VMware vSphere Automation REST API Programming Guide

Update 2

VMware vSphere 8.0

VMware ESXi 8.0

vCenter Server 8.0

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

https://docs.vmware.com/

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

Copyright © 2022 - 2023 VMware, Inc. All rights reserved. Copyright and trademark information.

Contents

About the vSphere Automation REST API Programming Guide

VMware vSphere Automation REST API Programming Guide provides information about how to use the VMware vSphere® Automation™ REST services to automate different vSphere management tasks.

At VMware, we value inclusion. To foster this principle within our customer, partner, and internal community, we have updated this guide to remove instances of non-inclusive language.

Intended Audience

This manual is intended for anyone who wants to develop applications for accessing and using vSphere features such as virtual machine management, tagging, content libraries, managing the life cycle of clusters with the vSphere Lifecycle Manager, managing internal and external certificates, vSphere with Tanzu, and so on. The information is written for developers who have understanding of the targeted vSphere features and some experience with the Representational State Transfer (REST) architectural style.

Introduction to the vSphere Automation REST API

1

Use the vSphere Automation REST API to manage your virtual infrastructure by using HTTP.

With the vSphere Automation REST API, you can manage vSphere components connected to authentication, tagging, content libraries, clusters, internal and external certificates, namespaces, vSphere Trust Authority, vSphere with Tanzu and more.

This programming guide introduces the most important features and use cases of the vSphere Automation REST API. You can use this guide in combination with the *vSphere Automation API Reference* that features all available API services, as well as many code samples.

Read the following topics next:

Components of the vSphere Automation Virtualization Layer

At the core of vSphere Automation is vSphere, which provides the virtualization layer of the software-defined data center. You can use vSphere deployment options for vCenter Server and ESXi hosts to build virtual environments of different scales.

Components and Services of the vSphere Environment

Starting with vSphere 7.0, the installation and setup of vSphere is simplified to the deployment and upgrade of vCenter Server.

vCenter Server is a preconfigured virtual machine optimized for running the vCenter Server service and the vCenter Server components. The vCenter Server service acts as a central administrator for ESXi hosts.

Components Installed with vCenter Server

vCenter Server is a central administration point for ESXi hosts. The group of components installed when you install vCenter Server include the vCenter Server service, vSphere Client, VMware vSphere® Auto Deploy™, VMware vSphere® ESXi™ Dump Collector, VMware vSphere® Syslog Collector, and vSphere Lifecycle Manager service.

You can use the vSphere Automation API endpoint to access the following services running on vCenter Server.

Content Library

You can use content libraries to share virtual machines, vApps, and other files, such as ISO, OVA, and text files, across the software-defined data center. You can create, share, and subscribe to content libraries on the same vCenter Server instance or on a remote instance. Sharing content libraries promotes consistency, compliance, efficiency, and automation in deploying workloads at scale.

You can also create OVF and VM templates from virtual machines and vApps in hosts, resource pools, and clusters. You can then use the OVF and VM templates to deploy new virtual machines and vApps.

Starting with vSphere 7.0, you can edit the contents of a VM template. You can check out the library item that contains the VM template. After editing the VM template, check in the library item to save the changes to the virtual machine.

Virtual Machine

You can use the vSphere Automation APIs to create, configure, and manage the life cycle of virtual machines in your environment.

Starting with vSphere 7.0, you can also clone, create an instant clone, migrate, register, and unregister a virtual machine.

vSphere Lifecycle Manager

Starting with vSphere 7.0, the life cycle of ESXi hosts and clusters can be managed through the VMware vSphere® Lifecycle Manager™ feature. Based on the current state of the hosts in a cluster, you can easily create a desired software specification by using the contents of a software depot. Then you validate the desired software specification and you apply the specification on all hosts in the cluster.

vSphere Tags

With vSphere tags you can attach metadata to vSphere objects, and as a result, make it easier to filter and sort these objects. You can use the vSphere Automation APIs to automate the management of vSphere tags.

vSphere with Tanzu

Starting with vSphere 7.0, you can enable vSphere with Tanzu on an existing vSphere cluster in your environment. Create and configure namespaces on the Supervisors to run Kubernetes workloads in dedicated resource pools.

About vSphere

vSphere is the VMware software stack that implements private-cloud data center management and the on-premises component of hybrid-cloud deployments.

A vSphere installation includes one or more instances of vCenter Server configured to manage one or more virtual data centers. Each virtual data center includes one or more instances of VMware ESXi.

About ESXi

Each instance of ESXi includes management agents and the VMware hypervisor layer, which runs several virtual machines. Each virtual machine contains a guest operating system, such as Windows or Linux, capable of running IT or user applications.

vCenter Server runs as a virtual machine on an ESXi host. vCenter Server provides an independent endpoint capable of handling API requests both for vCenter Server and for the vCenter Server Management API.

vCenter Server Management Overview

vCenter Server runs on a Photon OS™ guest operating system.

vCenter Server is a collection of services designed for managing and monitoring vSphere installations. vCenter Server responds to CLI commands, requests from the vSphere Client, and API requests from custom clients. API clients can be written in a choice of several software languages.

vCenter Server is managed by CLI, Web interfaces, or API requests. These requests help you manage vCenter Server configuration, monitor resource usage, or back up and restore the vCenter Server instance. You can also use API requests to check the health of vCenter Server. This programming guide explains how to use the vSphere Automation APIs that are available to manage and monitor vCenter Server.

For more information about the capabilities of vCenter Server, see *vCenter Server Configuration*.

Understanding the vSphere Automation API

You can use the vSphere Automation REST services to automate many aspects of your vSphere environment such as configuration, authentication, certificate management, tagging, managing the life cycle of clusters, content libraries, Kubernetes cluster management, and many more.

- How Does the vSphere Automation API Work
- The vSphere Automation API in Context
- Which REST Clients to Use with the vSphere Automation API

- Why Use the vSphere Automation API Through REST

How Does the vSphere Automation API Work

The vSphere Automation API follows a resource-based REST architecture with JavaScript Object Notation (JSON) requests and responses. REST, an acronym for Representational State Transfer, is a software architectural style in which programs use the Hypertext Transfer Protocol (HTTP) to exchange serialized representations of objects between a client and a server. The client sends an HTTP request and receives an HTTP response from the server. Request and response bodies are encoded using JSON, and must conform to the JSON schema associated with each operation.

The vSphere Automation API uses the main CRUD (Create, Retrieve, Update, Delete) functions over HTTP to manage representations of the objects that the API defines. In the vSphere Automation API, these objects are represented by a collection of JSON schemas.

You use:

- HTTP GET to retrieve a representation of an object
- HTTP POST to create an object
- HTTP PATCH or PUT to modify an object
- HTTP DELETE to delete an object

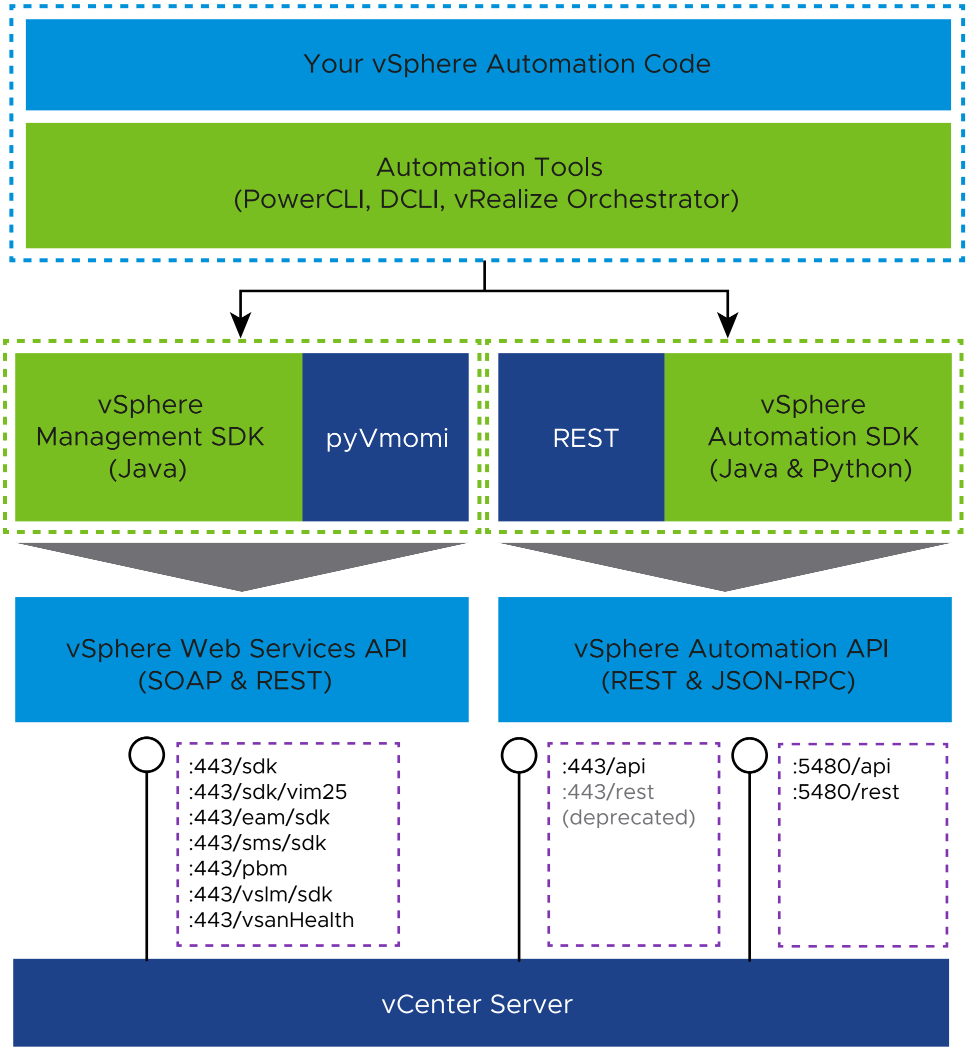
In addition, the HTTP POST operation is used for some custom actions outside of the CRUD framework such as registering or relocating virtual machines.

The HTTP verbs are also called methods. Each API operation is identified by a method and a URL. Operations can also contain a body that is composed of one or more data structures. The operation body contains parameters that are structured as key-value pairs. For more information, see vSphere Automation API Requests.

The vSphere Automation API in Context

To take full advantage of the vSphere Automation API, you must understand the broader context in which it functions.

Figure 1-1. vCenter Server Automation Architecture



vSphere Automation API: the REST API for vSphere Management

The underlying platform of the vSphere Automation API is vSphere, which consists of vCenter Server and ESXi. The vSphere Automation API is available to all licensed vSphere users.

Starting with vSphere 6.0, VMware introduced the vSphere Automation API to supplement the SOAP-based vSphere Web Services API. Since then, VMware engineers have been designing new vSphere features with a RESTful approach and these new services have been integrated into the vSphere Automation API. As a result, you can now use the vSphere Automation API to manage and automate many aspects of your vSphere environment such as vCenter Server configuration, authentication, tagging, content libraries, managing the life cycle of clusters, managing internal and external certificates, vSphere Trust Authority, vSphere with Tanzu, and so on.

However, you cannot manage all aspects of your vSphere environment with the vSphere Automation API as it does not cover all vSphere features. To expose the full set of vSphere functionalities you must use both the vSphere Automation API and the vSphere Web Services API. For both APIs, VMware has compiled SDKs that help programmers automate tasks.

vSphere Web Services API: the SOAP API for vSphere Management

The vSphere Web Services API, which is based on the SOAP protocol, provides basic vSphere functionalities such as host management, virtual machine provisioning, monitoring, vSAN and many more. It uses port 443 and the `/sdk` and `/sdk/vim25` base paths. The vSphere Web Services API can be consumed through:

- the vSphere Management SDK for Java
- the Python SDK for managing vSphere (pyVmomi)
- the Virtual Infrastructure JSON API

  Starting in vSphere version 8.0.1, you can communicate with the vSphere Web Services API by using HTTP and JSON. For more information, see the *vSphere Web Services SDK Programming Guide*.
- customer generated bindings from the WSDL files included in the vSphere Management SDK

vSphere Automation API Ports

The vSphere Automation API uses the standard HTTPS network port 443. A subset of the vSphere Automation API related to appliance configuration and life cycle is available also on port 5480. You can use the services on port 5480 immediately after installing vCenter Server and during life cycle and configuration operations when the services on port 443 are unavailable.

vSphere Automation API Base Paths

All existing and non-deprecated HTTP operations of the vSphere Automation API are available on port 443 and the `/api` base path. The APIs released up to vSphere version 7.0.2 are also available on the deprecated `/rest` base path. The `/api` base path will remain the only active base path when the `/rest` base path is removed in a future vSphere release.

vSphere Automation API Security

The vSphere Automation REST APIs are designed with security best practices in mind. The HTTP link between the client application and the vSphere Automation API is established through the Transport Layer Security (TLS) cryptographic protocol. User identity security is guaranteed by token-based authentication and by the OAuth 2.0 workflows for federated authentication.

SDKs for the vSphere Automation API

VMware supports open-source SDKs that empower developers to consume the vSphere Automation API through a programming language. The vSphere Automation SDK for Java and the vSphere Automation SDK for Python contain tools, libraries, documentation, and sample code for vSphere automation. Both SDKs are based on the JSON-RPC protocol. You can download or contribute to the vSphere Automation SDKs on GitHub.

vSphere Automation Tools

Automation tools such as PowerCLI, Data Center CLI (DCLI), and VMware Aria Automation Orchestrator provide another option for consuming the vSphere APIs. For example, you can use PowerCLI and the vSphere Automation SDK for PowerShell to communicate directly with the vSphere Automation API by using PowerShell scripts. For more information, see the *PowerCLI User's Guide*.

User Interface Tools

User interface tools such as the vSphere Client and the vCenter Server Management Interface consume the vSphere Automation API and the vSphere Web Services API in the back end.

Which REST Clients to Use with the vSphere Automation API

Client Applications for REST

You can use any REST client to communicate with the vSphere Automation API.

Some popular applications are:

- cURL - http://curl.haxx.se
- Postman - https://www.getpostman.com/
- jq parser - https://stedolan.github.io/jq/

API Explorer in the vSphere Client

Use the API Explorer, a built-in tool in the vSphere Client, to explore and try out vSphere APIs.

Why Use the vSphere Automation API Through REST

You can choose whether to access the vSphere Automation API directly through REST or through the various SDKs and automation tools binded to the API.

Possible use cases for using the vSphere Automation API through REST:

- There is no SDK available for your programming language. For example, you use PHP, JavaScript, Swift, Rust, and so on.
- Your integration footprint is small and you can't justify the overhead of installing and maintaining the vSphere Automation SDKs for Java or Python. For example, if you are designing a client application that fetches a few parameters from your vSphere environment, you might decide that installing and maintaining the Java or Python SDKs is unnecessary.

vSphere Automation API Base Path

You connect to the vSphere Automation REST API services through the /api base path.

The vSphere Automation REST API services append to the /api base path, following the host and port segments of the URL.

For example, you can reach the Session service through the following URL.

```
https://<vcenter_server_ip_address_or_fqdn>/api/session
```

For the full list of vSphere Automation API services, see the *vSphere Automation API Reference*.

Caution Starting with vSphere 7.0 Update 2, you should use the /api base path to connect to the vSphere Automation API. The old /rest base path has been deprecated and will be removed in a future release. All existing and non-deprecated HTTP operations that use the /rest base path are available on the /api base path, which will remain the only active base path after the /rest base path is removed.

vSphere Automation API Requests

To retrieve or manage resources, clients make HTTP requests to a vSphere Automation API server.

An API request through HTTP must include a method (or verb), a URL, and headers. Depending on the type of operation, the request might or might not include a body.

HTTP Method

The methods of the vSphere Automation API follow the logic of the CRUD (Create, Retrieve, Update, Delete) functions used to interact with database applications.

Table 1-1. vSphere Automation API Methods

HTTP Method/Verb	Operation Type	Operation Summary
POST	Create	Creates a resource. Also used for some custom actions such as registering or relocating virtual machines.
GET	Retrieve	Retrieves the representation of an existing resource in its current state.
PUT	Update	Updates or replaces an entire resource.
PATCH	Update	Updates or replaces a part of a resource.
DELETE	Delete	Deletes a resource.

URL

You can reach each vSphere Automation API operation through a specific URL.

The URL is typically composed of:

- the vCenter Server IP address or FQDN.
- the `/api` base path
- the package name
- the service (also called interface) name
- the resource identifier (if applicable)

```
https://<vcenter_server_ip_address_or_fqdn>/api/<package_name>/<service_name>/
<resource_identifier>
```

To determine what you want to do with the resource, you associate an HTTP verb (POST, GET, PUT, PATCH, or DELETE) to the URL.

For example, to retrieve information about a virtual machine, you contact the `VM` service of the `vcenter` package with the GET method. You pass the virtual machine identifier as a path parameter.

```
GET https://<vcenter_server_ip_address_or_fqdn>/api/vcenter/vm/<vm_identifier>
```

HTTP Headers

The HTTP headers transfer metadata associated with the API requests and responses. They are an integral part of HTTP requests and responses as they convey information about authorization, request and response content type, response caching, and so on.

You must typically set the following HTTP headers in your requests.

`Authorization`

All requests to create a vSphere Automation API session must include an `Authorization` header in the form prescribed by your identity provider. For vCenter Single Sign-On basic authentication, you must use the base-64 encoded value of your user name and password separated by a colon (`username:password`). For more information, see Chapter 2 Authentication Mechanisms.

