create Permission 19

- 4 Working with Tags 21**

- 5 Configuring Your Virtual Infrastructure 22**
 - Add a Cloud to VMware Telco Cloud Automation 22
 - Configure the Compute Profile 24
 - Edit a Virtual Infrastructure Account 26

- 6 Viewing Your Cloud Topology 27**

- 7 Deploying a Kubernetes Cluster 28**
 - Creating a Kubernetes Template 29
 - Create a Kubernetes Template 29
 - Deploy a Kubernetes Cluster 31

- 8 Managing Network Function Catalogs 34**
 - Onboarding a Network Function 34
 - Upload a Network Function Package 34
 - Designing a Network Function Descriptor 35
 - Edit Network Function Descriptor Drafts 41
 - Edit the Network Function Catalog Source Files 42
 - Delete a Network Function 42
 - Customizing Network Function Infrastructure Requirements 42
 - CNF with Customizations Example 43

9 Managing Network Function Lifecycle Operations 52

- Instantiating a Network Function 52
 - Instantiate a Virtual Network Function 52
 - Instantiate a Cloud Native Network Function 54
- Heal an Instantiated Network Function 55
- Scale an Instantiated VNF 55
- Scale an Instantiated CNF 57
- Operate an Instantiated Network Function 57
- Run a Workflow on an Instantiated Network Function 58
- Terminate a Network Function 58
- Download a Network Function 59

10 Managing Network Service Catalogs 60

- Onboarding a Network Service 60
 - Upload a Network Service Package 60
 - Design a Network Service Descriptor 61
 - Edit Network Service Descriptor Drafts 64
 - Delete a Network Service 64

11 Managing Network Service Lifecycle Operations 65

- Instantiate a Network Service 65
- Run a Workflow on a Network Service 67
- Heal a Network Service 67
- Terminate a Network Service 68
- Download a Network Service 69

12 Upgrading Network Functions and Network Services 70

- Upgrade a VNF Package 70
- Upgrade a CNF Package 71
- Update a CNF 71
- Upgrade a CNF 72
- Upgrade Network Service Package 73

13 Running Workflows with vRealize Orchestrator 74

- Key Concepts of Workflows 76
 - Workflow Parameters 77
 - Workflow Attributes and Variables 77
 - Workflow Bindings 77
- Creating a Workflow 78
 - Defining Workflow Variables and Parameters 79
- Workflow Examples 89

- [Ansible Workflow](#) 89
- [SSH Workflows](#) 90
- [File Workflow Example](#) 93
- [Custom vRO Workflows](#) 94
- [VMware Tools Script](#) 96
- [Multiple Steps](#) 97
- [Variables](#) 98
- [Conditions](#) 100

14 Monitoring Performance and Managing Faults 102

- [Managing Alarms](#) 102
- [Performance Management Reports](#) 103
 - [Schedule Performance Management Reports](#) 103
- [Monitor Instantiated Network Functions and Virtual Deployment Units](#) 104
- [Monitor Instantiated Network Services](#) 105

15 Administrating VMware Telco Cloud Automation 107

- [Performing System Updates](#) 107
- [Viewing Audit Logs](#) 107
- [Troubleshooting and Support](#) 107

16 Scheduling HCX Upgrades 108

- [Creating Upgrade Groups](#) 108
- [Scheduling Group Upgrades](#) 109
- [Monitoring Group Upgrades](#) 112
- [Deleting a Schedule](#) 114

17 Registering Partner Systems 116

- [Add a Partner System to VMware Telco Cloud Automation](#) 116
- [Edit a Registered Partner System](#) 117
- [Associate a Partner System Network Function Catalog](#) 117
- [Add a Harbor Repository](#) 118

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 life-cycle 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 HCX® for Telco Cloud - Provides the infrastructure for placing workloads across clouds using VMware Telco Cloud Automation.

This chapter includes the following topics:

- [Common Abbreviations](#)
- [Deployment Architecture](#)

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. SVNFM 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

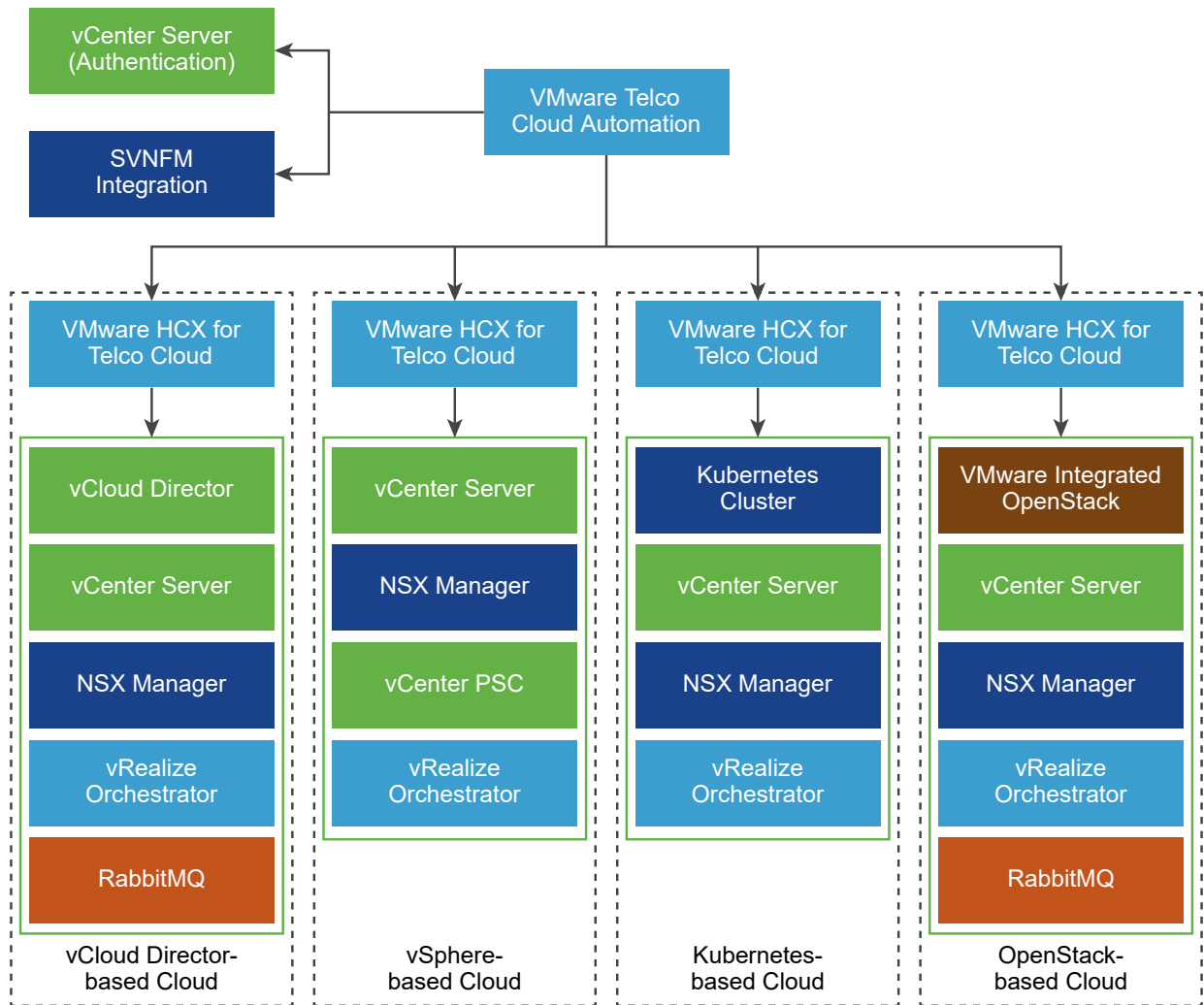
Container Storage Interface. A specification designed to enable persistent storage volume management on Container Orchestrators (COs) such as Kubernetes. The specification allows storage systems to integrate with containerized workloads running on Kubernetes. Using CSI, storage providers, such as VMware, 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.