`vmware-api-session-id`

This is a custom header that you use to authenticate your subsequent calls after you obtain a session identifier. Paste the session identifier as the header value.

`Content-Type`

This header is used to indicate the data format of the request body.

Use *application/x-www-form-urlencoded* for requests to the `Authentication Token` service.

Use *application/json* for all other requests.

Request Body

The HTTP body must be in the JSON format.

Some vSphere Automation API requests contain a body and some do not. For example, retrieve (GET) operations do not contain a body in the request.

To explore the request and response schemas, visit the *vSphere Automation API Reference*.

vSphere Automation API Responses

The vSphere Automation API responds to every HTTP request with either a success or an error status code.

The vSphere Automation API uses a specific set of HTTP success and error codes for the responses. The response structure depends on the request type: some responses include a document body, some include a string, and some are empty.

Table 1-2. HTTP Success Codes

HTTP Code	Message Type	Message Description
200	OK	The request is valid and was completed. The response includes a document body.
201	`Created`	The request is valid. The requested object was created and its identifier is returned with the response.
202	`Accepted`	The request is valid and a task was created to handle it. The response usually contains a task identifier.
204	`No Content`	The request is valid and was completed. The response does not include a body.

Table 1-3. HTTP Error Codes

HTTP Code	Error Type	Error Description
400	`ALREADY_EXISTS`	An attempt was made to create an entity but an entity with the same name or identifier already exists in that context.
400	`ALREADY_IN_DESIRED_STATE`	Indicates that an attempt to change the state of a resource or service had no effect because the resource or service is already in the desired state.
400	`CANCELED`	Indicates that an operation was canceled due to an explicit request to do so.
400	`FEATURE_IN_USE`	Indicates that an action cannot be completed because a feature is in use. For example, trying to deactivate snapshots on a virtual machine that has a snapshot.
400	`INVALID_ARGUMENT`	Indicates that the values accepted for one or more parameters are not acceptable. Examples: ■ A parameter value that is not of the expected type. ■ A parameter value that is not in the required range. ■ A parameter value that is different from the specifically allowed strings.

Table 1-3. HTTP Error Codes (continued)

HTTP Code	Error Type	Error Description
400	INVALID_ELEMENT_CONFIGURATION	Indicates that an attempt to modify the configuration of an element or a group containing the element failed due to the configuraton of the element. Examples: ■ An attempt to move a host with a fault-tolerant virtual machine out of a cluster. ■ An attempt to remove a host from a DRS cluster without putting the host into maintenance mode.
400	INVALID_ELEMENT_TYPE	Indicates that the server was unable to fulfill the request because an element of a specific type does not fit into a particular container type. Examples: ■ An attempt to put a virtual machine into a folder that can only contain hosts. ■ An attempt to attach an SCSI virtual disk to an IDE port.
400	INVALID_REQUEST	Indicates that the request is formed in such a way that the server is unable to process it. For example, sending an invalid JSON structure.
400	NOT_ALLOWED_IN_CURRENT_STATE	Indicates that the requested operation is not allowed with a resource or service in its current state. Examples: ■ Trying to upgrade the virtual hardware version of a suspended virtual machine. ■ Trying to power off, reset, or suspend a virtual machine that is not powered on.
400	RESOURCE_IN_USE	Indicates that the operation could not be completed because a resource is in use. Examples: ■ Trying to remove a datastore when there is a virtual machine registered on a host attached to the datastore. ■ Trying to add a virtual switch if the physical network adapter being bridged is already in use.
400	UNEXPECTED_INPUT	Indicates that the request body contains a parameter or a field whose name is unknown to the server.
400	UNSUPPORTED	Indicates that the operation is not supported by the underlying platform. Examples: ■ Trying to hot-plug a CPU when the current virtual machine configuration does not support CPU hot-plugging . ■ Trying to change the memory size to a value beyond the acceptable guest memory limits supported by the virtual machine's host.
400	UNVERIFIED_PEER	Indicates that an attempt to connect to an unknown or untrusted endpoint failed because the system was unable to verify the identity of the endpoint. Typically, the error data field of this error contains information about the endpoint. If you decide to trust the endpoint, the request can be resubmitted with an indication that the endpoint should be trusted. Examples: ■ The client provides an IP address or an URL of an endpoint the system should communicate with using a TLS connection, but the endpoint's TLS certificate is self-signed, expired, or otherwise untrustworthy. ■ The client provides an IP address of a host, with which the system must communicate through SSH, but SSH does not recognize the host's public key.
401	UNAUTHENTICATED	Indicates that the operation requires authentication and the user is not authenticated. For example, if the session identifier in the request header is missing or if it identifies a session that has expired.

Table 1-3. HTTP Error Codes (continued)

HTTP Code	Error Type	Error Description
403	`UNAUTHORIZED`	Indicates that the user is not authorized to perform the operation. For example, some operations might require that the user has one or more privileges over the operation or over a resource identifier but the user identified by the operation does not have the required privileges.
404	`NOT_FOUND`	Indicates that a specified resource could not be found. Examples: ■ Trying to retrieve information about a virtual machine by passing an ID that does not match an existing virtual machine. ■ Trying to remove a vSwitch by passing an ID that does not match an existing vSwitch.
404	`OPERATION_NOT_FOUND`	Indicates that the API infrastructure is unable to find the requested service or operation. For example, providing invalid REST API service name or method.
409	`CONCURRENT_CHANGE`	Indicates that a data structure, entity, or resource has been modified in comparison to some earlier point in time. For example, when the client is doing the write portion of a read-modify-write sequence and indicates that it wants the server to notify it if the data in the server has changed after the read was done, so that overwriting a change can be avoided.
500	`ERROR`	Indicates that the operation resulted in some error that does not fit into the standard error types.
500	`INTERNAL_SERVER_ERROR`	Indicates that the server encountered an unexpected condition that prevented it from fulfilling the request. Examples: ■ The operation returns a value type that does not match the operation schema. ■ The operation returns an error that is not included in the list of vSphere Automation API errors.
500	`RESOURCE_BUSY`	Indicates that the operation could not be completed because a resource is busy. For example, trying to power off a virtual machine that is in the process of being powered on.
500	`RESOURCE_INACCESSIBLE`	Indicates that the operation could not be completed because an entity is not accessible. For example, attempting to invoke some operation on a virtual machine when the virtual machine's configuration file is not accessible.
500	`TIMED_OUT`	Indicates that the operation did not complete within the allowed amount of time. The operation might or might not complete after the `TIMED_OUT` error is reported.
500	`UNABLE_TO_ALLOCATE_RESOURCE`	Indicates that the operation failed because it was unable to allocate or acquire a required resource. Examples: ■ Trying to power on a virtual machine when there are not enough licenses to do so. ■ Trying to power on a virtual machine that would violate a resource usage policy.
503	`SERVICE_UNAVAILABLE`	Indicates that the service you are trying to reach is unavailable. Examples: ■ Trying to reach the server when it is too busy. ■ Trying to reach the server while it's undergoing maintenance.

Supplementing the vSphere Automation APIs

Some features are not accessible by API. These features require direct shell access or specific user interfaces.

vSphere Client

The vSphere Client is a user interface for general management tasks. All administrative functions are available through the vSphere Client.

The vSphere Client is a cross-platform application that can connect only to vCenter Server. It has a full range of administrative functionality and an extensible plug-in-based architecture. Typical users are virtual infrastructure administrators, help desk, network operations center operators, and virtual machine owners.

vCenter Server Management Interface

The vCenter Server Management Interface is an interface for configuring, monitoring, and patching vCenter Server.

The vCenter Server Management Interface runs in a browser that connects to port 5480 of vCenter Server. The vCenter Server Management Interface provides access to all the service APIs of vCenter Server.

Direct Console User Interface

The Direct Console User Interface (DCUI) provides access to basic operations for vCenter Server management and set up.

The DCUI provides access to a subset of management functions. It provides direct access to vCenter Server if the vSphere Client and the vCenter Server Management Interface become unavailable.

After the vCenter Server startup is complete, the DCUI displays basic CPU, memory, and network information on the operator console. The root user can use the DCUI screen to configure network interfaces, DNS, and super administrator password.

vCenter Server Appliance Bash Shell

You can use the vCenter Server appliance Bash shell to access all vCenter Server commands and plug-ins that you use for monitoring, troubleshooting, and configuring the vCenter Server instance through the API. For more information about the appliance Bash shell, see *vCenter Server Configuration*.

Data Center CLI

The Data Center CLI (DCLI) is a CLI client of the vSphere Automation APIs. Almost all methods that are available in the vSphere Automation APIs are available as DCLI commands. For more information about DCLI, see *DCLI User's Guide*.

PowerCLI

PowerCLI contains modules of cmdlets based on Microsoft PowerShell. The main PowerCLI module provides cmdlets for automated administration of the vSphere environment.

PowerCLI, the most popular vSphere automation tool, provides alternative, and often more user-friendly, methods to consume the vSphere APIs. The vSphere Automation SDK for PowerShell that was introduced with VMware PowerCLI 12.4, enables you to prepare data structures and call the vSphere REST APIs directly from PowerShell. It uses the `Initialize-` and `Invoke-` cmdlet prefixes to prepare data structures and call API endpoints. This approach keeps the REST workflow intact but transfers it to a PowerShell interface. For more information, see the *Managing the vSphere Automation API with VMware PowerCLI* chapter in the *PowerCLI User's Guide*.

vSphere Web Services API

The vSphere API is exposed as a Web service, running on VMware vSphere server systems. The API provides access to the vSphere management components - the managed objects that you can use to manage, monitor, and control life-cycle operations of virtual machines and other virtual infrastructure components, such as data centers, datastores, networks, and so on. For more information about the vSphere Web Services API, see the *vSphere Web Services API Reference*.

Authentication Mechanisms 2

To perform operations in the vSphere environment, principals must authenticate to the vSphere Automation services. You can use the default vCenter Single Sign-On identity source or federate your vCenter Server to an external identity provider.

As a result from authenticating to the vSphere Automation services, client applications and users receive a session identifier. You can choose from several supported authentication mechanisms to obtain a session identifier that grants you access to the vSphere Automation services in the virtual environment.

First, you must decide whether to authenticate to vCenter Server by using the default identity provider, vCenter Single Sign-On, or leverage your enterprise configuration to an external identity provider.

Identity Provider Federation

vCenter Server acts as the default identity provider, by using its built-in vCenter Single Sign-On service to authenticate users and service accounts.

In vSphere 7.0 and later, you can delegate vSphere identity management to an external identity provider and take advantage of increased security and multifactor authentication. In this scenario, vCenter Single Sign-On is replaced by an external identity provider as the entity that performs principals' authentication.

Identity provider federation enhances the security and compliance of your applications and provides access to flexible benefits such as multifactor authentication (MFA) and automatic account synchronization.

You can federate vCenter Server authentication to:

- AD FS (vSphere 7.0 and later)
- Okta (starting in vSphere 8.0 Update 1)
- Azure AD (starting in vSphere 8.0 Update 2)

For more information, see Identity Provider Federation for vCenter Server.

Authentication Options

You can use basic and token-based authentication to authenticate principals to the vSphere Automation services. VMware encourages you to use token-based authentication as it provides more security and flexibility for your applications and users, as explained below.

Basic Authentication

Basic authentication passes a user name and password to vCenter Server.

By default, principals can use the vCenter Single Sign-On user name and password to connect to the vSphere Automation endpoint. The vSphere Automation endpoint checks whether the user name and password are present in the VMware Directory Service. On success, the system returns a session identifier for the vSphere Automation endpoint.

If your vCenter Server is federated to an external identity provider, you can use basic authentication through the OAuth 2.0 Password grant type.

Note VMware encourages you to refrain from using basic authentication and adopt token-based authentication, as it is more secure and provides more flexibility.

Token-based Authentication

Token-based authentication involves the use of an encrypted token that provides authentication and authorization data to the vSphere Automation endpoint. The vCenter Server token complies with the Security Assertion Markup Language (SAML) specification, an XML-based schema for communicating authentication data.

To acquire a SAML token, client applications issue a token request to vCenter Single Sign-On. Client applications then send the SAML token to the vSphere Automation API endpoint in exchange for a session identifier.

Starting with vSphere 7.0, you can use token-based authentication for your federated vCenter Server through a 3-step process:

1. Principals use the various OAuth 2.0 grant types with the external identity provider to obtain identity and access tokens in JSON Web Token (JWT) format.
2. Principals exchange the JWT tokens for a vCenter Single Sign-On SAML token.
3. Principals use the SAML token to obtain a session identifier from the vSphere Automation endpoint.

For more information, see Federated Authentication.

Read the following topics next:

- Authentication Terminolgy
- Authenticate to vCenter Server with vCenter Single Sign-On Credentials
- Identity Provider Federation for vCenter Server
- Federated Authentication

Authentication Terminolgy

To use the vSphere programming features effectively, you must understand a set of specific authentication terms and concepts.

Table 2-1. vCenter Single Sign-On Glossary

Term	Definition
Principal	An entity that can be authenticated, such as a user.
Identity Provider	A service that manages identity sources and authenticates principals. Examples: Microsoft Active Directory Federation Services (AD FS) and vCenter Single Sign-On.
Identity Source (Directory Service)	Stores and manages principals. Principals consist of a collection of attributes about a user or a service account such as name, address, email, and group membership. Examples: Microsoft Active Directory and VMware Directory Service (vmdir).
Authentication	The means of determining whether someone or something is, in fact, who or what it declares itself to be. For example, users are authenticated when they provide their credentials, such as smart cards, user name and correct password, and so on.
Authorization	The process of verifying what objects principals have access to.
Token	A signed collection of data comprising the identity information for a given principal. A token might include not only basic information about the principal such as email address and full name, but also, depending on the token type, the principal's groups and roles.
vmdir	VMware Directory Service. The internal (local) LDAP repository in vCenter Server that contains user identities, groups, and configuration data.
OAuth 2.0	An open authorization standard that enables the exchange of information among principals and web services without exposing principals’ credentials.
OpenID Connect (OIDC)	Authentication protocol based on OAuth 2.0 that augments OAuth with user-identifying information. It is represented by the ID token that the authorization server returns together with the access token during OAuth authentication. vCenter Server uses OIDC capabilities when interacting with Active Directory Federation Services (AD FS) and Okta.
System for Cross-domain Identity Management (SCIM)	The standard for automating the exchange of user identity information between identity domains or IT systems.

Table 2-1. vCenter Single Sign-On Glossary (continued)

Term	Definition
VMware Identity Services	Starting in version 8.0 Update 1, VMware Identity Services is a built-in container within vCenter Server that you can use for identity federation to external identity providers. It serves as an independent identity broker within vCenter Server and comes with its own set of APIs. Currently, Okta is the only external identity provider supported by VMware Identity Services.
Tenant	A VMware Identity Services concept. A tenant provides a logical separation of data from other tenants' data in one and the same virtual environment.
JSON Web Token (JWT)	A token format defined by the OAuth 2.0 specification. A JWT token carries authentication and authorization information about a principal.
Relying party	A relying party "relies" on the authorization server, VMware Identity Services or AD FS, for identity management. For example, through federation, vCenter Server establishes relying party trust to VMware Identity Services or AD FS.

Authenticate to vCenter Server with vCenter Single Sign-On Credentials

You can use your vCenter Single Sign-On credentials with basic authentication to establish a session with a vCenter Server system.

The single-factor basic authentication is the default method to authenticate to vCenter Server. You connect to the vSphere Automation API endpoint with your vCenter Single Sign-On user name and password. The vSphere Automation endpoint checks whether the user credentials are present in vmdir, a vCenter Single Sign-On component. On success, the system returns a session identifier that you can use for subsequent API calls.

Prerequisites

To perform this task, you must have:

- The IP address or FQDN of your vCenter Server system
- Valid vCenter Single Sign-On user name and password

Procedure

1 Call the `Session` interface with the base-64 encoded value of your vCenter Single Sign-On user name and password separated by a colon (`username:password`) in the authorization header.

```
POST https://<vcenter_server_ip_address_or_fqdn>/api/session
Authorization: Basic <username:password>
```

On success (status code 201), the system returns a session identifier. For example, `"a5adf880fc8e1e2944a541a4c7d74465"`.

2 To authenticate subsequent API calls, pass the session identifier as a request header.

```
vmware-api-session-id: <session_ID>
```

Note VMware encourages you to refrain from using basic authentication and adopt token-based authentication, as it is more secure and provides more flexibility.

Identity Provider Federation for vCenter Server

Starting with vSphere 7.0, you can federate your vCenter Server to external identity providers through the OAuth 2.0 authorization framework and the OpenID Connect (OIDC) authentication protocol.

With identity federation, you can use the same identity source for your vCenter Server that you use for your other federated desktop and cloud applications.

vCenter Server Identity Provider Federation Basics

In vSphere 7.0 and later, vCenter Server supports federated authentication. In this scenario, when a user logs in to vCenter Server, vCenter Server redirects the user login to the external identity provider. The user credentials are no longer provided to vCenter Server directly. Instead, the user provides credentials to the external identity provider. vCenter Server trusts the external identity provider to perform the authentication. In the federation model, users never provide credentials directly to any service or application but only to the identity provider. As a result, you "federate" your applications and services, such as vCenter Server, with your identity provider.

Why Is Identity Provider Federation Useful

Federating vCenter Server to an enterprise identity provider alleviates the burden of identity management and provides flexible options such as multifactor authentication (MFA), automatic registration and termination of users across services, and many more. Identity provider federation uses token-based authentication and minimizes the risk of bad actors acquiring protected credentials such as user names and passwords. Identity provider federation also helps your organization with compliance as various international standards already require MFA to guarantee data security. In addition, with identity provider federation, you can automate vCenter Server user management because you utilize the users and groups from your main enterprise identity source, for example Microsoft Active Directory.

vCenter Server External Identity Provider Support

vCenter Server supports the following external identity providers:

- AD FS (vSphere 7.0 and later)
- Okta (starting in vSphere 8.0 Update 1 and later)
- Azure AD (starting in vSphere 8.0 Update 2)

Identity Provider Federation to Microsoft Active Directory Federation Services (AD FS)

In vSphere 7.0 and later, you can activate identity federation to Microsoft Active Directory Federation Services (AD FS). In this scenario, vCenter Server federates directly to the enterprise identity provider, AD FS, without the use of an authentication intermediary service.

You can configure federation to AD FS with the help of the vSphere Automation API. For more information, see Federate vCenter Server to Microsoft Active Directory Federation Services (AD FS).

Identity Provider Federation to Okta and Azure AD Through VMware Identity Services - vCenter Server

Starting from vSphere 8.0 Update 1, you can activate federation to Okta as the identity provider. Starting from vSphere 8.0 Update 2, you can activate federation to Azure AD as the identity provider. Both configurations use VMware Identity Services - vCenter Server, an authentication intermediary that functions as a built-in container within vCenter Server. With VMware Identity Services - vCenter Server, you can configure principals to authenticate to vCenter Server by using an external identity provider.

vCenter Server supports only one configured external identity provider (one source), and the `vsphere.local` identity source (local source). You cannot use multiple external identity providers. vCenter Server Identity Provider Federation uses OpenID Connect (OIDC) for user login to vCenter Server.

Important Configuring VMware Identity Services - vCenter Server for Okta and Azure AD is not possible through the vSphere Automation API. To federate vCenter Server to Okta or Azure AD, you must use the vSphere Client. For more information, see `Configure vCenter Server Identity Provider Federation for Okta` in the *vSphere Authentication Guide*.

Authentication to a Federated vCenter Server

Once you have configured vCenter Server to point to AD FS, Okta, or Azure AD, you can use the vSphere Automation API to authenticate principals to your vCenter Server. Depending on the type of user or application you want to authenticate, you can choose among different token-based authentication options. vSphere supports the various OAuth 2.0 grant types. For more information, see Federated Authentication.

Federate vCenter Server to Microsoft Active Directory Federation Services (AD FS)

You can federate vCenter Server to Microsoft Active Directory Federation Services (AD FS) as an external identity provider by using the vCenter Server `Identity Providers` service.

Prerequisites

Note This process of configuring an AD FS identity provider requires that you have administrative access to both your vCenter Server and your AD FS server. During the configuration process, you enter information first in your vCenter Server, then in your AD FS server, then in your vCenter Server.

Active Directory Federation Services requirements:

- AD FS for Windows Server 2016 or later must already be deployed.
- AD FS must be connected to Active Directory.
- An Application Group for vCenter Server must be created in AD FS as part of the configuration process. See the VMware knowledge base article at https://kb.vmware.com/s/article/78029.
- An AD FS server certificate (or a CA or intermediate certificate that signed the AD FS server certificate) that you add to the Trusted Root Certificates Store.
- You have created a vCenter Server administrators group in AD FS that contains the users you want to grant vCenter Server administrator privileges to.

For more information about configuring AD FS, see the Microsoft documentation.

vCenter Server and other requirements:

- vSphere 7.0 or later
- vCenter Server must be able to connect to the AD FS discovery endpoint, and the authorization, token, logout, JWKS, and any other endpoints advertised in the discovery endpoint metadata.
- You need the **VcIdentityProviders.Manage** privilege to create, update, or delete a vCenter Server Identity Provider that is required for federated authentication. To limit a user to view the Identity Provider configuration information only, assign the **VcIdentityProviders.Read** privilege.

Procedure

1 Authenticate to the vSphere Automation API endpoint and establish a session.

2 Add your AD FS root CA certificate to the Trusted Root Certificates Store.

See Managing Certificates.

3 Fill in the `oidc` data structure by using the Application Group configuration from AD FS.

Parameter	Description
`discovery_endpoint`	The OpenID address of the AD FS server.
`client_id`	The client identifier of the AD FS Application Group.
`client_secret`	The secret shared between the client and the provider.
`claim_map`	This parameter is required but not applicable to AD FS. Use an empty array **`[]`**.

4 Fill in the `active_directory_over_ldap` data structure.

Parameter	Description
`user_name`	The user name of a user in the domain who has a minimum of read-only access to the base Distinguished Name (DN) for users and groups.
`password`	The password of a user in the domain who has a minimum of read-only access to the base DN for users and groups.
`users_base_dn`	The base DN for users in the Active Directory environment connected to AD FS that you want to be able to federate with vCenter Server.
`groups_base_dn`	The base DN for groups in the Active Directory environment connected to AD FS that you want to be able to federate with vCenter Server.
`server_endpoints`	Active Directory server endpoints. At least one Active Directory server endpoint must be set. Use the format: **`ldap://`*`<hostname>`*`:`*`<port>`*** or **`ldaps://`*`<hostname>`*`:`*`<port>`***. The port is typically 389 for LDAP connections and 636 for LDAPS connections. For Active Directory multi-domain controller deployments, the port is typically 3268 for LDAP and 3269 for LDAPS.
`cert_chain`	The SSL certificate chain in base64 encoding. You can skip this parameter only if all the Active Directory server endpoints use the LDAP (and not the LDAPS) protocol.

5 Add the identity provider by using the `Identity Providers` service.

a Fill in the request body parameters.

Parameter	Description
`config_tag`	The configuration type of the identity provider. The possible values are *Oauth2* and *Oidc*. For AD FS federation, use *Oidc*.
`name`	The user-friendly name for the identity provider. For proper configuration, you must use the exact string *Microsoft ADFS*.
`upn_claim`	The name of the claim in the AD FS JWT token that contains the user principal name (UPN) of the user that is logging in. You must use the same value that you used when you set up the AD FS Application Group. The procedure from the article in the prerequisites uses *upn*. If unset, the default value is *acct*.
`groups_claim`	The name of the claim in the AD FS JWT token that contains the group membership of the user that is logging in. You must use the same value that you used when you set up the AD FS Application Group. The procedure from the article in the prerequisites uses *group*. If unset, the groups for the subject consist of the groups in `group_names` and `group_ids` claims.
`is_default`	Set to *true*. Specifies whether the provider is the default provider. Setting `is_default` to *true* makes all other providers non-default. If unset: ■ In case it is the first created provider, it is set as the default provider. ■ In case it is not the first created provider, it is not set as the default provider.
`oidc`	Data structure for `oidc`.
`idm_protocol`	The communication protocol used to connect to AD FS to search for users and groups when assigning permissions in vCenter Server. You must use *LDAP*. If unset, no communication protocol is configured for the users and groups search.
`active_directory_over_ldap`	Data structure for `active_directory_over_ldap`.

b Invoke the operation.

```
POST https://<vcenter_server_ip_address_or_fqdn>/api/vcenter/identity/providers
```

The operation returns the ID of the provider you created.

6 Configure vCenter Server permissions for Active Directory users or groups in your AD FS environment.

You can do this in two ways:

- Add a user from your AD FS environment to a group in vCenter Server.
- Configure Global Permissions for an AD FS user.

Note In vSphere 8.0 and later, you cannot configure permissions through the vSphere Automation API. Instead, you use either the vSphere Client or the vSphere Web Services API. For more information, see the *vSphere Authentication Guide* or the *vSphere Web Services SDK Programming Guide*.

7 (Optional) Copy the two redirect URIs from the Identity Provider Configuration page in the vSphere Client and add them to your AD FS Application Group.

Note You must do this step to enable logging in to vCenter Server through AD FS by using the vSphere Client.

Results

You configured vCenter Server to use AD FS as the identity provider.

Federated Authentication

If your vCenter Server is federated to an enterprise identity provider, such as Microsoft Active Directory Federation Services (AD FS), Okta, or Azure AD, you authenticate by using the OAuth 2.0 grant types. The vSphere implementation of OAuth 2.0 uses the OpenID Connect (OIDC) authentication protocol.

vCenter Server External Identity Provider Support

vCenter Server supports the following external identity providers:

- AD FS (vSphere 7.0 and later)
- Okta (starting in vSphere 8.0 Update 1 and later)
- Azure AD (starting in vSphere 8.0 Update 2)

What is OAuth 2.0

OAuth 2.0 is an open authorization standard that allows information exchanges among web services without exposing sensitive data such as user names and passwords. OAuth 2.0 boosts security and prevents malicious actors from stealing users' credentials over the Internet. OAuth 2.0 is an open standard and can be customized according to the organizational requirements.

You can use different OAuth 2.0 grant types to authenticate your applications to vCenter Server. The OAuth grant types represent different methods by which a principal can obtain an access token from an authorization server. In the vSphere implementation, the access token is used to obtain a SAML token and authenticate principals to vCenter Server.

OAuth 2.0 Grant Types

Password

In the Password grant type, the client application exchanges the user's credentials, user name and password, for an access token, and optionally an ID token (JWT tokens), from the authorization server. OAuth 2.0 has been designed to prevent the exchange of user credentials, so the Password grant type must be avoided whenever possible, especially for third-party application authentication. The latest *OAuth 2.0 Security Best Current Practice* recommends avoiding the Password grant type altogether. As a single-factor method, the Password grant type is incompatible with the requirements of multifactor authentication (MFA). For more information, see Authenticate to an AD FS-Federated vCenter Server by Using the Password Grant Type.

Important The Password grant type is disallowed by the latest *OAuth 2.0 Security Best Current Practice*. The Password grant is excluded entirely from OAuth 2.1.

Authorization Code

In the Authorization Code grant type, the client application uses an authorization code to obtain JWT tokens from the authorization server. This grant type requires a user to verify an authentication request through a web browser. The Authorization Code grant type is compatible with multi-factor authentication and guarantees a substantial degree of security. This grant type is appropriate for regular web applications running on a server. For more information, see Authenticate to an AD FS-Federated vCenter Server by Using the Authorization Code Grant Type.

Client Credentials

The Client Credentials grant type permits web services to authenticate to other web services without the involvement of a user. This grant type is useful for applications that are resource owners and can take advantage of headless, or backend automation, when a physical user is not present. For example, a client application might use this grant type to authenticate and fetch details about a virtual environment.

Refresh Token

The Refresh Token grant type is used by clients to exchange an expiring access token for a refresh token that can be used to receive a new access token. With this grant type, a client application can fetch a valid access token without asking the user to log in again. It allows client applications to provide a seamless user experience without further interaction with the user after timeouts.

What Is OpenID Connect (OIDC)

The vSphere implementation of OAuth 2.0 uses the OpenID Connect (OIDC) authentication protocol. With OIDC, the authorization server returns an ID token containing data about who the users are and what permissions they have on the resource server. As a result, vCenter Server can identify users and grant them the respective permissions that are defined on the authorization server.

AD FS Federation Workflows

Use these authentication workflows when your vCenter Server is federated to Microsoft Active Directory Federation Services (AD FS).

You can use the following OAuth 2.0 grant types:

- Password (not recommended)
- Authorization Code
- Client Credentials
- Refresh Token

Authenticate to an AD FS-Federated vCenter Server by Using the Password Grant Type

If your vCenter Server is federated to AD FS, you can authenticate with the OAuth 2.0 Password grant type.

You can use the Password grant type to exchange user credentials for an access token and an ID token from the authorization server. From the user's perspective, the password grant type functions exactly as basic authentication with the only difference that you use your AD FS, and not your local vCenter Single Sign-On credentials.

The Password grant type is possible among native apps but is not recommended for authentication to third-party apps. Modern security best practices require that primary user credentials such as passwords do not leave the native API environment. Therefore, it is more secure to use other OAuth 2.0 grant types for authentication to third-party apps.

To use the Password grant type with AD FS, you provide your AD FS user name and password to the `Common Infrastructure Services (CIS) Session` endpoint.

In the background, the client application calls the authorization server (AD FS) and obtains an access token and an ID token in JWT format. vCenter Single Sign-On converts the JWT tokens into a SAML token which is used to obtain a session identifier and authenticate your application to vCenter Server.

Important The Password grant type is disallowed by the latest *OAuth 2.0 Security Best Current Practice*. The Password grant is excluded entirely from OAuth 2.1.

Prerequisites

- Verify that your vCenter Server is federated to AD FS. For more information, see *Federate vCenter Server to Active Directory Federation Services (AD FS).*
- You must have an AD FS account and user credentials with the necessary permissions to view and manage vCenter Server.
- You must register an OAuth client for your application on the authorization server (AD FS).

Procedure

1 Call the Session interface with the base-64 encoded value of your AD FS user name and password separated by a colon (username:password) in the authorization header.

```
POST https://<vcenter_server_ip_address_or_fqdn>/api/session
Authorization: Basic <username:password>
```

On success (status code 201), the system returns an authentication session identifier.

2 To authenticate subsequent API calls, pass the session identifier as a request header.

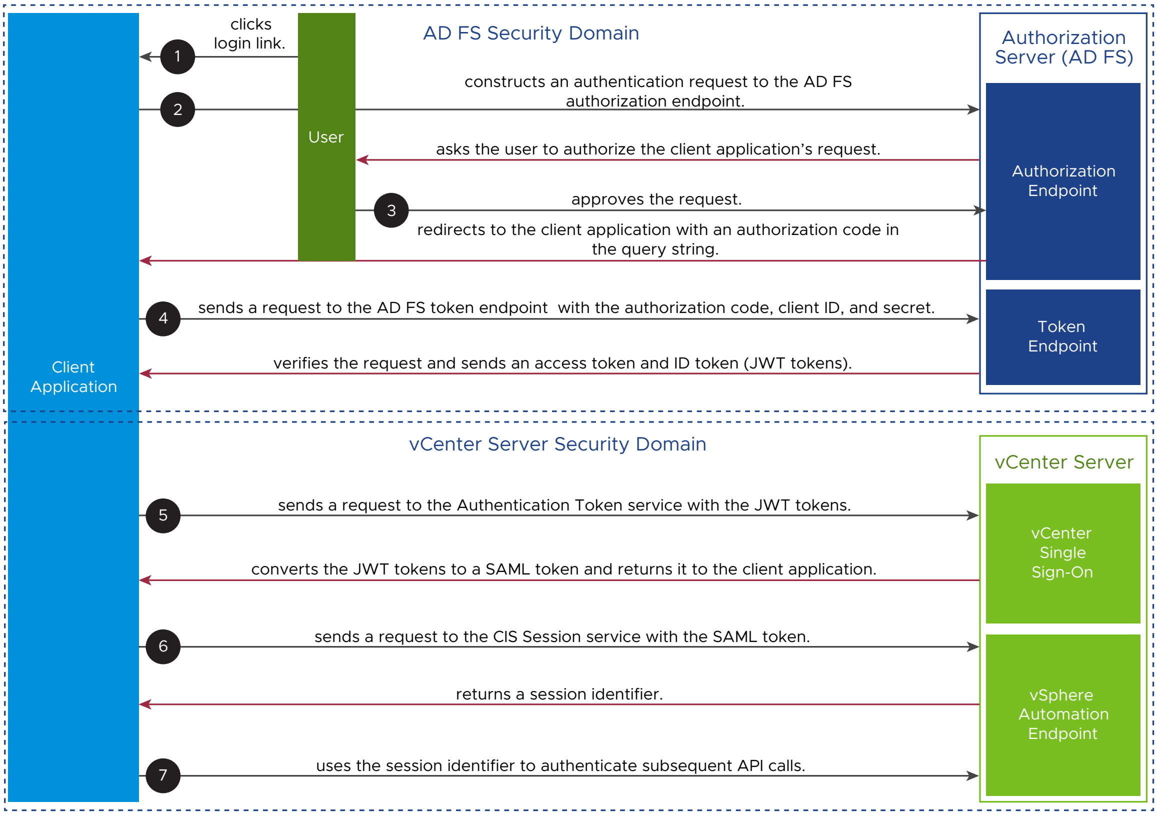
```
vmware-api-session-id: <session_ID>
```

Authenticate to an AD FS-Federated vCenter Server by Using the Authorization Code Grant Type

If your vCenter Server is federated to an enterprise identity provider, such as Active Directory Federation Services (AD FS), you can authenticate your client applications by using the OAuth 2.0 Authorization Code grant type.

You can use the Authorization Code grant type if you want to authenticate client apps to vCenter Server through multifactor authentication (MFA). This grant type is applicable for regular web applications that involve a human user interacting with a server.

Figure 2-1. Authentication to an AD FS-Federated vCenter Server Through the Authorization Code Grant Type



AD FS Security Domain
Client Application
User
1
clicks login link.
2
constructs an authentication request to the AD FS authorization endpoint.
asks the user to authorize the client application’s request.
3
approves the request.
redirects to the client application with an authorization code in the query string.
4
sends a request to the AD FS token endpoint with the authorization code, client ID, and secret.
verifies the request and sends an access token and ID token (JWT tokens).
Authorization Server (AD FS)
Authorization Endpoint
Token Endpoint
vCenter Server Security Domain
5
sends a request to the Authentication Token service with the JWT tokens.
converts the JWT tokens to a SAML token and returns it to the client application.
6
sends a request to the CIS Session service with the SAML token.
returns a session identifier.
7
uses the session identifier to authenticate subsequent API calls.
vCenter Server
vCenter Single Sign-On
vSphere Automation Endpoint

A principal authenticates to vCenter Server through the Authorization Code grant type as follows.