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 HCX For Telco Cloud is deployed on the VIM and paired with VMware Telco Cloud Automation.

- VMware Telco Cloud Automation connects with VMware HCX For Telco Cloud 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 VMware HCX For Telco Cloud 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 VMware HCX for Telco Cloud, see *VMware HCX For Telco Cloud Deployment Guide*.
- RabbitMQ is used to track vCloud Director and VMware Integrated OpenStack notifications and is required only for deployments on these clouds.

Getting Started

2

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

1 Install and set up:

- VMware HCX For Telco Cloud
- VMware Telco Cloud Automation

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

2 Create Roles and assign Permissions. See [Chapter 3 Managing Roles and Permissions](#).

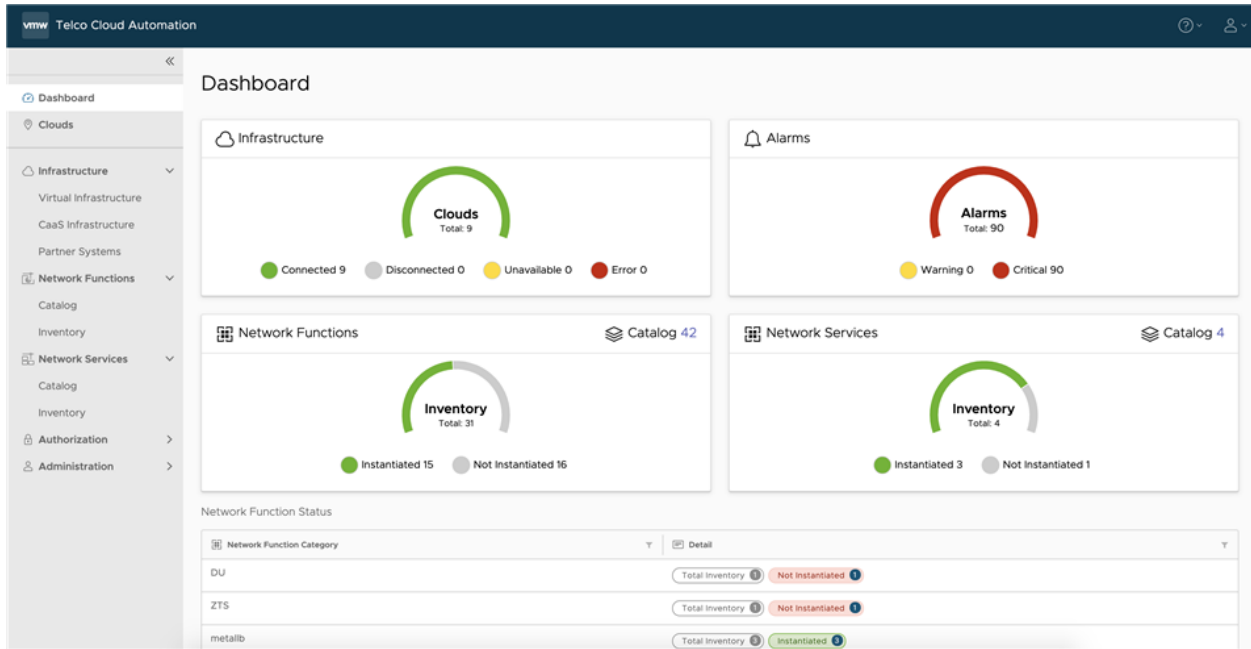
3 Configure your VIMs. See [Chapter 5 Configuring Your Virtual Infrastructure](#).

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.



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.

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](#)
- [Object Level Access Permissions](#)
- [Privileges and Roles](#)
- [Creating Roles and Permissions](#)

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 top-right corner, click the drop-down menu next to the **User** icon. Go to **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 Administrative privileges for all operations.	<ul style="list-style-type: none"> ■ Role Admin ■ System Audit ■ Virtual Infrastructure Audit ■ Virtual Infrastructure Admin ■ Virtual Infrastructure Consume ■ Network Function Catalog Design ■ Network Function Catalog Read ■ Network Function Catalog Instantiate ■ Network Function Instance Read ■ Network Function Instance Lifecycle Management ■ Network Service Catalog Design ■ Network Service Catalog Read ■ Network Service Catalog Instantiate ■ Network Service Instance Read ■ Network Service Instance Lifecycle Management ■ Partner System Read ■ Partner System Admin ■ Role Audit 	<ul style="list-style-type: none"> ■ Network Function Catalog ■ Network Service Catalog ■ Virtual Infrastructure ■ Network Function Instance ■ Network Service Instance
Virtual Infrastructure Admin Administrative privileges for VIM.	Virtual Infrastructure Audit	Virtual Infrastructure
Virtual Infrastructure Audit Read privileges for VIM.		Virtual Infrastructure
Partner System Read Read privileges for partner systems.		Virtual Infrastructure
Partner System Admin Administrative privileges for partner systems.	<ul style="list-style-type: none"> ■ Partner System Read ■ Network Function Catalog Read ■ Virtual Infrastructure Audit 	Virtual Infrastructure
Virtual Infrastructure Consume Deploy privileges for VIM.	Virtual Infrastructure Audit	Virtual Infrastructure
Network Function Catalog Design Design privileges for Network Function Catalog.	Network Function Catalog Read	Network Function Catalog
Network Function Catalog Read Read privileges for Network Function Catalog.		Network Function Catalog