Prerequisites

- Verify that your vCenter Server is federated to AD FS. For more information, see *Federate vCenter Server to Active Directory Federation Services (AD FS).*
- You must have an AD FS account and user credentials with the necessary permissions to view and manage vCenter Server.
- You must register an OAuth client for your application on the authorization server (AD FS).

Procedure

1. The user clicks **Login** within the client application.
2. The client application constructs a request to the AD FS authorization endpoint.

   To retrieve the authorization endpoint, you must query the AD FS discovery endpoint at, for example, `https://<your_adfs_server.com>/adfs/.well-known/openid-configuration`.

   The client application's request must include the application's client ID and a redirect URI that tells the authorization server where to redirect the user after approving the request.

   As a result, the authorization server opens a prompt page asking the user to authorize the client application's request.
3. The user confirms the request.

   The browser is redirected back to the client application with an authorization code in the query string.
4. The client application sends a request to the AD FS token endpoint with the authorization code, client ID, and secret.

   The authorization server verifies the request and sends an access token and an ID token (JWT tokens).
5. The client application sends the JWT tokens to vCenter Single Sign-On by using the `Authentication Token` service.

   For example,

   ```
   curl --location --request POST 'https://<vcenter_server_ip_address_or_fqdn>/api/vcenter/
   authentication/token' \
   --header 'Authorization: Bearer <ACCESS_TOKEN>' \
   --header 'Content-Type: application/x-www-form-urlencoded' \
   --data-urlencode 'subject_token_type=urn:ietf:params:oauth:token-type:access_token' \
   --data-urlencode "subject_token=<ACCESS_TOKEN>" \
   --data-urlencode "actor_token_type=urn:ietf:params:oauth:token-type:id_token" \
   --data-urlencode "actor_token=<ID_TOKEN>" \
   --data-urlencode 'grant_type=urn:ietf:params:oauth:grant-type:token-exchange' \
   --data-urlencode 'requested_token_type=urn:ietf:params:oauth:token-type:saml2'
   ```

   vCenter Single Sign-On returns a SAML token.

6 The client application calls the `Common Infrastructure Services (CIS) Session` service.

By using the `CIS Session` service, principals can manage session tokens - including creation and deletion of session identifiers, and obtaining information about sessions.

a Base-64 decode the SAML token, compress it, and then Base-64 encode it again.

b Call the `Session` service with the SAML token.

For example,

```
curl --location --request POST 'https://<vcenter_server_ip_address_or_fqdn>/api/
session' \
--header 'Authorization: SIGN token=<COMP_TOKEN>'
```

On success (status code 201), the vSphere Automation endpoint returns a session identifier in the response.

7 To authenticate, the client application uses the session identifier in subsequent API calls to the vSphere Automation API endpoints.

For example,

```
curl --location --request POST 'https://<vcenter_server_ip_address_or_fqdn>/api/cis/
tagging/category' \
--header 'vmware-api-session-id: <session_ID>'
```

Bash Shell Example: Get JWT Tokens, Exchange for SAML, and Obtain Session Identifier

This Bash script illustrates the use case where you get JWT tokens from the authorization server (AD FS or VMware Identity Services - vCenter Server), exchange them for a vCenter Server SAML token, and obtain an authentication session identifier from the vSphere Automation endpoint.

This script consists of three parts:

1 Obtain JWT tokens from the authorization server (AD FS or VMware Identity Services - vCenter Server) by using the OAuth 2.0 Password grant type.

2 Exchange the JWT tokens for a vCenter Server SAML token.

3 Use the SAML token to obtain a session identifier for the vSphere Automation API.

```
#!/bin/bash

: '
Variable definitions:
$vcip = The IP address or FQDN of your vCenter Server.
$ACCESS_TOKEN = The access token in JWT format that you received from the authentication
server.
$ID_TOKEN = The ID token in JWT format that you received from the authentication server.
'
if [ -z "$1" ] || [ -z "$2" ] || [ -z "$3" ] || [ -z "$4" ] || [ -z "$5" ] || [ -z "$6" ];
then
  echo "Usage: <vc-ip> <pwgrant-userid> <pwgrant-password> <client-id> <client-secret> <token-
```

```
endpoint>"
  exit 0
fi

vcip="$1"
userid="$2"
password="$3"
clientid="$4"
clientsecret="$5"
tokenendpoint="$6"

echo "Obtaining JWT access and ID tokens for user $userid ..."

PWGRANT_OUTPUT=$(curl -k --silent --location -u "$clientid:$clientsecret" --request POST
"$tokenendpoint" \
  --header "Content-Type: application/x-www-form-urlencoded" \
  --data-urlencode "grant_type=password" \
  --data-urlencode "username=$userid" \
  --data-urlencode "password=$password")
ACCESS_TOKEN=$(echo $PWGRANT_OUTPUT | jq -r '.access_token')
ID_TOKEN=$(echo $PWGRANT_OUTPUT | jq -r '.id_token')

echo
echo "Access token: $ACCESS_TOKEN"
echo
echo "ID token: $ID_TOKEN"
echo

echo "Exchanging JWT tokens for vCenter SAML token ..."

TOKEN_EXCHANGE_OUTPUT=$(curl -k --silent --location --request POST "https://$vcip/api/vcenter/
authentication/token" \
  --header "Content-Type: application/x-www-form-urlencoded" \
  --header "Authorization: Bearer $ACCESS_TOKEN" \
  --data-urlencode "subject_token_type=urn:ietf:params:oauth:token-type:access_token" \
  --data-urlencode "subject_token=$ACCESS_TOKEN" \
  --data-urlencode "actor_token_type=urn:ietf:params:oauth:token-type:id_token" \
  --data-urlencode "actor_token=$ID_TOKEN" \
  --data-urlencode "grant_type=urn:ietf:params:oauth:grant-type:token-exchange" \
  --data-urlencode "requested_token_type=urn:ietf:params:oauth:token-type:saml2")

echo
echo "$TOKEN_EXCHANGE_OUTPUT"
echo

SAML_TOKEN=$(echo $TOKEN_EXCHANGE_OUTPUT | jq -r '.access_token')

echo "vCenter SAML token: $SAML_TOKEN"
echo

echo "Establishing vCenter session with SAML token ${SAML_TOKEN::7}..."
echo
COMP_TOKEN=$(echo $SAML_TOKEN | base64 -d | gzip | base64 -w0)

SESSION_OUTPUT=$(curl -k --silent --location --request POST "https://$vcip/api/session" \
  --header "Authorization: SIGN token=\"$COMP_TOKEN\"")

echo "Create Session Response: $SESSION_OUTPUT"

SESSION_ID=$(echo "$SESSION_OUTPUT" | tr -d '"')

echo "Tagging categories:"
curl -k "https://$vcip/api/cis/tagging/category" --header "vmware-api-session-id: $SESSION_ID"

echo
echo "Done!"
```

ESXi Hosts

3

Use the vSphere Automation APIs to run general operations on the ESXi hosts in your vSphere environment.

You can retrieve information about the hosts, create a standalone host, disconnect, and reconnect an ESXi host to a vCenter Server system.

Read the following topics next:

- How Do I Retrieve Information About ESXi Hosts
- How Do I Add a Standalone ESXi Host to vCenter Server
- How Do I Disconnect and Reconnect ESXi Hosts
- How Do I Configure ESXi Entropy

How Do I Retrieve Information About ESXi Hosts

You retrieve information about the ESXi hosts running in a vCenter Server instance by listing only the ESXi hosts that you are interested in.

To filter the ESXi hosts on a vCenter Server instance and get only the ones you want, use the `GET https://`*`<vcenter_ip_address_or_fqdn>`*`/api/vcenter/host` HTTP request and specify your filter criteria through the query parameters. Combine several filter criteria by including one or more of the following parameters:

- The name or unique identifier of the host.
- Clusters, data centers, or folders that contain the host.
- Connection state of the host which can be one of the following: `CONNECTED`, `DISCONNECTED`, or `NOT_RESPONDING`.
- Power state of the host which can be one of the following: `POWERED_OFF`, `POWERED_ON`, or `STANDBY`.

The request returns a `HostSummary` JSON object for each host that matches your filter. You can view information for up to 2500 hosts.

How Do I Add a Standalone ESXi Host to vCenter Server

You can use the vSphere AutomationREST APIs to add a standalone host to a vCenter Server instance.

Add a single ESXi host to a vCenter Server instance by using the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/host` request and submitting a `HostCreateSpec` JSON object in the request body. When you construct the host object, make sure that you set values for the IP address or the DNS resolvable host name and the administrator credentials.

How Do I Disconnect and Reconnect ESXi Hosts

You can use the vSphere Automation APIs to connect ESXi hosts to a vCenter Server instance and make the hosts managed. You can temporarily disconnect a managed host from a vCenter Server instance and reconnect the host, for example, to refresh the ESX agents on the host.

When you add a host to a vCenter Server instance, the host is connected to vCenter Server and becomes a managed host. To disconnect a managed host from a vCenter Server instance, use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/host/<host_id>?action=disconnect` request and submit the host identifier as a path parameter. The managed host and its associated virtual machines remain in the inventory but vCenter Server temporarily stops managing and monitoring them.

To reconnect a managed host to a vCenter Server instance, use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/host/<host_id>?action=connect` request and submit the host identifier as a path parameter. As a result, the connection status of the host changes, and vCenter Server resumes managing the host and its associated virtual machines.

If you want to delete a host and all its associated virtual machines from the inventory, you can remove the host from the vCenter Server instance. To delete a disconnected host from a vCenter Server instance, use the `DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/host/<host_id>` and submit the host identifier as a path parameter.

How Do I Configure ESXi Entropy

You can use the vSphere Automation API to feed external entropy data to an ESXi host in your inventory. You can also query the current entropy levels on the host and add external entropy data when needed.

Entropy is a measure of the randomness or diversity of a data-generating function. In releases prior to vSphere 8.0 Update 1, ESXi supported only CPU-based entropy data generated from interrupts or manufacturer provided interfaces, such as RDSEED and RDRAND. High-quality entropy is important for the proper functioning of security-related operations such as generating encryption keys for secure communication over the network. Starting with vSphere 8.0 Update 1, you can add external entropy sources to an ESXi host and in this way ensure the high quality of the entropy data on that host. You provide external entropy data by using devices such as hardware security modules (HSMs) which are FIPS 140-3 and EAL4 certified.

You can configure the ESXi entropy sources by using the VMkernel boot options. To use external entropy sources, set the `entropySources` value to more than or equal to **8**. For more information about how to set the desired entropy sources by using the VMkernel boot options, see Controlling ESXi Entropy in the *vSphere Security* documentation.

You can also configure external entropy sources in the kickstart file for the ESXi scripted installation. See Configuring External Entropy Sources During Scripted Installation.

Note If a host is configured to use only external entropy sources, that is, `entropySources` is set to **8**, you must keep supplying the external entropy data through the vSphere Automation API. In case the entropy in the host gets exhausted, the host becomes unresponsive and might require a hard reboot or re-installation to recover the host from this situation.

Querying Entropy Data on a Host

To retrieve details about the external entropy available on an ESXi host, use the `ExternalPool` service. You must have the **Host.Entropy.Read** privilege.

You can check whether an external entropy source is added to a host by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/host/<host_id>/entropy/external-pool` HTTP request and passing the host ID as a path parameter. The operation returns an `ExternalPool.Info` JSON object in the response body that contains detailed information about the external entropy data on the host.

Table 3-1. Details for the External Entropy Data on a Host

Property	Description
`status`	Indicates whether an external entropy source is added for a host.
`capacity`	Shows the maximum capacity of external entropy data in bytes that a host can store in the VMkernel entropy pool. When you feed the host with additional external entropy data, make sure that you do not exceed this maximum capacity. Otherwise, all extra entropy data is discarded.
`currently_available`	Indicates the current amount of entropy data in bytes available in the VMkernel entropy pool on the host.

Table 3-1. Details for the External Entropy Data on a Host (continued)

Property	Description
`inactive_source_timeout`	Indicates the amount of time in seconds that can elapse without any activity between the host and the external entropy source. Your application must check periodically the levels of entropy data on the host and send entropy data from the external source when required. When this timeout exceeds an `esx.audit.entropy.external.source.disconnected` VMkernel Observation (VOB) is logged. This system event suggests possible loss of connection to the external entropy source. For the full list of available entropy VOBs in vSphere 8.0, see the VMware knowledge base article at https://kb.vmware.com/s/article/89074.
`low_watermark`	Indicates the threshold in bits sufficient for the in-memory cache to operate successfully. When the in-memory entropy cache is running low and the threshold is reached, an `esx.audit.entropy.available.low` VOB is logged. For more information about the entropy VOBs, see the VMware knowledge base article at https://kb.vmware.com/s/article/89074.

Adding External Entropy Data to a Host

To add entropy data from an external entropy source to a host, you must have the **Host.Entropy.Write** privilege. Use the `ExternalPool` service and send the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/host/<host_id>/entropy/external-pool?action=add` request. Pass the host ID as a path parameter and an `ExternalPool.AddSpec` data structure in the request body. The `ExternalPool.AddSpec` JSON object contains the Base64 encoded external entropy data. You must convert the binary data coming from the external entropy source to Base64 format so that the host can consume it. The request returns an `ExternalPool.AddResult` JSON object in the response body, if the request succeeds.

When the external entropy data reaches the host, the entropy daemon stores it first in the in-memory cache and the storage cache. Then the external entropy data is pushed to the VMkernel entropy pool from which it is fed to the applications in need.

The entropy data in the storage cache persists on the host disk and is only consumed during the ESXi booting. The default storage file size is 4MiB and can be configured through the ESXCLI commands. See the *ESXCLI Command Reference* documentation.

Managing the Life Cycle of Hosts and Clusters 4

You can manage the life cycle of ESXi hosts collectively by using the vSphere Lifecycle Manager feature through the vSphere Automation API. Starting with vSphere 8.0, you can manage the lifecycle of a standalone ESXi host by using an image only through the vSphere Lifecycle Manager automation API.

You can automate the life cycle management of a cluster or standalone host by performing the following operations:

- Retrieve information about the current state of the cluster or the standalone host.
- Create a desired state that includes a specific version of the ESXi host. You can also add some compatible partner software and firmware components and add-ons.
- Validate the desired state to detect any discrepancies between the desired state and the host hardware.
- Check the compliance of a cluster or host against the desired state and determine whether some additional steps must be taken to ensure the success of the cluster or host remediation.
- Apply the desired state on a cluster or a standalone host.

You can use the vSphere Lifecycle Manager to manage the life cycle of hosts in a cluster that meet the following requirements:

- Hosts must be of version 7.0 and later.
- Hosts must be stateful.
- All hosts in the cluster must be from the same vendor and with identical hardware.
- The hosts must include only integrated solutions, such as VMware vSAN™, vSphere with Tanzu, NSX and VMware vSphere® High Availability.

A standalone host is a host that is added to a vCenter Server instance but is not part of any cluster. For more information about how to add, connect, and disconnect standalone host, see Chapter 3 ESXi Hosts. You can manage the life cycle of a standalone host by performing almost all vSphere Lifecycle Manager operations that you can perform on a cluster level. The only limitation for managing the life cycle of a standalone host through the vSphere Automation API, is that you can't update the firmware of the host.

Read the following topics next:

vSphere Lifecycle Manager Terms

You must understand the basic terminology that is used within this chapter to be able to use the vSphere Lifecycle Manager functionality efficiently.

vSphere Lifecycle Manager Terminology

Term	Definition
Upgrade, update, and patch	You can upgrade to another major version of the software running on an ESXi host, and install patches and updates that include smaller changes, bug fixes, or other small improvements.
Depot	A depot is a well-defined folder structure that is used for distributing payloads and their metadata. Depots are consumed by different products and features such as the vSphere Lifecycle Manager and ESXCLI. The vSphere Lifecycle Manager works with three types of depots: online, offline, and UMDS. See Software Depots.
Component	A component is the smallest unit that the vSphere Lifecycle Manager uses during the installation and update processes. Software vendors use components to encapsulate a group of payloads that can be managed together.
Base image	A base image is a collection of components that shape the bootable ESXi used for the installation or upgrade process. Base images are currently distributed only by VMware and support x86 servers. VMware provides new versions of the base image for each upgrade, update, and patch release of the ESXi. Base images are hosted at the VMware online depot that is available by default to the vSphere Lifecycle Manager. Furthermore, you can download a different base image version, in the form of an offline ZIP bundle, from https://my.vmware.com/web/vmware/downloads.