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

Privilege	Included Privileges	Accessible Objects
Network Function Catalog Instantiate Instantiation privileges for Network Function Catalog	<ul style="list-style-type: none"> ■ Network Function Catalog Read ■ Virtual Infrastructure Consume ■ Network Function Instance Read 	Network Function Catalog
Network Function Instance Read Read privileges for Network Function Instance.		<ul style="list-style-type: none"> ■ Network Function Instance ■ Network Function Catalog
Network Function Instance Lifecycle Management Lifecycle management privileges for Network Function Instance.	<ul style="list-style-type: none"> ■ Network Function Instance Read ■ Network Function Catalog Instantiate ■ Network Function Catalog Read ■ Virtual Infrastructure Consume 	Network Function Instance
Network Service Catalog Design Design privileges for Network Service Catalog.	<ul style="list-style-type: none"> ■ Network Service Catalog Read ■ Network Function Catalog Read 	Network Service Catalog
Network Service Catalog Read Read privileges for Network Service Catalog.	Network Function Catalog Read	Network Service Catalog
Network Service Catalog Instantiate Instantiation privileges for Network Service Catalog.	<ul style="list-style-type: none"> ■ Network Service Catalog Read ■ Virtual Infrastructure Consume ■ Network Function Instance Read ■ Network Service Instance Read 	Network Service Catalog
Network Service Instance Read Read privileges for Network Service Instance.		<ul style="list-style-type: none"> ■ Network Service Instance ■ Network Service Catalog
Network Service Instance Lifecycle Management Lifecycle Management privileges for Network Service Instance.	<ul style="list-style-type: none"> ■ Network Service Instance Read ■ Network Service Catalog Instantiate ■ Network Function Catalog Read ■ Network Function Instance Read ■ Virtual Infrastructure Consume ■ Network Function Catalog Read ■ Network Function Catalog Instantiate 	Network Service Instance
System Audit Read privileges for all operations.	<ul style="list-style-type: none"> ■ Virtual Infrastructure Audit ■ Partner System Read ■ Network Service Instance Read ■ Network Service Catalog Read ■ Network Function Instance Read ■ Network Function Catalog Read ■ Role Audit 	<ul style="list-style-type: none"> ■ Network Function Instance ■ Network Service Instance ■ Virtual Infrastructure ■ Network Function Catalog ■ Network Service Catalog

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

Privilege	Included Privileges	Accessible Objects
Role Admin Administration privileges for all roles operations.	Role Audit	
Role Audit Read privileges for all Role operations.		

Table 3-2. System Defined Roles

Role	Privileges
System Administrator The users assigned to this role can perform all the available actions in VMware Telco Cloud Automation.	<ul style="list-style-type: none"> ■ Role Admin ■ System Audit ■ Virtual Infrastructure Audit ■ Virtual Infrastructure Admin ■ Virtual Infrastructure Consume ■ Network Function Catalog Design ■ Network Function Catalog Read ■ Network Function Catalog Instantiate ■ Network Function Instance Read ■ Network Function Instance Lifecycle Management ■ Network Service Catalog Design ■ Network Service Catalog Read ■ Network Service Catalog Instantiate ■ Network Service Instance Read ■ Network Service Instance Lifecycle Management ■ Partner System Read ■ Partner System Admin ■ Role Audit
Network Function Designer The users assigned to this role can perform all the network function actions such as designing, uploading, and managing the Network Function Catalogs.	<ul style="list-style-type: none"> ■ Network Function Catalog Read ■ Network Function Instance Read
Network Function Deployer 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.	<ul style="list-style-type: none"> ■ Network Function Instance Read ■ Network Function Catalog Instantiate ■ Network Function Catalog Read ■ Virtual Infrastructure Consume ■ ■ Virtual Infrastructure Audit ■ Network Function Instance Lifecycle Management
Virtual Infrastructure Administrator The users assigned to this role can perform all the virtual infrastructure-related actions in VMware Telco Cloud Automation.	<ul style="list-style-type: none"> ■ Virtual Infrastructure Audit ■ Virtual Infrastructure Admin ■ Virtual Infrastructure Consume

Table 3-2. System Defined Roles (continued)

Role	Privileges
<p>Virtual Infrastructure Auditor The users assigned to this role can view all the virtual infrastructure entities in VMware Telco Cloud Automation.</p>	<p>Virtual Infrastructure Audit</p>
<p>Network Service Designer The users assigned to this role can perform all the network service actions such as designing, uploading, and managing the Network Service Catalogs.</p>	<ul style="list-style-type: none"> ■ Network Service Catalog Design ■ Network Service Catalog Read ■ Network Function Catalog Read ■ Network Function Instance Read ■ Network Service Instance Read
<p>Network Service Deployer 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.</p>	<ul style="list-style-type: none"> ■ Network Service Instance Read ■ Network Service Catalog Instantiate ■ Network Service Catalog Read ■ Network Function Instance Read ■ Virtual Infrastructure Consume ■ Network Function Catalog Read ■ Network Function Catalog Instantiate ■ Virtual Infrastructure Audit ■ Network Service Instance Lifecycle Management ■ Network Function Instance Lifecycle Management
<p>System Auditor The users assigned to this role can view all the entities in VMware Telco Cloud Automation.</p>	<ul style="list-style-type: none"> ■ System Audit ■ Virtual Infrastructure Audit ■ Network Service Instance Read ■ Network Service Catalog Read ■ Network Function Catalog Read ■ Network Function Instance Read ■ Partner System Read ■ Role Audit
<p>Role Administrator The users assigned to this role can perform all the object access control related actions in VMware Telco Cloud Automation.</p>	<ul style="list-style-type: none"> ■ Role Admin ■ Role Audit
<p>Partner System Administrator The users assigned to this role can perform all the partner system-related actions in VMware Telco Cloud Automation.</p>	<ul style="list-style-type: none"> ■ Partner System Read ■ Partner System Admin ■ Network Function Catalog Read ■ Virtual Infrastructure Audit
<p>Partner System Read Only The users assigned to this role can view all the partner system entities in VMware Telco Cloud Automation.</p>	<p>Partner System Read</p>
<p>Role Auditor The users assigned to this role can view all the object access control related roles and permissions in VMware Telco Cloud Automation.</p>	<p>Role Audit</p>

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 HCX 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](#)
- [Configure the Compute Profile](#)
- [Edit a Virtual Infrastructure Account](#)

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 For self-signed certificates, import the HCX certificate from VMware Telco Cloud Automation Appliance Management. For more information, see the *VMware HCX for Telco Cloud Deployment Guide*.

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

Cloud Name	Enter a name for your virtual infrastructure.
Cloud URL	Enter the VMware HCX 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. <ul style="list-style-type: none"> ■ The format for a vCloud Director-based cloud is <code>username@organization-name</code>. ■ The role for vCloud Director is Organization Administrator. ■ 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 VMware HCX 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 HCX 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 VMware HCX 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 <code>username@domain-name</code> .
Password	Enter the infrastructure user password.