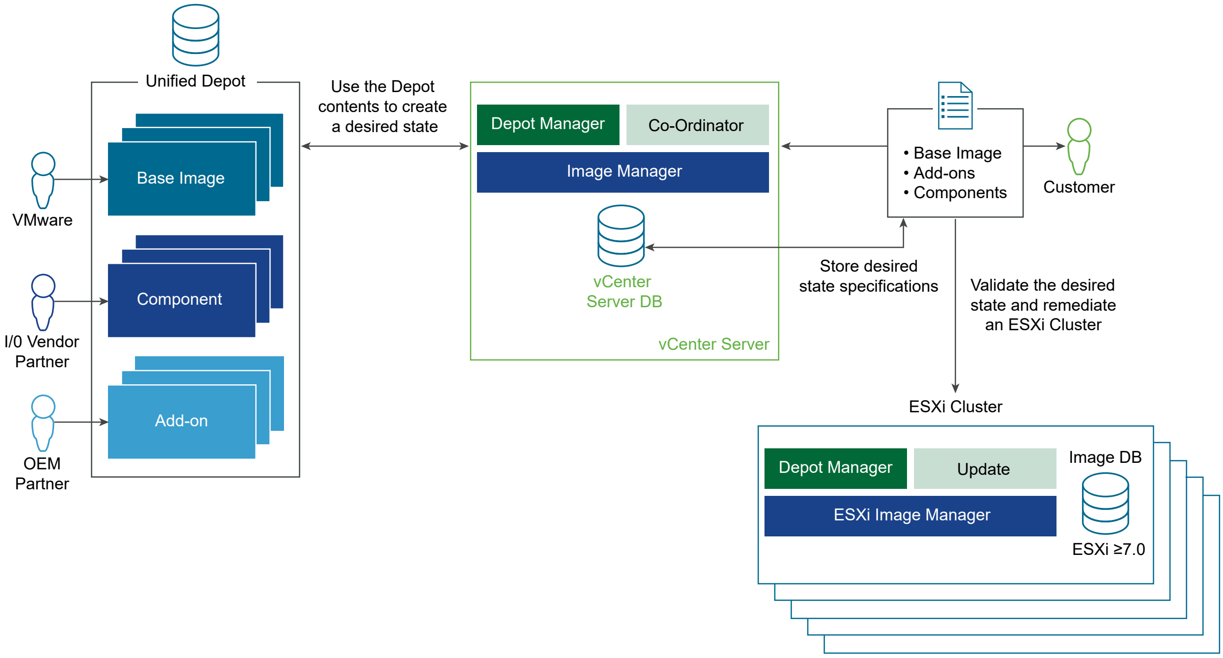
Term	Definition
Vendor add-on	An add-on is a collection of components that different OEMs provide on top of a base image. Vendors use an add-on to group some customizations for a family of servers. Unlike base images, add-ons are not complete and are not sufficient to boot an ESXi. Each add-on must have a unique name and version. An add-on allows vendors to add, remove, or update components that are part of the VMware base image, if there are no unresolved dependencies and conflicts between the components.
Solution	A solution contains one or more components, and provides information about its constraints and compatibility with the different ESXi versions. For example, from the perspective of the vSphere Lifecycle Manager solutions are VMware NSX, VMware vSphere® High Availability, vSAN.
Desired state	A desired state of a cluster is represented with a software specification. The desired state defines a set of components that a user wants to install on a single ESXi host or on a cluster of hosts.
OEMs	Original Equipment Manufacturers. VMware partners enrolled in the VMware Partner Connect application, such as Dell, Inc., HP Inc., Lenovo Group Ltd., and so on.
IOVP	I/O Vendor Partner. Qualified VMware partners providing certified I/O device drivers for network and storage host bus adapters.
Third-party software providers	Providers of I/O filters, device drivers, CIM modules, and so on, that are not part of VMware partner programs.
vSphere Configuration Profiles	With vSphere Configuration Profiles, you can manage the configuration of all hosts in a cluster collectively. This ensures consistency in host configuration at a cluster level. You can enable vSphere Configuration Profiles only on clusters that have the vSphere Lifecycle Manager enabled.

vSphere Lifecycle Manager Overview

The vSphere Lifecycle Manager feature provides means for managing the life cycle of hosts on a cluster level. The functionality of this feature is achieved through several major services that are running on both vCenter Server and ESXi.

After you install vSphere 8.0 Update 1, you can access the feature through several major services available on vCenter Server and on each ESXi host.

Figure 4-1. vSphere Lifecycle Manager System Architecture



Depot Manager

For each upgrade and patch release, VMware, OEMs, and other third-party companies make the software updates of their products available to the customers. Software updates are distributed to different locations and in different formats depending on the way they are accessed, downloaded, and used. Depot Manager allows these different software resources to be presented to the vSphere customers in a unified format and as a result, makes them easier to use.

Depending on your environment and specific use case, Depot Manager gives you access to the software updates within three different types of depots: online, offline, and UMDS. See Software Depots.

Depot Manager that runs on the vCenter Server instance achieves the following goals:

- Represents the contents of all depots in a unified way.
- Caches the payloads and their metadata locally prior to their use.
- Enables depot overrides on a cluster or host level for Remote Office/Branch Office (ROBO) environments, or Edge computing environments.

Depot Manager that runs on the hosts serves as a proxy to the vCenter Server Depot Manager. If ROBO cluster or ROBO host, or Edge computing environments are enabled, Depot Manager that runs on the ESXi hosts serves as a proxy to the nearby vSphere Lifecycle Manager compatible depot.

Image Manager

Image Manager allows you to create a desired state that you can apply on a cluster or a standalone host. The specification describes all components, add-ons, and the base image that you can use to update or upgrade the hosts in your environment. Image Manager supports validation of the desired state and the detecting drifts from the desired state.

Coordinator and ESXi Updater Managers

Coordinator Manager runs on the vCenter Server instance and makes sure that the desired state is applied to all hosts in the cluster. This module also runs pre-checks to evaluate how each host in the cluster is affected by the remediation and whether you must take some additional steps to ensure the success of the procedure. Coordinator Manager also allows you to query the status of the remediation operation for the cluster and for each host part of the cluster.

The ESXi Updater Manager takes care of the actual remediation happening on each host.

Options for Managing the ESXi Life Cycle

Based on your needs and environment setup, you can choose from several methods for managing the life cycle of the ESXi hosts. The vSphere Lifecycle Manager provides means for updating all hosts in a cluster or a standalone host with a desired software state.

To manage the life cycle of the hosts in your environment, you can use the vSphere Lifecycle Manager through the vSphere Client. See Managing Host and Cluster Lifecycle.

This chapter of the vSphere Automation REST API Programming Guide discusses how you can access and use the functionality provided by the vSphere Lifecycle Manager Automation API.

vSphere Lifecycle Manager Features

You can use the vSphere Automation API to manage the life cycle of all hosts in a cluster or of standalone hosts by using a vSphere Lifecycle Manager image. You can access and use the following vSphere Lifecycle Manager functionality:

- Depot management. You can add, remove, explore the contents of different types of depots. See Software Depots. The content of the depots is provided by VMware and VMware partners. Partners can use the ESXi Packaging Kit (EPK) to assemble a custom bootable ESXi image. The custom image can then be shared to other third-party customers and used through the Depot Manager service. For more information about how to create custom ESXi images, see *ESXi Packaging Kit (EPK) Development Guide*.
- Desired software state. You can create, edit, and delete a desired software state for a cluster or a standalone host on which the vSphere Lifecycle Manager is enabled. A desired software state must contain at least a single ESXi image provided by VMware. You can also set an add-on provided by OEMs, and one or more components by different software vendors. Furthermore, during the process of creating the desired software state, you can check the validity of the specification and compare the current state of the hosts in the cluster or the standalone host with the desired software state.

- Cluster remediation. You can apply the desired state on each of the hosts in a cluster which current state is different from the desired specification. Applying a desired state on a cluster level has the following prerequisites:
  - The cluster must have the vSphere Lifecycle Manager enabled.
  - All hosts in the cluster must store their data on a local or remote disk, or on a USB drive.
  - All hosts in the cluster must be of version 7.0 or higher.
  - All hosts must contain only components that the vSphere Lifecycle Manager can recognize and maintain. If a host contains some old content that the vSphere Lifecycle Manager does not recognize, the content is removed from the host during remediation.
- Standalone host remediation. You can apply the desired software state on a standalone host that is managed with baselines or on a standalone host that is managed with images but its current state differs from the desired state.

Required Privileges for Using vSphere Lifecycle Manager and vSphere Configuration Profiles

To configure vSphere Lifecycle Manager settings and to use successfully vSphere Lifecycle Manager baselines and images, you must have the proper privileges. Similarly, you need the proper privileges to use vSphere Configuration Profiles.

You can assign vSphere Lifecycle Manager and vSphere Configuration Profiles privileges to different roles in the vSphere Client.

vSphere Lifecycle Manager Privileges For Using Images

When you use vSphere Lifecycle Manager images, you need a different set of privileges for each task.

Table 4-1. VMware vSphere vSphere Lifecycle Manager Privileges For Using Images

Task	Privilege in the vSphere Client	Privilege in the API
Set Up Image	■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Read** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Write** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read**	■ `VcIntegrity.lifecycleSoftwareSpecification.Read` ■ `VcIntegrity.lifecycleSoftwareSpecification.Write` ■ `VcIntegrity.lifecycleSettings.Read`
Import Image	■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Read** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Write** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read** ■ **VMware vSphere Lifecycle Manager .Upload File.Upload upgrade images and offline bundles**	■ `VcIntegrity.lifecycleSoftwareSpecification.Read` ■ `VcIntegrity.lifecycleSoftwareSpecification.Write` ■ `VcIntegrity.lifecycleSettings.Read` ■ `VcIntegrity.FileUpload.com.vmware.vcIntegrity.ImportFile`
Export Image	**VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Read**	`VcIntegrity.lifecycleSoftwareSpecification.Read`
Edit Image	■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Read** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Write** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read**	■ `VcIntegrity.lifecycleSoftwareSpecification.Read` ■ `VcIntegrity.lifecycleSoftwareSpecification.Write` ■ `VcIntegrity.lifecycleSettings.Read`
Work with Recommendations	■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Read** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Write** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read**	■ `VcIntegrity.lifecycleSoftwareSpecification.Read` ■ `VcIntegrity.lifecycleSoftwareSpecification.Write` ■ `VcIntegrity.lifecycleSettings.Read`

Table 4-1. VMware vSphere vSphere Lifecycle Manager Privileges For Using Images (continued)

Task	Privilege in the vSphere Client	Privilege in the API
Work with Depot	- **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read** - **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Write**	- `VcIntegrity.lifecycleSoftwareSpecification.Read` - `VcIntegrity.lifecycleSoftwareSpecification.Write`
Manage Depot Overrides	- **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read** - **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Write**	- `VcIntegrity.lifecycleSoftwareSpecification.Read` - `VcIntegrity.lifecycleSoftwareSpecification.Write`
Check Compliance	**VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Read**	`VcIntegrity.lifecycleSoftwareSpecification.Read`
Run Remediation Pre-Check	- **VMware vSphere Lifecycle Manager.Lifecycle Manager: General Privileges.Read** - **VMware vSphere Lifecycle Manager.ESXi Health Perspectives.Read** - **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Remediation Privileges.Read**	- `VcIntegrity.lifecycleGeneral.Read` - `VcIntegrity.lifecycleHealth.Read` - `VcIntegrity.lifecycleSoftwareRemediation.Read`
Remediate Against an Image	- **VMware vSphere Lifecycle Manager.Lifecycle Manager: General Privileges.Read** - **VMware vSphere Lifecycle Manager.ESXi Health Perspectives.Read** - **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Remediation Privileges.Read** - **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Remediation Privileges.Write** - **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read**	- `VcIntegrity.lifecycleGeneral.Read` - `VcIntegrity.lifecycleHealth.Read` - `VcIntegrity.lifecycleSoftwareRemediation.Read` - `VcIntegrity.lifecycleSoftwareRemediation.Write` - `VcIntegrity.lifecycleSettings.Read`
Edit Remediation Settings	- **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read** - **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Write**	- `VcIntegrity.lifecycleSettings.Read` - `VcIntegrity.lifecycleSettings.Write`

Table 4-1. VMware vSphere vSphere Lifecycle Manager Privileges For Using Images (continued)

Task	Privilege in the vSphere Client	Privilege in the API
Update Firmware	■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Read** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Privileges.Write** ■ **VMware vSphere Lifecycle Manager.Lifecycle Manager: Settings Privileges.Read**	■ `VcIntegrity.lifecycleSoftwareSpecification.Read` ■ `VcIntegrity.lifecycleSoftwareSpecification.Write` ■ `VcIntegrity.lifecycleSettings.Write`
View Hardware Compatibility	**VMware vSphere Lifecycle Manager.Lifecycle Manager: Hardware Compatibility Privileges.Access Hardware Compatibility**	`VcIntegrity.HardwareCompatibility.Read`
Stage an Image	**VMware vSphere Lifecycle Manager.Lifecycle Manager: Image Remediation Privileges.Write**	`VcIntegrity.lifecycleSoftwareRemediation.Write`

For more information about managing users, groups, roles, and permissions, see the *vSphere Security* documentation.

Required Privileges for Using vSphere Configuration Profiles

When you use vSphere Configuration Profiles, you need a different set of privileges for each task.

Table 4-2. Required Privileges For Using vSphere Configuration Profiles

Task	Privilege in the vSphere Client	Privilege in the API	Description
Create Cluster with vSphere Configuration Profiles enabled	**Host.Inventory.CreateCluster**	`Host.Inventory.CreateCluster`	Create a cluster along with its initial configuration.
View Host Settings in the Cluster Configuration	**VMware vSphere Lifecycle Manager .Desired Configuration Management Privileges.Read-only access to desired configuration management platform**	`VcIntegrity.ClusterConfiguration.View`	View the draft configuration document, current configuration settings, compliance reports, and pre-check reports.
Check Cluster Compliance			
View Compliance Results			
Run Remediation Pre-Check			
View Remediation Pre-Check Results			
View Draft Pre-Check Results			
Remediate a Cluster Against the Desired Cluster Configuration	**VMware vSphere Lifecycle Manager .Desired**	`VcIntegrity.ClusterConfiguration.Remediate`	Remediate a cluster against the draft configuration or transition

Table 4-2. Required Privileges For Using vSphere Configuration Profiles (continued)

Task	Privilege in the vSphere Client	Privilege in the API	Description
Transition to vSphere Configuration Profiles	**Configuration Management Privileges.Remediate cluster to the desired configuration.**		a cluster to the desired configuration.
Export a Configuration or Configuration Schema	**VMware vSphere Lifecycle Manager .Desired Configuration Management Privileges.Export desired cluster configuration**	`VcIntegrity.ClusterConfiguration.Export`	Export the desired cluster configuration document.
Extract a Configuration Document from a Reference Host	**VMware vSphere Lifecycle Manager .Desired Configuration Management Privileges.Modify desired cluster configuration.**	`VcIntegrity.ClusterConfiguration.Modify`	Modify the desired cluster configuration by crating a draft configuration, extracting a configuration document from a reference host, or importing a configuration.
Create a Draft Configuration for a Cluster			
Import a Configuration			

Software Depots

A software depot represents a well-defined file structure used for storing and hosting the ESXi software updates, patches, and upgrades that VMware, partners, and third-party vendors provide. You can use the vSphere Automation APIs to manage the life cycle of the hosts in your environment by applying software updates hosted on different depots.

Software depots are managed by the Depot Manager which is part of the vSphere Lifecycle Manager. Software depots contain the actual payloads and the metadata of the software updates. Depending on the way you access the software updates, the Depot Manager recognizes three types of software depots: online, offline, and UMDS.

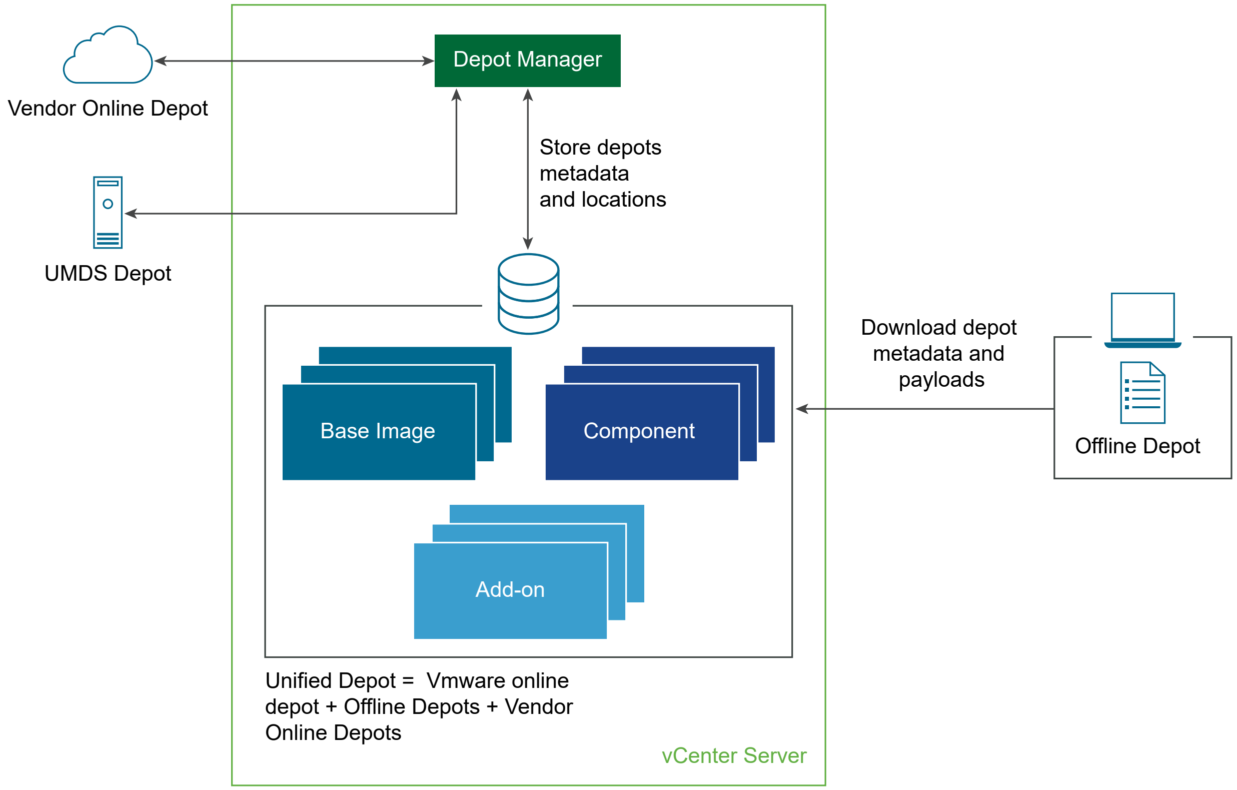
The following section of the documentation explains the concept of a software depot in terms of the vSphere Lifecycle Manager feature. You can also find common use cases available through the APIs for working with the different types of depots and their content.

Types of Software Depots

Depot Manager works with the software updates provided by three different types of software depots: online, offline, and UMDS.

Regardless of the different way in which you access each type of software depot, all depots have the same structure. The same depot structure allows content from different vendors to be uploaded to one depot. By default, you can access the content of the VMware online depot at `https://hostupdate.vmware.com/software/VUM/PRODUCTION/...`. Furthermore, partners and third-party customers can use the ESXi Packaging Kit to build and distribute software updates in the form of offline or online depots. You can access their content by adding the online vendor depot to the vSphere Lifecycle Manager or by downloading the content of the offline depot to the vCenter Server instance.

Figure 4-2. Types of Software Depots


Vendor Online Depot
Depot Manager
Store depots metadata and locations
UMDS Depot
Download depot metadata and payloads
Base Image
Component
Add-on
Offline Depot
Unified Depot = Vmware online depot + Offline Depots + Vendor Online Depots
vCenter Server

Online Depot

VMware and partners upload software updates to the VMware online depot at `https://hostupdate.vmware.com/software/VUM/PRODUCTION/...` or to a custom online depot. Software updates can be patches of the ESXi base image, different versions of the partner add-ons, the IOVP drivers certified by VMware, and the VMware Tools™ updates. Online depots are accessible through a URL. By default, you can see the base images, add-ons, and components provided within the VMware online depot at the following locations:

- `https://hostupdate.vmware.com/software/VUM/PRODUCTION/main/vmw-depot-index.xml`
- `https://hostupdate.vmware.com/software/VUM/PRODUCTION/addon-main/vmw-depot-index.xml`
- `https://hostupdate.vmware.com/software/VUM/PRODUCTION/iovp-main/vmw-depot-index.xml`

- `https://hostupdate.vmware.com/software/VUM/PRODUCTION/vmtools-main/vmw-depot-index.xml`

When you deploy the vCenter Server, the vSphere Lifecycle Manager is configured to access the VMware online depot, by default. You can use the vSphere Automation APIs to add a custom online depot to be managed by Depot Manager. The metadata of the newly added online depot is not synchronized immediately. To synchronize the metadata, you can run a synchronization operation or wait for the scheduled synchronization to take place.

The Depot Manager stores in the vCenter Server database only the metadata of the software updates and the location of the added online depots. You can create a schedule to synchronize the software updates metadata stored in the vCenter Server with the metadata available in the accessible depots. The payloads of the software updates are downloaded only during the cluster remediation process.

To add, remove, list, and retrieve information about the online depots, you can use the `Online` service. See Working with Online Depots.

Offline Depot

The offline depot is also called an offline bundle and is distributed as a downloadable ZIP file. Offline depots contain both the metadata and the payloads of the software update. Partners and third-party customers can use the ESXi Packaging Kit to build and distribute offline bundles. You can download offline bundles from the VMware website or from the websites of third-party vendors. When you add an offline depot to the vSphere Lifecycle Manager depot, the software updates are downloaded to the vCenter Server database.

To manage offline depots, you can use the `Offline` service. See Working with Offline Depots.

UMDS Depot

In case, the vCenter Server instance is in an air-gapped environment and has no access to any wire or wireless network, you can use a UMDS depot. The Update Manager Download Service (UMDS) is available as a `VMware-UMDS-8.0.1.-build_number.tar.gz` file within the ISO image of the vCenter Server appliance 8.0 UMDS is a 64-bit application and requires a 64-bit Linux-based system. Install UMDS on a machine that has Internet access and is different from the machine on which the vSphere Lifecycle Manager is running. For further information about how to install and configure the UMDS module, see the *Managing Host and Cluster Lifecycle*.

You can set up a synchronization schedule for downloading specific software updates from online vendor depots to the UMDS depot. Then use these updates to create desired software state for the clusters in your environment.

To manage UMDS depots through the vSphere Automation API, you can use the `UMDS` service.

Working with Online Depots

You can use the vSphere Automation APIs to add online depots to the list of currently configured online software depots.

Use online depots to add new content over time to the management scope of Depot Manager. Depot Manager periodically updates the software depots metadata stored on the vCenter Server instance. In case, new software updates are uploaded to the online depots, the Depot Manager makes sure that the metadata stored on the vCenter Server database is updated accordingly.

To add an online depot to the Depot Manager, you must first create the online depot specification by using the `Online.CreateSpec` data structure. To specify the URI to the `vendor-index.xml` file of the online depot, use the `location` property of the `Online.CreateSpec` data structure. Optionally, you can add a description and enable the depot. By default, when you add an online depot to the Depot Manager, the depot is enabled and its metadata is synchronized following the defined schedule. If you want to synchronize the added online depot immediately, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots?action=sync&vmw-task=true` request. When you complete the depot specification, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/online` request and submit the `Online.CreateSpec` JSON object to add the depot.

You can edit the depot description and deactivate the depot by using the `PATCH https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/online/`**`<depot_id>`** request and submitting the depot ID as path parameter and an `Online.UpdateSpec` JSON object in the request body.

You can remove an online depot from the list of currently configured depots by using the `DELETE https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/online/`**`<depot_id>`**`?vmw-task=true` request and submitting the depot ID as a pat parameter. This operation does not remove the already downloaded metadata and payloads from the deleted depot. You cannot delete the default VMware online depot, you can only deactivate it.

To retrieve a list of currently configured online depots, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/online` request. You can also retrieve information about a currently configured online depot by using the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/online/`**`<depot_id>`** request and submitting the depot ID as path parameter.

Working with UMDS Depots

In an air-gapped vCenter Server environment, you can use the vSphere Automation REST API to add a UMDS depot to the depots managed by Depot Manager.

After you install and configure the Update Manager Download Service (UMDS) on a physical machine with Internet access, you can add the UMDS depot to Depot Manager. Only one UMDS depot can be added at a time to Depot Manager. When you add a UMDS depot, its content is not immediately synchronized. To synchronize the content of the UMDS depots, you must use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots?action=sync&vmw-task=true` request or wait for the scheduled synchronization to take place.

To add a UMDS depot, use the `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/umds` request and submit a `Umds.SetSpec` JSON object in the request body.. The UMDS specification must contain the URI location to the `index.xml` file of the depot. Optionally, you can set a description and indicate whether the depot must be enabled. By enabling the UMDS depot, you instruct Depot Manager to synchronize only the content that is available on that depot.

You can always edit the initial UMDS depot settings, by using the `PATCH https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/umds` request and submit a `Umds.UpdateSpec` JSON object in the request body.

To retrieve information about the currently configured UMDS depot, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/umds` request. You can remove a currently configured UMDS depot and all its downloaded content from Depot Manager by using the `DELETE https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/umds?vmw-task=true` request.

Synchronizing Software Depots

The VMware online depot, the vendor online depots, and the UMDS depot must be synchronized regularly if you want to have the most recent software updates delivered by VMware, partners, and other third-party vendors. Use the vSphere Automation APIs to create a synchronization schedule or to synchronize the added depot immediately.

The Depot Manager does not synchronize immediately the metadata of the newly added online and UMDS depots. If you want to force the synchronization and not wait for the scheduled synchronization to take place, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots?action=sync&vmw-task=true` request. . You can also define a custom schedule to sync the metadata from the currently configured online or UMDS depots.

To create a custom schedule for checking for new software updates, you must first define the schedule parameters by using the `SyncSchedule.Schedule` data structure.. Then you can add the schedule to the schedule specification by using the `SyncSchedule.Spec` data structure.. Optionally, you can use the schedule specification to add an email to which notifications will be sent and define whether updates will be downloaded automatically. To apply the custom schedule, use the `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/sync-schedule` request and submit the `SyncSchedule.Spec` JSON object in the request body.

The default schedule is set to update the metadata daily at a random time. To reset the schedule to the default settings, use the `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/sync-schedule` and leave the `SyncSchedule.Spec` data structure unset..

Working with Offline Depots

An offline depot is a ZIP file that contains the metadata and payloads of software updates and follows the same structure as the online depot. Use the vSphere Automation APIs to import the content of an offline depot to the vCenter Server database.

To add an offline depot to the depots managed by the Depot Manager, you must create an offline depot specification by using the `Offline.CreateSpec` data structure. When you define the offline depot parameters, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depots/offline?vmw-task=true` request and submit the `Offline.CreateSpec` JSON object. Depending on the location of the offline depot, when you create the offline depot specification, you must provide either the URI location or the file ID returned by the Jetty Web server embedded in the vSphere Lifecycle Manager. You set the type of the source from which the offline depot is downloaded by using the `source_type` property of the `Offline.CreateSpec` data structure.

Pull Depot Content from a URI

To indicate that the offline depot resides on a URI location, set the depot location as a value to the `location` property of the `Offline.CreateSpec` data structure. You can use one of the following URI schemes: `http`, `https`, or `file`. If you provide an HTTPS location to the offline depot, make sure you also provide a certificate trusted by the VMware Certificate Authority (VMCA) or a custom certificate from the VMware Endpoint Certificate Store (VECS). For detailed information about how to manage certificates, see the *vSphere Authentication* documentation.

Push Depot Content to the Depot Manager

To push the content of an offline depot to the Depot Manager, you must first upload the ZIP file to the Jetty Web server at the `https://`*`<vcenter_FQDN>`*`:9087/vum-fileupload` URL. The server returns a file identifier that you canset as a value to the `file_id` property of the `Offline.CreateSpec` data structure.

Managing Depot Overrides

In case, you have a smaller Remote Office/ Branch Office (ROBO) cluster environment, or Edge computing environment, your clusters have no, or limited access to the Internet and limited connection to a vCenter Server instance. In such an environment, you can use Depot Manager to fetch the metadata and payloads of a desired software state from a local to the cluster depot.

To remediate a ROBO cluster, you must have access to a local software depot that hosts the components of the desired state. You can use Depot Manager to either export the whole vSphere Lifecycle Manager depot to an offline bundle, or export only the content of the desired state required for remediating the ROBO cluster.

To export the desired state image from the vSphere Lifecycle Manager depot, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_ID>`**`/software?action=export` request and submit the cluster ID as path parameter and an `Software.ExportSpec` JSON object. You receive the URI of the offline bundle that is hosted on the vSphere Lifecycle Manager Jetty Web server. To move the content of the offline bundle to the ROBO location, you must physically copy the ZIP file, unarchive, and mount its content to an HTTP server inside the ROBO environment.

To redirect a ROBO cluster to download software updates from a local repository and not from the vSphere Lifecycle Manager depot on the vCenter Server instance, the vSphere Automation REST API offer the cluster `DepotOverrides` service. To add a depot override location to a ROBO cluster, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_ID>`**`/depot-overrides?action=add` request. Specify the URI location of the local depot with the `location` property of the `DepotOverrides.Depot` data structure which you submit to the request body. During the ROBO cluster remediation, Depot Manager instructs the hosts in the ROBO cluster to download the software updates from the configured local depots within the ROBO cluster.

Inspecting Depot Contents

You can use the vSphere Automation REST API to inspect the contents of the already synchronized and imported depots. You can list the available base images, add-ons, and components, or retrieve some detailed information about a specific software update.

To retrieve a list of the base images available on a vCenter Server instance, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depot-content/base-images` request. You receive in the response body, a list of `BaseImages.Summary` JSON objects for each available base image. To get more information about a single base image, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depot-content/base-images/versions/`**`<version>`** request. You also get information about the components present in this base image.

To retrieve a list of all currently available add-ons in the vSphere Lifecycle Manager depot, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depot-content/add-ons` request and submit some of the query parameters to filter the list of retrieved add-ons. To retrieve some detailed information about an add-on, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depot-content/add-ons/`**`<name>`**`/versions/`**`<version>`** request and submit the name and version of the add-on.

To retrieve a list of all components currently available in the vSphere Lifecycle Manager depot, you can use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depot-content/components` request and submit some of the query parameters to filter the list of retrieved components. To retrieve detailed information about a component, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/depot-content/components/`**`<name>`**`/versions/`**`<version>`** request and submit the name and version of the component.

Enabling a Cluster to Use a Software Specification

If you want to use the vSphere Lifecycle Manager to manage the life cycle of clusters in your environment, you have two options for enabling this feature. You can enable the vSphere Lifecycle Manager when you create the cluster. You can also turn an already created cluster into one managed by the vSphere Lifecycle Manager.

Creating a Cluster with Enabled vSphere Lifecycle Manager

You can use only the vSphere Web Services API or the Virtual Infrastructure Management API to create a cluster in your virtual environment and specify the initial desired state of the cluster. Use the vSphere Automation REST API to specify the detailed desired state after the cluster creation.

Starting with vSphere 8.0 Update 1, you can use the Virtual Infrastructure Management API which introduced a new way of accessing and using the vSphere Web Services. That is, you can now use the JSON data format over HTTP and create REST-like requests to the vSphere Web Services. Use the Virtual Infrastructure API or the vSphere Web Services API to create a cluster. For more information about how to authenticate and use the Virtual Infrastructure Management API, see the *vSphere Web Services SDK Programming Guide*, *An Introduction to vSphere APIs*, and *Getting Started with vSphere APIs* documentation. You can also check the *Virtual Infrastructure JSON API* reference documentation at https://developer.vmware.com/apis/vi-json/latest/.

You can use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**` /sdk/vim25/`**`<release_number>`**`/Folder/`**`<moID>`**`/CreateClusterEx` request and submit the name of the cluster and a `ClusterConfigSpecEx` data object in the request body. In the data object, among other properties, you can specify the desired state for the cluster. The `desiredSoftwareSpec` property in the `ComputeResourceConfigSpec` data object contains the desired software specification for the cluster. This property is available for applications using the vSphere Web Services API of version 7.0 and later. You can create a `DesiredSoftwareSpec` data object and specify the base image that must be applied on the cluster with the `baseImageSpec` property. Optionally, you can specify a vendor add-on to be added to the software specification with the `vendorAddOnSpec` property.

Starting with vSphere 8.0, you can also enable the cluster to use vSphere Configuration Profiles by setting the `enableConfigManager` property of the `ComputeResourceConfigSpec` data object to **`true`**.

Enabling an Existing Cluster to Use vSphere Lifecycle Manager

If you want to manage the life cycle of a cluster by using a single software specification, you must first enable vSphere Lifecycle Manager on that cluster. You can use the vSphere Automation REST API to enable a cluster to use the vSphere Lifecycle Manager feature.

Before you enable vSphere Lifecycle Manager on a cluster, you can check whether the cluster meets all prerequisites. vSphere Lifecycle Manager can be enabled for a cluster only if the following requirements are met:

- All hosts in the cluster are of version 7.0 or later.
- All hosts in the cluster are stateful.
- All hosts in the cluster include only components that belong to integrated solutions, such as VMware vSAN™ and VMware vSphere® High Availability.
- None of the hosts in the cluster are in the process of active remediation through the VMware vSphere® Update Manager™.

- The cluster has a desired state already created for it.

If you want to run a preliminary check about whether all hosts in the cluster meet these requirements, use the `POST https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/<cluster_id>/enablement/software?action=check&vmw-task=true` request and submit the cluster ID and optionally a `Software.CheckSpec` JSON object.

The cluster ID represents the unique identifier for a cluster resource. To retrieve a list of clusters available on your vCenter Server instance, use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/cluster` request and filter the list of clusters by using the query parameters. You receive an array of `Cluster.Summary` JSON objects that contain the cluster ID among the other commonly used information.

You submit a `Software.CheckSpec` structure in the request body to specify which checks can be skipped during the cluster preliminary check. Though you can skip some checks with this operation, the Image Manager runs all checks during the enablement operation. If you leave the check specification empty, all checks are run for each host in the cluster. You can select among the following checks to be skipped when running a pre-check operation:

- `SOFTWARE`. Checks whether there are any orphaned vSphere Installation Bundles (VIBs) and any software that cannot co-exist with vSphere Lifecycle Manager.
- `VERSION`. Checks whether all hosts in the cluster are of version greater than a predefined one.
- `STATELESSNESS`. Checks whether there are any stateless hosts in the cluster. vSphere Lifecycle Manager can be enabled only if the cluster does not contain stateless hosts.
- `VUM_REMEDIATION`. Checks whether any of the hosts in the cluster are currently remediated through the VMware vSphere® Update Manager™.
- `SOFTWARE_SPECIFICATION_EXISTENCE`. Checks whether there is a software specification already associated with this cluster. In case, this check reports that the cluster does not have a software specification, you must first create a draft software specification for this cluster and then commit the draft.
- `VSAN_WITNESS_ELIGIBILITY`. Checks whether the software specification can be used on any vSAN witness hosts in the cluster. For information about how you can manage a vSAN cluster by using vSphere Lifecycle Manager, see *vSAN Clusters and vSphere Lifecycle Manager* chapter in the *Managing Host and Cluster Lifecycle* documentation.