- 4 Click **Add** and click **Validate**.

The configuration is validated.

- 5 Optionally, you can add tags to your cloud. Tags are used for filtering and grouping clouds, network functions, and network services.
- 6 Click **OK** after the validation is complete.

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 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 click the desired cloud.
The cloud details page is displayed.
- 3 Under **Compute Profiles**, click **Add**.

4 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.

5 Click **Add**.

Results

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

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

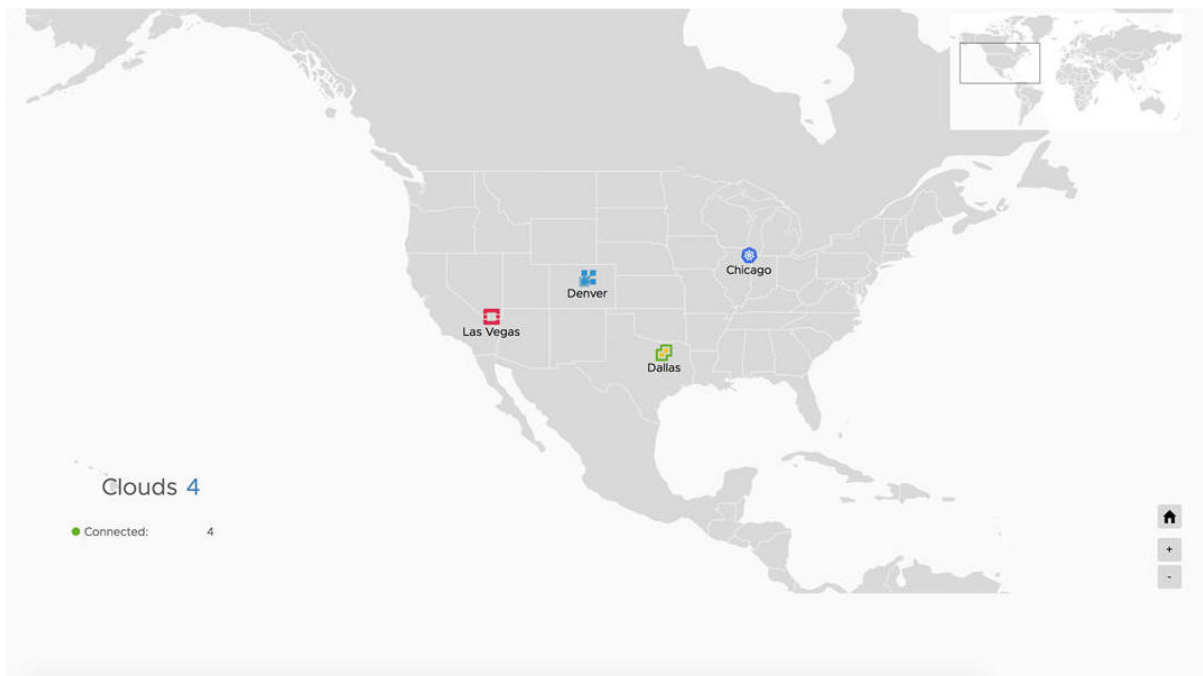
6

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.

Deploying a Kubernetes Cluster

7

You deploy your CNF on a Kubernetes cluster. A Kubernetes cluster is a set of nodes that run containerized applications.

Note The CaaS Infrastructure features are on technology preview for this release.

Containerized applications are more lightweight and flexible than virtual machines. 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. Kubernetes containers are not restricted to a specific operating system, unlike virtual machines. Instead, they are able to share operating systems and run anywhere.

Kubernetes clusters are composed of several master nodes and worker nodes. The master nodes control the state of the cluster. For example, a master node controls the applications that are running and their corresponding container images. The master node is the origin for all task assignments.

The worker nodes are the components that run these applications. Worker nodes perform tasks assigned by the master node. They are virtual machines operating as part of one system.

There must be a minimum of one master 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.

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 cluster and both these clusters can be deployed on different vCenter Servers.

For more information about the VMware Tanzu Kubernetes Grid concepts, see the VMware Tanzu Kubernetes Grid product documentation at <https://docs.vmware.com/en/VMware-Tanzu-Kubernetes-Grid/1.1/vmware-tanzu-kubernetes-grid-11/GUID-tkg-concepts.html>.

Kubernetes Cluster Deployment Process

The process to deploy a Kubernetes cluster is:

- 1 Onboard a vCenter Server as a cloud.

- 2 Define a Kubernetes template for the Management cluster.
- 3 Define a Kubernetes template for the Workload cluster, for the required CNF.
- 4 Deploy the Management and Workload cluster. After the Workload cluster is deployed, it is automatically registered in the VMware Telco Cloud Automation as a Kubernetes Cloud VIM.

This chapter includes the following topics:

- [Creating a Kubernetes Template](#)
- [Deploy a Kubernetes Cluster](#)

Creating a Kubernetes Template

Before creating a Kubernetes cluster, you must create a Kubernetes template to deploy the cluster. Using VMware Telco Cloud Automation, you can create a Kubernetes template and use it to deploy multiple clusters.

When you create a Kubernetes template, you must provide two types of configuration information:

- **Cluster Configuration** - Here, you specify the details about the Container Storage Interfaces (CSI) such as vSphere-CSI and NFS Client, Container Network Interface (CNI) such as *Calico* and *Multus*, 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 and worker nodes, and so on.

Create a Kubernetes Template

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

Prerequisites

You require **System Administrator** permissions to perform this operation.

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 template.
 - **Cluster Type** - Select whether your cluster is a **Management Cluster** or a **Workload Cluster**.
 - **Version** - Enter the template version number.
 - **Description** (Optional) - Enter a description for the template.

- **Tags** (Optional) - Add appropriate tags to the template.
 - **Kubernetes Version** - If you have selected **Management Cluster** as the **Cluster Type**, select the Kubernetes version from the drop-down menu. The supported Kubernetes version for the Management Cluster is **1.18.2**. The supported Kubernetes version for the Workload Cluster is **1.17.3**.
- 4 Click **Next**.
- 5 If you have selected **Workload Cluster** as the **Cluster Type**, provide the following details in the **Cluster Configuration** tab:
- **Kubernetes Version** - Select the Kubernetes version from the drop-down menu. The supported versions are **1.17.3** and **1.18.2**.
 - **CNI** - Click **Add** and select a Container Network Interface (CNI). The supported CNIs are **Multus** and **Calico**. Enter details such as version and properties. If you do not provide a version number, the default version number is populated. To add additional properties, click **Add** under **Properties**. You can also add additional CNIs.
 - **CSI** - Click **Add** and select a Container Storage Interface (CSI) such as vSphere CSI or NFS Client. To add a timeout value, click **Add** under **Additional Properties** and select **Timeout** from the drop-down menu. Assign a timeout value.
 - **Tools** - Click **Add** and select a tool such as **Helm** from the drop-down menu. Enter the Helm version.
- 6 Click **Next**.
- 7 In the **Master Node Configuration** tab, add a node profile. A node profile is a set of nodes that have similar VMs. Profiling 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 profiles with different groups of VMs. To add a node profile, click **Add** and enter the following details:
- **Name** - Name of the profile
 - **CPU** - Number of vCPUs
 - **Memory** - Memory in MB
 - **Storage** - Storage size in GB
 - **Taint** - If **Taint** is enabled, you cannot instantiate CNFs on the master nodes for this profile.
 - **Networks** - Enter the labels to group the networks. The labels are useful when you want to deploy a Kubernetes profile with a specific set of network preferences.
 - **Labels** - Enter the appropriate labels for this profile.
 - **Nodes** - Enter the following information for each node:
 - **Name** - Name of the node.

- **Replica** - Number of Master Node VMs to be created.
 - **Add Label** - Click to define the label key and value for the node.
 - To add another node, click the **+** symbol.
- 8** In the **Worker Node Configuration** tab, add a node profile. A node profile is a set of nodes that have similar VMs. Profiling 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 profiles with different groups of VMs. To add a node profile, click **Add** and enter the following details:
- **Name** - Name of the profile
 - **CPU** - Number of vCPUs
 - **Memory** - Memory in MB
 - **Storage** - Storage size in GB
 - **Networks** - Enter the labels to group the networks. The labels are useful when you want to deploy a Kubernetes profile with a specific set of network preferences.
 - **Labels** - Enter the appropriate labels for this profile.
 - **Nodes** - Enter the following information for each node:
 - **Name** - Name of the node.
 - **Replica** - Number of Worker Node VMs to be created.
 - **Add Label** - Click to define the label key and value for the node.
 - To add another node, click the **+** symbol.
- 9** Click **Add Template**.

Results

The template is created.

What to do next

Deploy a Kubernetes cluster.

Deploy a Kubernetes Cluster

Deploy a Kubernetes cluster on the VMs in the Kubernetes template.

Prerequisites

- You must have the **Virtual Infrastructure Admin** privileges to perform this task.
- You must have uploaded the Virtual Machine template and Load Balancer template to VMware Telco Cloud Automation.

Procedure

- 1 Log in to the VMware Telco Cloud Automation web interface.
- 2 Go to **CaaS Infrastructure** and click **Deploy Kubernetes Cluster**.
- 3 Select a cloud on which you want to deploy the Kubernetes cluster.
- 4 Click **Next**.
- 5 The **Select Cluster Templates** tab displays the available Kubernetes clusters. Select the Kubernetes template that you have created.
- 6 Click **Next**.
- 7 In the **Kubernetes Cluster Details** tab, provide the following details:
 - **Name** - Enter the cluster name.
 - **Description** (Optional) - Enter an optional description of the cluster.
 - **Virtual Machine Template** - Select the virtual machine template that you have uploaded.
 - **Load Balancer Template** - Select the Load Balancer template that you have uploaded.

Note Only those templates that VMware Tanzu Kubernetes Grid supports are displayed for selection.

 - **Network** - Select between a Management/DVPG or Private network.
 - **VM Folder** - Select a folder to use for your cluster.
 - **Resource Pool** - Select a resource pool for your cluster.
 - **Datastore** - Select a datastore for your cluster.
 - **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.
- 8 Click **Next**.
- 9 In the **Master Node Configuration** tab, provide the following details:
 - **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 network.
- 10 Click **Next**.
- 11 In the **Worker Node Configuration** tab, provide the following details:
 - **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.

12 Click **Deploy**.

Results

The Kubernetes 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.

Managing Network Function Catalogs



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](#)
- [Customizing Network Function Infrastructure Requirements](#)

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 13 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 13 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**.

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.

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.

CNF with Customizations Example

Here are some CNF customization examples.

Example 1

```
tosca_definitions_version: toska_simple_profile_for_nfv_1_0_0
description: Sample DU CNF with Node Customization
imports:
  - vmware_etsi_nfv_sol001_vnfd_2_5_1_types.yaml
node_types:
  toska.nodes.nfv.VMware.CNF.Vendor-DU-1.1:
    derived_from: toska.nodes.nfv.VMware.CNF
    interfaces:
      Vnflcm:
        type: toska.interfaces.nfv.Vnflcm
topology_template:
  substitution_mappings:
    node_type: toska.nodes.nfv.VMware.CNF.Vendor-DU-1.1
  properties:
    descriptor_id: vnfd_ae24679d-3228-4f21-95c8-dedd66858cc1
    provider: Vendor
    vendor: Vendor
    product_name: DU
    version: '1.1'
    id: id
    software_version: '1.0'
    descriptor_version: '1.1'
```

```

flavour_id: default
flavour_description: default
vnfm_info:
  - gvnfmdriver
infra_requirements:
  node_components:
    isNumaConfigNeeded: true
  kernel:
    kernel_type:
      name: linux-rt
      version: 4.19.98-4.ph3
    kernel_args:
      - key: nosoftlockup
      - key: intel_idle.max_cstate
        value: 1
      - key: mce
        value: ignore_ce
      - 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: 2-{{tca.node.vmNumCPUs}}
      - key: skew_tick
        value: 1
      - key: nohz
        value: on
      - key: nohz_full
        value: 2-{{tca.node.vmNumCPUs}}
      - key: rcu_nocb_poll
        value: 1
      - key: rcu_nocbs
        value: 2-{{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: 0,1
      - key: clocksource