To enable a cluster to be managed with vSphere Lifecycle Manager, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/`**`<cluster_id>`**`/enablement/software?vmw-task=true` request and submit the cluster ID as path parameter. To specify which check can be skipped during the enablement process, submit a `Software.EnableSpec` JSON object. Currently, you can only skip the software checks.

You can also get information about which clusters in your environment are managed with a single software specification. Use the `GET https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/`**`<cluster_id>`**`/enablement/software` request and submit the cluster ID as a path parameter.

Enabling a Standalone Host to Use a Software Specification

Starting with vSphere 8.0, you can use the vSphere Lifecycle Manager to manage the life cycle of standalone hosts in your vCenter Server system.

You can add a standalone host to a vCenter Server instance under a data center object or into a folder. See Chapter 3 ESXi Hosts. If you remove a host from a cluster but leave it in the data center, the host is also considered a standalone host.

To manage a standalone host with a software specification, you must first enable the vSphere Lifecycle Manager on that host. You can run a preliminary check to establish whether the standalone host meets all requirements for enabling the vSphere Lifecycle Manager. Use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/enablement/software?action=check&vmw-task=true` request and submit the host ID as path parameter. The pre-checks that you can skip before enabling the vSphere Lifecycle Manager on a standalone host are the same as for a cluster.

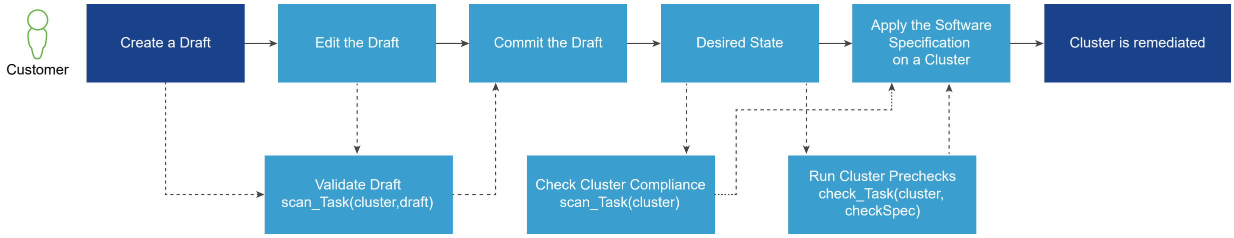
To enable vSphere Lifecycle Manager on a standalone host, use the `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/enablement/software?vmw-task=true` request and submit the host ID as path parameter. Optionally, to define whether you want to skip any checks during the feature enablement, you can add a `Software.EnableSpec` data structure to the request body. Currently, you can skip only the check for orphaned vSphere Installation Bundles (VIBs).

Working with Draft Software Specifications

You create a draft software specification to describe the components of the desired state that you want to apply on a cluster or a standalone host.

The draft software specification is the working copy of the desired software state. Only one user at a time is allowed to edit a single draft for a cluster or a standalone host. Before saving the changes to the edited draft version, you can validate the content of the draft.

Figure 4-3. Drafts Workflow for a Cluster


Customer
Create a Draft
Edit the Draft
Commit the Draft
Desired State
Apply the Software Specification on a Cluster
Cluster is remediated
Validate Draft scan_Task(cluster,draft)
Check Cluster Compliance scan_Task(cluster)
Run Cluster Prechecks check_Task(cluster, checkSpec)

A typical workflow for working with draft software specifications starts with creating a draft software specification for a specific cluster or a standalone host. If the cluster or the standalone host already have a software specification defined, this action takes the latest committed draft and you can edit its contents according to your needs. If the cluster or the standalone host have no software specification created yet, this method creates an empty draft. The only mandatory item for a draft is a base image of a specific version.

After you complete adding components, an add-on, and a base image, you can save your changes by committing the draft. This operation results in setting the committed draft as the current desired state of the cluster or the standalone host. Before committing the draft software specification, you can validate the contents of the draft or check whether all hosts in the cluster or the single standalone host are compliant with the draft.

If the commit operation is successful, the draft becomes the desired state for the cluster or the standalone host. You can now export the software specification and use it, for example, in a ROBO cluster scenario or another standalone host. You can also validate the compliance of the hosts in the cluster or the standalone host against the desired state and then apply the software specification, if feasible.

Creating a Draft Software Specification

To describe the components of a desired state for a cluster or a standalone host, create a draft software specification and save the desired state when ready.

To edit an existing desired state or to create an empty draft software specification:

- For a cluster, use the `POST https://`*`<vcenter_ip_address_or_fqdn>`*`/api/esx/settings/clusters/`*`<cluster_id>`*`/software/drafts` request.
- For a standalone host, use the `POST https://`*`<vcenter_ip_address_or_fqdn>`*`/api/esx/settings/hosts/`*`<host_id>`*`/software/drafts` request.

Submit the cluster ID or the host ID as path parameters to the respective request. As a result, you receive a draft ID which you can use to add a base image, an add-on, or some components to the draft.

Editing a Draft Software Specification

After you have created a draft software specification, use the vSphere Automation REST API to edit its items.

To set a base image to a draft software specification:

- For a cluster, use the `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/software/drafts/`**`<draft_id>`**`/software/base-image` request and submit the cluster and draft IDs as path parameters and a base image specification in the request body.

- For a standalone host, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/esx/settings/hosts/<host_id>/software/drafts/<draft_id>/software/base-image` request and submit the host and draft IDs as path parameters and a base image specification in the request body.

If the draft contains a base image, this method overwrites the existing image. The base image specification contains the version of the bootable ESXi that must be included in the desired state. To retrieve details about the base image that is currently present in a draft, use the `GET https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/drafts/<draft_id>/software/base-image` or `GET https://<vcenter_ip_address_or_fqdn>/api/esx/settings/hosts/<host_id>/software/drafts/<draft_id>/software/base-image` requests and submit as path parameters the cluster or the standalone host ID, and the draft ID. You receive a `BaseImageInfo` JSON object in the request body that holds the details about the version, display name and version, and the release date of the ESXi host.

To add an OEM add-on to a draft software specification:

- For a cluster, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/drafts/<draft_id>/software/add-on` request.
- For a standalone host, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/esx/settings/hosts/<host_id>/software/drafts/<draft_id>/software/add-on` request.

Pass as arguments to these methods the cluster or the standalone host ID, the draft ID, and the add-on specification. If you want to remove an add-on from a draft, use the `DELETE https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/drafts/<draft_id>/software/add-on` and `DELETE https://<vcenter_ip_address_or_fqdn>/api/esx/settings/hosts/<host_id>/software/drafts/<draft_id>/software/add-on` and submit as path parameters the cluster or the standalone host ID, and the draft ID.

You can add a component, change the version, or delete an existing component from a draft software specification. To change the version of a component included in a draft:

- For a cluster, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/drafts/<draft_id>/software/components/<component_id>` request.
- For a standalone host, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/esx/settings/hosts/<host_id>/software/drafts/<draft_id>/software/components/<component_id>` request.

As a result, you add the component specified with the `component_ID` and `version` arguments to the draft, if it is missing. To remove a component from a draft, use the `DELETE https://<vcenter_ip_address_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/drafts/<draft_id>/software/components/<component_id>` and `DELETE https://<vcenter_ip_address_or_fqdn>/api/esx/settings/hosts/<host_id>/software/`

drafts/**<draft_id>**/software/components/**<component_id>** requests. You can change multiple components in a draft by using the PATCH https://**<vcenter_ip_address_or_fqdn>**/api/esx/settings/clusters/**<cluster_id>**/software/drafts/**<draft_id>**/software/components and PATCH https://**<vcenter_ip_address_or_fqdn>**/api/esx/settings/hosts/**<host_id>**/software/drafts/**<draft_id>**/software/components requests. To specify the components you want to remove, add, or update for a given draft, submit a cluster or a host Components.UpdateSpec JSON object to the request body of the respective request.

To retrieve information about all components present in a draft, or a single component, use the list requests of the respective Components service.

Validating the Draft Software Specification

Before saving a draft and turning it into a desired state for a cluster or a standalone host, you can check whether the specification is complete and valid. You can also check whether the draft specification drifts in any way from the current state of the cluster or the host.

The Drafts service offers two methods for validating the draft software specification. To check whether a draft is complete and there are no conflicts between the draft components, or unresolved dependencies, use the POST https://**<vcenter_ip_address_or_fqdn>**/api/esx/settings/clusters/**<cluster_id>**/software/drafts/**<draft_id>**?action=validate&vmw-task=true or POST https://**<vcenter_ip_address_or_fqdn>**/api/esx/settings/hosts/**<host_id>**/software/drafts/**<draft_id>**?action=validate&vmw-task=true request depending on the targeted vSphere inventory object. Submit as path parameters the cluster ID or the standalone host ID, and the draft ID. You validate whether there were any other drafts committed for this cluster or standalone host, which can make the current commit operation invalid. The method also validates whether all components defined in the software specification are available in the depot metadata. This method does not run compliance checks against the cluster or the standalone host.

To check whether all hosts in the cluster or the standalone host are compliant with the draft software specification, use the POST https://**<vcenter_ip_address_or_fqdn>**/api/esx/settings/clusters/**<cluster_id>**/software/drafts/**<draft_id>**?action=scan&vmw-task=truePOST https://**<vcenter_ip_address_or_fqdn>**/api/esx/settings/hosts/**<host_id>**/software/drafts/**<draft_id>**?action=scan&vmw-task=true request depending on the targeted vSphere inventory object. Submit as path parameters the cluster ID or the standalone host ID, and the draft ID. This method results in running a comparison between the draft specification and the current state of each host in the cluster or the current state of the standalone host.

Committing the Draft Software Specification

When you commit a draft software specification, it becomes the desired state for the cluster or the standalone host.

To save the draft that you created for a cluster or a standalone host, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/software/drafts/`**`<draft_id>`**`?action=commit&vmw-task=true` and `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/software/drafts/`**`<draft_id>`**`?action=commit&vmw-task=true` request depending on the targeted vSphere inventory object. The Image Manager component runs a validation check before the draft gets saved to the database. This method returns an identifier of the commit operation. You can use the ID to retrieve information about a specific commit such as the author of the commit operation, the time when the draft was committed, and so on.

Working with Desired Software States

When you commit a draft software specification, you make the committed draft the desired state for that cluster or standalone host. If all hosts in the cluster are compliant with the desired state, you can remediate the cluster. If the single standalone host is compliant with the desired state, you can update the ESXi host to that state.

You can use the methods provided with the cluster and host `Software` services to manage a desired state for a cluster or a standalone host. Before you apply a desired state on a cluster or a standalone host, you can run pre-checks to ensure that all hosts in the cluster or the single host are in a good state to be remediated. The pre-checks verify whether any of the hosts in the cluster or the single standalone host must be rebooted or are in maintenance mode. You can also check the compliance of the cluster or the standalone host against the desired state. See Checking the Compliance Against the Desired State.

You can export a software specification created for a cluster or a standalone host by using one of the following formats:

- An offline bundle in a ZIP file format.
- An ISO image.
- A JSON file.

Use the vSphere Lifecycle Manager API to import a software specification as a draft and then edit it. You have several options for running the import operation depending on the location and format of the desired software state.

Exporting and Importing a Desired State

Use the vSphere Automation REST API to export the desired software state of a cluster or a standalone host. Then you can import the desired state to a different cluster or host in the same or a different vCenter Server instance.

Exporting a Desired State

To export a desired state, you can use one of the following methods:

- For a cluster, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/software?action=export` request.
- For a standalone host, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/software?action=export` request.

This method does not export any information about the solutions available on the cluster or the standalone host since the constraints set by these solutions might not be applicable for another cluster or host. Submit the cluster or the standalone host ID, and a `Software.ExportSpec` JSON object in the request body.

You can choose how to export the desired software specification.

- Export as an ISO image. Set **`true`** to the `export_iso_image` property of the `Software.ExportSpec` JSON object that you submit in the request body. Use the exported ISO image for performing clean installs and for bootstrapping purposes. You can upload the ISO file into the Jetty Web server on the target vCenter Server instance but you cannot use ISO files to manage the life cycle of clusters or a standalone host through the vSphere Lifecycle Manager feature.
- Export as an offline bundle in a ZIP file format. Set **`true`** to the `export_offline_bundle` property of the `Software.ExportSpec` JSON object that you submit in the request body. You can use the exported offline bundle to create a depot and add its components to the resources managed by the Depot Manager module.
- Export as a JSON file holding the desired state specification. Set **`true`** to the `export_software_spec` property of the `Software.ExportSpec` JSON object that you submit to the request body. You can then reuse the JSON file to apply the desired state that it contains to another cluster or standalone host in the same or in a different vCenter Server instance. Note that the JSON file holds only the description of the desired state. You must check whether all components described in the JSON file are available in the depot for the target cluster. See Importing a Desired State Specification for information about how you can use a desired state specification for another cluster or standalone host.

Importing a Desired State Specification

To import a desired state of a cluster or a standalone host and assign it to another cluster or standalone host in the same or different vCenter Server instance:

- For a cluster, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/software/drafts?action=import-software-spec` request.
- For a standalone host, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/software/drafts?action=import-software-spec` request.

Submit as pat parameters the cluster or the standalone host ID, and a `Drafts.ImportSpec` JSON object in the request body. Use the instance of the import specification to describe the download source and the source type of the imported software specification. Depending on the location and the source type of the exported desired state, you can choose from the following import options:

- Import a file from the vCenter Server or your local file system. Use the `file_id` property of the `Drafts.ImportSpec` data structure to set the file ID of the software specification which was previously uploaded on the Jetty Web server running on the vCenter Server at `https://<vcenter_FQDN>:9087/vum-fileupload` URL. You can also use this option to import a specification file that resides on your local file system. Make sure you set the source type of the import specification to `PUSH` through the `source_type` property of the `Drafts.ImportSpec` data structure.
- Import a file that resides on a URI location. Use the `location` property of the `Drafts.ImportSpec` data structure to submit the URI location of the software specification file. The software specification can be pulled from a URI location with one of the following schemes: `file`, `http`, or `https`. You can use this import mechanism only if you set the `source_type` property to `PULL`.
- Import a desired state as a JSON string. Use the `software_spec` property of the `Drafts.ImportSpec` data structure to set the JSON string representing the software specification you want to import. Use this method only if you set the `source_type` property to `JSON_STRING`.

Checking the Compliance Against the Desired State

Before applying a desired state on a cluster, you can scan all hosts in the cluster against the desired state and check the cluster compliance against the desired state. Before applying a desired state on a standalone host, you can scan the host and check its compliance against the desired state.

To check the compliance of all hosts in a cluster or of the single standalone host:

- For a cluster, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/software?action=scan&vmw-task=true` request.
- For a standalone host, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/software?action=scan&vmw-task=true`request.

Pass as an parameter the cluster or the host identifier. The method compares the desired state against the current state of each host in the cluster or of the single host and as a result calculates the compliance.

You can retrieve the cluster or the standalone host compliance status by using the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/software/compliance` or `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/software/compliance` requests and passing the cluster or standalone host ID as path parameter. As a result, you receive a `ClusterComliance` or `HostCompliance` JSON object that contains the following information:

- The overall cluster compliance status regarding the target version of the components described within the cluster desired state.
- The impact of applying the desired state on the cluster in case the cluster is non-compliant.
- A list of all compliant hosts in the cluster.
- A list of the incompatible hosts in the cluster.
- A list of the non-compliant hosts in the cluster.
- A list of the unavailable hosts which cannot be checked for compliance against the desired state.
- The compliance status of each host in the cluster.
- The notifications returned by the compliance check operation.
- The time that the compliance check takes.
- The ID of the committed draft for which the compliance check is performed. If the `commit` property of the cluster or host compliance data structure is unset, the compliance check is run against a draft software specification.
- The compliance overall stage status of the cluster or the standalone host and is relevant when the compliance status of the cluster or the standalone host is `NON_COMPLIANT`.
- The compliance of the OEM add-on on the standalone host with respect to the add-on in the desired state.
- The compliance of the base image on the standalone host with respect to the base image in the desired state.
- The compliance of all effective components and all components present on the standalone host in respect to the components in the desired state.
- The compliance of all DPU devices on the standalone host with respect to the desired state.
- The compliance of the hardware support on the standalone host with respect to the hardware support defined in the target state.

A cluster or a standalone host can have one of the following compliance statuses regarding the target versions:

- `COMPLIANT`. The target versions of the components described in the desired state of the cluster or the standalone host are the same as the versions of the components currently present on the hosts in the cluster or on the standalone host.

- `NON_COMPLIANT`. The desired state of the cluster or the standalone host describes components with higher versions than the versions of the components currently present on the hosts in the cluster or on the standalone host. Non-compliant clusters or hosts are those clusters or hosts which have orphaned VIBs, or components on the hosts that are not present in the desired state specification.
- `INCOMPATIBLE`. One or more hosts in the cluster or in the standalone host have components with higher versions than the components described in the desired state specification.
- `UNAVAILABLE`. The current state of one or more hosts in the cluster or of the standalone host cannot be retrieved and as a result the compliance check cannot be performed.

You can check the compliance impact of applying the desired state on a non-compliant cluster or a standalone host by using the `impact` property of the respective compliance data structure which contains the `ComplianceImpact` data structure. Use it to retrieve information about the steps you must take to remediate the cluster or the standalone host successfully. You might need to reboot a host or put a host into maintenance mode to remediate the cluster or the standalone host successfully.