```

```

        value: tsc
      - key: tsc
        value: reliable
    kernel_modules:
      - name: dpdk-rt
        version: 19.08.2
    custom_packages:
      - name: pciutils
        version: 3.6.2-1.ph3
      - name: tuned
        version: 2.13.0-1.ph3
    additional_config:
      - name: tuned
        value: '[{"name": "vendor-du"}]'
    caas_components:
      - name: host-device
        type: cni
      - name: static
        type: cni
      - name: sriov
        type: cni
      - name: vendor-tuned
        type: cni
      - name: vendor-tuned-rt
        type: cni
  node_templates:
    Vendor-DU-1.1:
      node_type: tosca.nodes.nfv.VMware.CNF.Vendor-DU-1.1
      properties:
        descriptor_id: vnfd_ae24679d-3228-4f21-95c8-dedd66858cc1
        provider: Vendor
        vendor: Vendor
        product_name: DU
        version: '1.1'
        id: id
        software_version: '1.0'
        descriptor_version: '1.1'
        flavour_id: default
        flavour_description: default
      vnf_info:
        - gvnfmdriver
      infra_requirements:
      node_components:
        isNumaConfigNeeded: true
      kernel:
        kernel_type:
          name: linux-rt
          version: 4.19.98-4.ph3
        kernel_args:
          - key: nosoftlockup
          - key: intel_idle.max_cstate
            value: 1
          - key: mce
            value: ignore_ce
          - 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: 2-{{tca.node.vmNumCPUs}}
- key: skew_tick
  value: 1
- key: nohz
  value: on
- key: nohz_full
  value: 2-{{tca.node.vmNumCPUs}}
- key: rcu_nocb_poll
  value: 1
- key: rcu_nocbs
  value: 2-{{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: 0,1
- key: clocksource
  value: tsc
- key: tsc
  value: reliable
kernel_modules:
- name: dpdk-rt
  version: 19.08.2
custom_packages:
- name: pciutils
  version: 3.6.2-1.ph3
- name: tuned
  version: 2.13.0-1.ph3
additional_config:
- name: tuned
  value: '[{"name": "vendor-du"}]'
caas_components:
- name: host-device
  type: cni
- name: static

```

```

    type: cni
  - name: sriov
    type: cni
  - name: vendor-tuned
    type: cni
  - name: vendor-tuned-rt
    type: cni

```

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.vendor_UP_v1.1:
    derived_from: tosca.nodes.nfv.VMware.CNF
    interfaces:
      Vnflcm:
        type: tosca.interfaces.nfv.Vnflcm
  tosca.nodes.nfv.Vdu.Compute.Helm.vendor-cu-up:
    derived_from: tosca.nodes.nfv.Vdu.Compute.Helm
    properties:
      configurable_properties:
        type: tosca.datatypes.nfv.VnfcConfigurableProperties.vendor-cu-up
        required: true
data_types:
  tosca.datatypes.nfv.VnfcConfigurableProperties.vendor-cu-up:
    derived_from: tosca.datatypes.nfv.VnfcConfigurableProperties
    properties:
      additional_vnfc_configurable_properties:
        type: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.vendor-cu-up
        description: Describes additional configuration for VNFC that can be configured
        required: true
  tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.vendor-cu-up:
    derived_from: tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties
    properties:
      values:
        required: true
        propertyName: values
        description: 'Overrides for chart values '
        default: ''
        type: string
        format: file
      FIU:
        required: true
        propertyName: FIU
        description: ''
        default: ''
        type: string
        format: string
      NGU:
        required: true
        propertyName: NGU

```

```

description: ''
default: ''
type: string
format: string
E1C:
  required: true
  propertyName: E1C
  description: ''
  default: ''
  type: string
  format: string
topology_template:
  substitution_mappings:
    node_type: toasca.nodes.nfv.VMware.CNF.Vendor_UP_v1.1
  properties:
    descriptor_id: vnf_5389c52e-6b83-42c7-bb97-40537e8f620e
    provider: Vendor
    vendor: Vendor
    product_name: CU-UP
    version: '1.0'
    id: id
    software_version: 0.1.0
    descriptor_version: '1.1'
    flavour_id: default
    flavour_description: default
    vnf_info:
      - gvnfmdriver
    infra_requirements:
      node_components:
        kernel:
          kernel_type:
            name: linux
            version: 4.19.97-2.ph3
          kernel_args:
            - key: default_hugepagesz
              value: 1G
            - key: hugepagesz
              value: 1G
            - key: hugepages
              value: 8
            - 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: 4-19
  - key: nosoftlockup
  - key: nohz
    value: on
  - key: nohz_full
    value: 4-19
  - key: rcu_nocbs
    value: 4-19
kernel_modules:
  - name: dpdk
    version: 19.11.1
custom_packages:
  - name: pciutils
    version: 3.6.2-1.ph3
network:
  devices:
  - deviceType: sriov
    networkName: F1U
    resourceName: ani_netdevice_f1u
    dpdkBinding: igb_uio
  - deviceType: sriov
    networkName: NGU
    resourceName: ani_netdevice_ngu
    dpdkBinding: igb_uio
  - deviceType: sriov
    networkName: E1C
    resourceName: ani_netdevice_e1c_2
caas_components:
  - name: multus
    type: cni
    version: 1.0.0.1
  - name: host-device
    type: cni
  - name: static
    type: cni
  - name: sriov
    type: cni
interfaces:
  Vnflcm: {}
node_templates:
  Vendor_UP_v1.1:
    node_type: tosca.nodes.nfv.VMware.CNF.Vendor_UP_v1.1
    properties:
      descriptor_id: vnfd_5389c52e-6b83-42c7-bb97-40537e8f620e
      provider: Vendor
      vendor: Vendor
      product_name: CU-UP

```

```
version: '1.0'  
id: id  
software_version: 0.1.0  
descriptor_version: '1.1'  
flavour_id: default  
flavour_description: default  
vnfm_info:  
  - gvnfmdriver  
infra_requirements:  
  node_components:  
    kernel:  
      kernel_type:  
        name: linux  
        version: 4.19.97-2.ph3  
      kernel_args:  
        - key: default_hugepagesz  
          value: 1G  
        - key: hugepagesz  
          value: 1G  
        - key: hugepages  
          value: 8  
        - 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: 4-19  
        - key: nosoftlockup  
        - key: nohz  
          value: on  
        - key: nohz_full  
          value: 4-19  
        - key: rcu_nocbs  
          value: 4-19  
      kernel_modules:  
        - name: dpdk  
          version: 19.11.1
```

```

    custom_packages:
      - name: pciutils
        version: 3.6.2-1.ph3
  network:
    devices:
      - deviceType: sriov
        networkName: FIU
        resourceName: ani_netdevice_f1u
        dpdkBinding: igb_uio
      - deviceType: sriov
        networkName: NGU
        resourceName: ani_netdevice_ngu
        dpdkBinding: igb_uio
      - deviceType: sriov
        networkName: E1C
        resourceName: ani_netdevice_e1c_2
  caas_components:
    - name: multus
      type: cni
      version: 1.0.0.1
    - name: host-device
      type: cni
    - name: static
      type: cni
    - name: sriov
      type: cni
  interfaces:
    Vnflcm: {}
  vendor-cu-up:
    type: tosca.nodes.nfv.Vdu.Compute.Helm.vendor-cu-up
  properties:
    name: vendor-cu-up
    description: vendor cu-up
    chartName: vendor-vcu-up-chart
    chartVersion: 0.1.0
    helmVersion: v3
    id: helm1
  configurable_properties:
    additional_vnfc_configurable_properties:
      type: >-
      tosca.datatypes.nfv.VnfcAdditionalConfigurableProperties.vendor-cu-up
    values: ''
    NGU: ''
    FIU: ''
    E1C: ''

```

Managing Network Function Lifecycle Operations

9

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](#)
- [Heal an Instantiated Network Function](#)
- [Scale an Instantiated VNF](#)
- [Scale an Instantiated CNF](#)
- [Operate an Instantiated Network Function](#)
- [Run a Workflow on an Instantiated Network Function](#)
- [Terminate a Network Function](#)
- [Download a Network Function](#)

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.

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.

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.

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.
- 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 at the end.
 - 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 13 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 HCX for Telco Cloud. For more information, see the *VMware HCX for Telco Cloud 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.

Download a Network Function

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 Service Catalogs

10

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

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 13 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 Add network functions (VNFs), cloud-native network functions (CNFs), and networks (NS Virtual Link) by dragging the icons from the Toolbar into the design area. You can also add 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.

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.

Managing Network Service Lifecycle Operations

11

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

This chapter includes the following topics:

- [Instantiate a Network Service](#)
- [Run a Workflow on a Network Service](#)
- [Heal a Network Service](#)
- [Terminate a Network Service](#)
- [Download a Network Service](#)

Instantiate a Network Service

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

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** > **Catalog**.
- 3 Select the desired network service and click **Instantiate**.
- 4 Enter the following details: a name for your **Network Service** instance.
 - 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**.
 - 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 and 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 HCX for Telco Cloud. For more information, see the *VMware HCX for Telco Cloud 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.

Download a Network Service

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.

Upgrading Network Functions and Network Services

12

You can now map your upgraded VNFs, CNFs, and Network Services to the latest version in the Catalog.

VMware Telco Cloud Automation allows you to make minor software updates and major package and component upgrades to your network functions and network services.

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. When you upgrade the components of a CNF, you can map them to the new version. For example, after you rebrand the VDUs in a CNF, you can map the components of the CNF to its newer version.

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.

This chapter includes the following topics:

- [Upgrade a VNF Package](#)
- [Upgrade a CNF Package](#)
- [Update a CNF](#)
- [Upgrade a CNF](#)
- [Upgrade Network Service Package](#)

Upgrade a VNF Package

Map your VNF package to the latest version using VMware Telco Cloud Automation.

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

You can upgrade a CNF package using VMware Telco Cloud Automation.

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

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

When you upgrade your CNF, you can map its software version, Descriptor version, upgraded components, repository details, any instantiation properties, and Network Function properties of your CNF to the newer version.

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

Map your Network Service package to the latest version using VMware Telco Cloud Automation.

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.

Running Workflows with vRealize Orchestrator

13

Use vRealize Orchestrator to run operations that are not supported natively on VMware Telco Cloud Manager.

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. 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. <ul style="list-style-type: none"> ■ name - Name of the workflow. ■ type - String ■ exportName - User-provided variable. ■ 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](#)
- [Workflow Examples](#)

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
  ]
}
}
}

```

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":"step1",
  "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":"SSH_OUTPUT", "description": "SSH Command Output", "type": "string"},
    {"name":"ANSIBLE_OUTPUT", "description": "Ansible Command Output", "type": "string"}
  ],
  "steps":[
    {
      "stepId":"step1",
      "workflow":"VRO_CUSTOM_WORKFLOW",
      "namespace": "nfv",
      "type":"task",
      "description": "Unzip and 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": "step2"
  },
  {
    "stepId": "step2",
    "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"}
      ],
      "outBinding": [
        {"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"}
      ],
      "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"
}
]
}

```

Monitoring Performance and Managing Faults

14

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](#)
- [Performance Management Reports](#)
- [Monitor Instantiated Network Functions and Virtual Deployment Units](#)
- [Monitor Instantiated Network Services](#)

Managing Alarms

The **Dashboard** tab displays the total number of alarms triggered. It also displays the number of alarms according to their severity.

To view the alarms of individual CNFs and VNF instances, go to the **Inventory** tab, click a VNF or CNF instance, and click **Alarms**.

VNF alarms are triggered when VMware Telco Cloud Automation identifies anomalies in the network connection status or when the power state changes.

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.

You can view the global alarms for all entities and users from the **Administration > Alarms** tab. VNF and CNF alarms are listed at the corresponding Network Service level. For a VNF, the alarms are also listed at the corresponding VDU level.

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, click **Acknowledge**. When the acknowledgment is successful, the state of the alarm changes to **Acknowledged**.

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 instance.

You can generate reports for performance metrics such as **CPU Usage** and **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.

The following performance metrics can be collected:

- 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.

Schedule Performance Management Reports

Create and schedule a Performance Management Job Report.

Procedure

- 1 Log in to the VMware Telco Cloud Automation web interface.
- 2 Select **Network Functions > Inventory**.
- 3 Click the desired VNF, and from the VNF 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 Management 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

Mean Memory Usage

Number of Incoming Bytes

Disk Write

Number of Outgoing Bytes

Number of Incoming Packets

Number of Outgoing Packets

CANCEL

SCHEDULE REPORTS

6 Click **Schedule Reports**.

The report is scheduled and is available under **PM Reports** in the VNF 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 Network Functions and Virtual Deployment Units

After you instantiate a network function, 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:
 - To view more details of a Virtual Deployment Unit (VDU), click the **i** icon. To view more information about a virtual link, point to it.
 - To heal, scale, **terminate**, or run a workflow on the selected network function, click the **Actions** menu.
 - To perform a heal operation on a specific VDU, click the **More (...)** icon on the desired VDU in the network function and click **Heal**.
 - To view detailed information about the VDU and the VNFs, their performance data, alarms, and reports, click the **More (...)** 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**.
 - **Reports**: 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.
 - 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 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 heal, scale, *terminate*, run a workflow, retry, or roll back, the network service, click the **Actions** menu.
- To view and acknowledge the consolidated alarms of all the VNFs 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.

Administering VMware Telco Cloud Automation

15

Perform system updates, view logs, and download logs for auditing and troubleshooting.

This chapter includes the following topics:

- [Performing System Updates](#)
- [Viewing Audit Logs](#)
- [Troubleshooting and Support](#)

Performing System Updates

When a new version of VMware Telco Cloud Automation becomes available, you can update your deployment from the web interface.

To check for a newer version of VMware Telco Cloud Automation, go to **Administration** > **System Updates**.

If a newer version is available, select the version and follow the prompts.

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.

Scheduling HCX Upgrades

16

Telco Cloud Automation enables you to schedule upgrades for HCX systems installed at Telco Cloud sites through a centralized management interface.

Using the HCX Upgrade Manager interface provided by Telco Cloud Automation, you organize individual HCX systems into logical groups. You then apply an upgrade schedule to each group. The group schedule is synchronized down to the HCX systems associated with that group. When the schedule occurs, the individual HCX systems in that group automatically begin the upgrade process. Each system provides its upgrade status back to the HCX Upgrade Manager.

Note Because the HCX 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](#)
- [Scheduling Group Upgrades](#)
- [Monitoring Group Upgrades](#)
- [Deleting a Schedule](#)

Creating Upgrade Groups

Managing upgrades is made simpler by creating logical groups for HCX 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 > HCX Upgrade Manager** and click **HCX Systems**.

The list of HCX systems that are paired with VMware Telco Cloud Automation appears in the window.

The screenshot shows the 'HCX Upgrade Manager' interface. At the top, there are tabs for 'HCX Systems' (selected) and 'HCX Groups'. Below the tabs are buttons for 'CREATE GROUP', 'ADD TO GROUP', and 'REFRESH'. The main area contains a table with the following columns: System Name, Cloud Type, System ID, Version, Release Code, and Group Name. There are two rows of data in the table.

	System Name	Cloud Type	System ID	Version	Release Code	Group Name
<input type="checkbox"/>	sc2-hs3-hcx-nmb-vm372.eng.vmware.com	VCD	20200408192958908-f20c6020-3093-410b-98be-715bfa71d4a3	3.5.3.15994870	139	tt-11
<input type="checkbox"/>	sc2-hs3-hcx-nmb-vm577.eng.vmware.com	VC	20200408192303211-4f346e43-6b90-4ccf-b92c-3a3ee0fd0850	3.5.3.15994870	139	

At the bottom right of the table, there is a 'Systems per page' dropdown set to '10' and a status indicator '1 - 2 of 2 Systems'.

- 3 Select the HCX systems that you want to group for upgrading.

An HCX system can be part of only one group. Group creation fails if you select HCX systems that are already associated with a group.

Note You can add additional HCX systems to an existing group at any time. To add more systems to a group, navigate to **HCX 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 HCX connected systems.

Results

The systems associated with TCA are now logically grouped for upgrade operations.

You can view the group name associated with a specific HCX system from **HCX Upgrade Manager > HCX Systems**. Or, you can review the list of systems available in each group at **HCX Upgrade Manager > HCX 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 **HCX 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 HCX 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 > HCX Upgrade Manager** and click **HCX Groups**.

The system displays the list of groups.

- 3 Click a group name.

The group information window lists the HCX 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. 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 HCX service interface (https://hcxip-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	<p>You have two options when selecting a monthly schedule:</p> <ul style="list-style-type: none"> ■ Update on a specific date in a month ■ Update on a specific day in a month <p>Enter the date or day, the monthly interval, and the time of day that you want the upgrade to occur.</p> <p>Note Create schedule intervals that adhere system support policies.</p>

The system displays your selections as a CRON expression.

7 Click **Add**.

The HCX Upgrade Manager in Telco Cloud Automation applies the schedule information to each HCX 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 HCX systems, or for the overall group.

The screenshot displays the 'Edit Group' configuration page for a group named 'UpgradeGroup'. It shows the group's name, description, and an 'EDIT' button. Below this are two summary charts: 'Recent Upgrade Status' and 'Schedule Status'. The 'Recent Upgrade Status' chart shows 2 systems in a 'COMPLETE' state. The 'Schedule Status' chart shows 1 system 'IN_PROGRESS' and 1 system 'SCHEDULED'. A table below lists the systems with their names, IDs, versions, and current upgrade and schedule statuses. At the bottom, there is a 'Schedules' section showing a 'WeeklySchedule' that is enabled and set to the 'America/Los_Angeles' time zone.

System Name	System ID	Version	Upgrade Status	Schedule Status
wdc-pod3-vcfs-vc02-4-142.eng.vmware.com	20200229021417825-1c96c568-38ae-4536-d915-7d574637699a	3.5.3.34819587	Complete	Scheduled
wdc-pod3-vcfs-vc01-2-84L.eng.vmware.com	20200229022730832-3f96a0c2-6eef-43e0-ac20-49c56da2154	3.5.3.34819587	Complete	In Progress

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 HCX system upgrades and activities.

Monitoring Group Upgrades

You can monitor the upgrade status for HCX systems through the HCX Upgrade Manager interface in VMware Telco Cloud Automation.

Procedure

- 1 Log in to the VMware Telco Cloud Automation web interface.
- 2 Navigate to **Administration > HCX Upgrade Manager** and click **HCX Groups**.

The system displays the list of groups.

- 3 Click a group name.

The group information window lists the HCX 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 HCX system has no upgrade history.

Option	Description
Status Chart	Description
Upgrade Status	<p>Provides the status of the most recent scheduled upgrade.</p> <ul style="list-style-type: none"> ■ Downloading - Indicates the number of systems that have started the download. ■ Downloaded- Indicates the number of systems that have completed the download. ■ Running - Specifies the number of systems installing the upgrade. ■ Complete - Indicates the number of systems that fully completed the upgrade. ■ Failed - Specifies how many systems were not upgraded during the scheduled time.
Schedule Status	<p>Provides the schedule synchronization status. When you add a group schedule, it is updated on each of the systems belonging to the group.</p> <ul style="list-style-type: none"> ■ In-Progress - Identifies the number of systems receiving the schedule information. ■ Scheduled - Specifies the number of systems that have been updated with the latest schedule. ■ 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 HCX 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 HCX system entry.
A pop-up window appears with selections for **Upgrade History** and **Activity History**.
 - c Choose from the selections to review HCX system information:

System History	Description
Upgrade	<p>Provides the history of software version updates on the HCX system. The Status column lists the current state of the upgrade: Downloading, Downloaded, Complete, Running, Failed.</p> <p>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.</p>
Activity	<p>Provides status information about schedule updates applied on the system from HCX Upgrade Manager.</p> <ul style="list-style-type: none"> ■ Type - Identifies the schedule change applied on the HCX system. ■ Status - Indicates the state of the change applied on the system: Initiated, Running, Failed, Complete. <p>Note A failure can indicate that the system was down or unresponsive. Check the connectivity of the systems using Link Last Communicated.</p> <ul style="list-style-type: none"> ■ Message - Provides additional information about the change status. ■ Schedule Name - Identifies the name of the schedule in the group that originated the change. ■ 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 HCX Upgrade Manager, but also from each member HCX system of the group.

When you delete a schedule, HCX 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 HCX 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 > HCX Upgrade Manager** and click **HCX Groups**.

The system displays the list of groups.

- 3 Click a group name.

The group information window lists the HCX 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 HCX 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.

Registering Partner Systems

17

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](#)
- [Edit a Registered Partner System](#)
- [Associate a Partner System Network Function Catalog](#)
- [Add a Harbor Repository](#)

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.

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.

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 HCX for Telco Cloud.

Note If your Harbor repository does not have self-signed certificates, you must import the certificates to VMware HCX for Telco Cloud.

- **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.