Hardware Compatibility Data

The hardware compatibility data contains information about the compatibility between ESXi hosts and ESXi versions.

You can use the `CompatibilityData` service to retrieve information about the compatibility data or to update the local compatibility data on a vCenter Server instance. To retrieve information about the compatibility data stored on your vCenter Server instance, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/hcl/compatibility-data/status` request. To update the local compatibility data with the latest version available from the official VMware source, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/hcl/compatibility-data?action=download&vmw-task=true` request.

Checking the Hardware Compatibility of an ESXi Host

You can query the hardware compatibility for a host before upgrading to a new ESXi version. You can also download the information generated by the hardware compatibility report.

To use services form the `esx hcl` package, you must verify that you have accepted to participate in the CEIP and there is available compatibility data.

You can use the `CompatibilityReleases` service to list available releases for generating a compatibility report for a specific ESXi host. To list the locally available ESXi releases for the host that can be used to generate a compatibility report,use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/hcl/hosts/`**`<host_id>`**`/compatibility-releases` request. The list includes only major and update releases. Patch releases are not listed.

You can use the `CompatibilityReport` service to generate a hardware compatibility report for an ESXi host against a specific ESXi release. To return the last generated hardware compatibility report for a specific host, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/hcl/hosts/`**`<host_id>`**`/compatibility-report` request. To generate a hardware compatibility report for a specific host against specific ESXi release, use the `POST https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/hcl/hosts/`**`<host_id>`**`/compatibility-report?vmw-task=true` request.

You can use the `Reports` service to download information generated by the hardware compatibility report. To retrieve the URI location for downloading a compatibility report, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/hcl/reports/`**`<report_id>`** request.

Configuring Remediation Settings

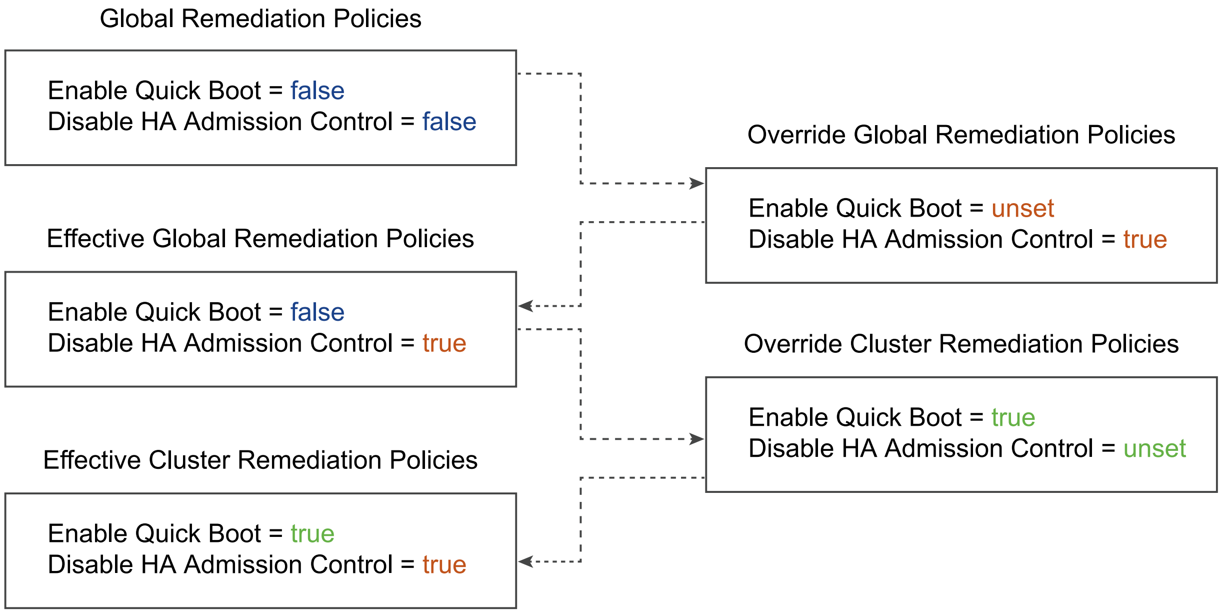
You can control the behavior of the ESXi hosts and virtual machines during the remediation process. You can create a global remediation policy that applies to all clusters and standalone hosts in a vCenter Server instance. You can also set a remediation policy to a specific cluster and a standalone host.

When you run cluster and standalone host compliance checks, the Coordinator module runs a series of checks on each host to determine their state and whether some additional actions must be taken to ensure the success of the remediation operation. In case one or more hosts in the cluster and any of the standalone hosts are evaluated as non-compliant, additional checks are run on those hosts to evaluate whether they must be rebooted or put into maintenance mode. Currently, VMware provides a set of behavior controls (remediation policies) regarding the virtual machines and the hosts in a cluster or a standalone host. This set of remediation policies might change with the next vSphere release.

How Remediation Policies Overrides Work

The vSphere Lifecycle Manager provides a default global policy configuration that must be applied on each cluster and standalone host during remediation. Through the vSphere Automation REST API, you can change the global policies and create some cluster-and host-specific policies. Before remediating a cluster and standalone host, you can use the API to determine the effective global and cluster-and host-specific remediation policies. The following graphic describes how the mechanism of the policy overrides works.

Figure 4-4. How Remediation Policies Work


Global Remediation Policies
Enable Quick Boot = false
Disable HA Admission Control = false
Override Global Remediation Policies
Enable Quick Boot = unset
Disable HA Admission Control = true
Effective Global Remediation Policies
Enable Quick Boot = false
Disable HA Admission Control = true
Override Cluster Remediation Policies
Enable Quick Boot = true
Disable HA Admission Control = unset
Effective Cluster Remediation Policies
Enable Quick Boot = true
Disable HA Admission Control = true

All clusters and standalone hosts in a vCenter Server instance inherit the default or the overridden global policy settings unless the global policy is explicitly overridden on a cluster and host level.

Editing Global or Cluster- and Host-Specific Remediation Policies

To view the currently set global remediation policy, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/defaults/clusters/policies/apply` or `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/defaults/hosts/policies/apply` request. You receive a `Apply.ConfiguredPolicySpec` data structure that contains the settings of the global remediation policy. To edit a global remediation policy, use the `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/defaults/clusters/policies/apply` or `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/defaults/hosts/policies/apply` request and submit the respective `Apply.ConfiguredPolicySpec` data structure to define new values to the global policy settings. To view the effective global remediation policy settings for a cluster and host, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/defaults/clusters/policies/apply/effective` or `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/defaults/hosts/policies/apply/effective` request. You receive the respective `Effective.EffectivePolicySpec` JSON object that contains the effective global policies applicable for all clusters and hosts in your vCenter Server environment.

To view the cluster- and host-specific remediation policies, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/policies/apply` or `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/policies/apply` request. You receive an `Apply.ConfiguredPolicySpec` JSON object that contain the cluster- and host-specific policies to be applied during remediation. To change the cluster- and host-specific policy, use the `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/policies/apply` or `PUT https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/policies/apply` request and submit an`Apply.ConfiguredPolicySpec` data structure to describe the cluster- and host-specific remediation policies. To view the effective cluster- and host-specific policies, use the `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/clusters/`**`<cluster_id>`**`/policies/apply/effective` or `GET https://`**`<vcenter_ip_address_or_fqdn>`**`/api/esx/settings/hosts/`**`<host_id>`**`/policies/apply/effective` request. You receive the respective `Effective.EffectivePolicySpec` data structure that describes the effective cluster- and host-specific policies.

Remediation Policy Options for Clusters

To describe a global or cluster-specific remediation policy, use the respective cluster `Apply.ConfiguredPolicySpec` data structure. For the vSphere 8.0 release, VMware provides the following methods to configure a global or cluster-specific policy.

Property	Description
`disable_dpm`	Deactivate the VMware Distributed Power Management (DPM) feature for all clusters or for a specific cluster. DPM monitors the resource consumption of the virtual machines in a cluster. If the total available resource capacity of the hosts in a cluster is exceeded, DPM powers off (or recommends powering off) one or more hosts after migrating their virtual machines. When resources are considered underutilized and capacity is needed, DPM powers on (or recommends powering on) hosts. Virtual machines are migrated back to these hosts. During the cluster remediation, the vSphere Lifecycle Manager cannot wake up and remediate hosts that are automatically put into a stand-by mode by DPM. These hosts stay non-compliant when DPM turns them on. The vSphere Distributed Resource Scheduler (DRS) is unable to migrate virtual machines to the hosts which are not remediated with the desired state for the cluster. To deactivate DPM during the cluster remediation, set **`true`** to the `disable_dpm` property. By default, the vSphere Lifecycle Manager temporarily deactivates DPM and turns on the hosts to complete the remediation. DPM is enabled again when the cluster remediation finishes.
`disable_hac`	Deactivate the vSphere HA admission control. vSphere HA uses admission control to ensure that a cluster has sufficient resources to guarantee the virtual machines recovery when a host fails. If vSphere HA admission control is enabled during remediation, putting a cluster into maintenance mode fails because vMotion cannot migrate virtual machines within the cluster for capacity reasons. To allow the vSphere Lifecycle Manager to temporary deactivate vSphere HA admission control, set **`true`** to the `disable_hac` property. By default, the vSphere HA admission control is enabled because DRS should be able to detect issues with the admission control and deactivate it to allow the remediation to complete.
`evacuate_offline_vms`	Migrate the suspended and powered off virtual machines from the hosts that must enter maintenance mode to other hosts in the cluster. To enable this remediation policy, set **`true`** to the `evacuate_offline_vms` property. By default, this setting is deactivated in the global remediation policy.

Property	Description
`failure_action`	Specify what actions vSphere Lifecycle Manager must take if a host fails to enter maintenance mode during the remediation. To configure this policy on a global or cluster-specific level, set the properties of the `Apply.FailureAction` data structure. You can set the number of times that vSphere Lifecycle Manager tries to put a host into maintenance mode and the delay between the tries. When the threshold is reached and the host failed to enter maintenance mode, the cluster remediation fails. By default, vSphere Lifecycle Manager tries to put a host into maintenance mode three times with a five minute delay between each try before the cluster remediation fails.
`enforce_hcl_validation`	Prevents the remediation of vSAN clusters if vSphere Lifecycle Manager reports hardware compatibility issues during the hardware compatibility check performed as part of the remediation pre-check or the remediation tasks of the cluster.If you leave the `enforce_hcl_validation` property unset, detected hardware issues are reported as warnings and do not prevent the remediation of the vSAN cluster.
`parallel_remediation_action`	Enable simultaneous remediation of all hosts that are in maintenance mode with in the cluster. Submit the `Apply.ParallelRemediationAction` data structure to indicate the maximum number of hosts that can be remediated in parallel. **Note** If the hosts have NSX virtual distributed switches that are ready to be migrated to vSphere Distributed Switches, you must manually set the maximum number of parallel remediations to no more than 4. In cases when host switch migration is needed, if more than 4 hosts are remediated in parallel, the remediation might fail, because the host switch migration takes more time than the time vSphere Lifecycle Manager needs to complete the parallel remediation.
`pre_remediation_power_action`	Specify how the power state of the virtual machines must change before the host enters maintenance mode. If DRS is not enabled on a cluster or the automation level of a DRS cluster is not set to fully automated, the Coordinator module fails to remediate the cluster if the remediation requires a reboot or maintenance mode. You can set a policy that powers off or suspends the virtual machines on hosts that must be rebooted or must enter maintenance mode during remediation. The DRS takes care of changing the power state of the virtual machines when the host enters and exits maintenance mode. To set a policy for the power state of the virtual machines during the remediation, submit the `Apply.ConfiguredPolicySpec.PreRemediationPowerAction` enumerated type and set one of the following values: ■ `DO_NOT_CHANGE_VMS_POWER_STATE`. Indicates that the power state of the virtual machines must not be changes. ■ `POWER_OFF_VMS`. Indicates that the virtual machines must be powered off before the hosts enter maintenance mode. ■ `SUSPEND_VMS`. Indicates that the virtual machines must be suspended before the hosts enter maintenance mode. ■ `SUSPEND_VMS_TO_MEMORY`. Indicates that the virtual machines must be suspended before the hosts enter maintenance mode. By default, the Coordinator must leave the power state of the virtual machines unchanged.
`enable_quick_boot`	Reduce the reboot time of an ESXi host by skipping all the hardware initialization processes and restarting only the hypervisor. This policy is applicable only if the host platform supports the Quick Boot feature. To enable the Quick Boot feature on the hosts during the remediation, set **`true`** to the `enable_quick_boot` property. By default, this policy is deactivated.

Remediation Policy Options for Standalone Hosts

To describe a global or cluster-specific remediation policy, use the respective host Apply.ConfiguredPolicySpec data structure. For the vSphere 8.0 release, VMware provides the following methods to configure a global or host-specific policy.

Property	Description
enable_quick_boot	Optimize the host patching and upgrade operations by reducing the reboot time of an ESXi host. Since the patching and upgrading operations do not affect the hardware of the host, the hardware initialization processes can be skipped. This policy is applicable only if the host platform supports the Quick Boot feature. For more information about which hosts are Quick Boot compatible, see the following KB article https://kb.vmware.com/s/article/52477.
pre_remediation_power_action	Specify how the power state of the virtual machines must change before the standalone host enters maintenance mode. You can choose between the following power actions: ■ DO_NOT_CHANGE_VMS_POWER_STATE. Indicates that the power state of the virtual machines must not be changed. ■ POWER_OFF_VMS. Indicates that the virtual machines must be powered off before the standalone host enters maintenance mode. ■ SUSPEND_VMS. Indicates that the virtual machines must be suspended before the standalone host enters maintenance mode. ■ SUSPEND_VMS_TO_MEMORY. Indicates that the virtual machines must be suspended before the standalone host enters maintenance mode.
failure_action	Specify what actions vSphere Lifecycle Manager must take if a standalone host fails to enter maintenance mode during the remediation. You can set the number of times that vSphere Lifecycle Manager tries to put a standalone host into maintenance mode and the delay between the tries. When the threshold is reached and the standalone host failed to enter maintenance mode, the host remediation fails.

Remediating an ESXi Cluster and a Standalone Host

You can use the vSphere Automation API to initiate the remediation of a cluster or a standalone host through the vSphere Lifecycle Manager.

To remediate a cluster with a desired state, use the POST https://**<vcenter_ip_address_or_fqdn>**api/esx/settings/clusters/**<cluster_id>**/software?action=apply&vmw-task=true request and submit a Software.ApplySpec JSON object in the request body. Specify whether the VMware End User License Agreement (EULA) must be accepted. You can also set the minimum commit ID of the draft software specification that must be used for remediating the cluster. Upon successful completion of the remediation task, all hosts in the cluster have the same software state.

To remediate a standalone host with a desired state, use the POST https://**<vcenter_ip_address_or_fqdn>**/api/esx/settings/hosts/**<host_id>**/software?action=apply&vmw-task=true request and submit a Software.ApplySpec JSON object in the request body. Use the software apply specification to set the minimum commit identifier of the draft software specification. See Committing the Draft Software Specification. You can also specify whether the VMware End User License Agreement (EULA) must be accepted.

Integrate Third-Party Solutions with vSphere Lifecycle Manager

A solution is an ESXi software package that extends the functionality and capabilities of a host and integrates with the vCenter Server system. To be able to manage the life cycle of your third-party solutions on a cluster managed with images, you must integrate the solutions with vSphere Lifecycle Manager.

Examples of VMware integrated solutions are VMware NSX®, vSAN, and vSphere with Tanzu. For more information about the VMware integrated solutions, see *vSphere Lifecycle Manager Images and Other VMware Products and Solutions* section in the *Managing Host and Cluster Lifecycle* documentation.

Third-party software providers can use the vSphere APIs, VMware vSphere APIs for I/O Filtering (VAIO), VMware Daemon Software Development Kit, and others to develop third-party solutions for their vSphere platforms.

You can use the vSphere Lifecycle Manager automation APIs to manage the life cycle of third-party solutions on a cluster managed with a single image. First you need to package and upload your solution components to thevSphere Lifecycle Manager depot. Use the depot to store and manage the software updates for your third-party solutions. To make a solution available on a cluster, create a software specification that contains the solution and remediate all hosts in the cluster with that image. vSphere Lifecycle Manager manages the life cycle of the solution components by consuming the software updates from the vSphere Lifecycle Manager depot.

Note If you export an image from the cluster where your third-party solution is running, the solution components are not part of the exported image.

Prerequisites

To enable vSphere Lifecycle Manager to manage your third-party solutions, you must use the ESXi Packaging Kit (EPK) to create installable packages. As of the vSphere 7.0 release, partner development kits generate components as installation packages. For more information about how to use the EPK to create components, assemble components into add-ons, then merge the add-ons with the base image to author a depot, see the *ESXi Packaging Kit (EPK) Development Guide* documentation.

Integrate a Third-Party Solution to Work with vSphere Lifecycle Manager

1 Create an online or offline depot to host your third-party solutions. See Working with Online Depots and Working with Offline Depots.

   - The following example creates an online depot that can be accessed through the `http://my_online_depot.com` URL.

     ```
     POST https://<vcenter_server_ip_or_fqdn>/api/esx/settings/depots/online
      {
     ```

```
    "description" : "My online depot adds the My Solution component to the ESXi 7.0U3d
base image",
    "ownerdata" : "ACME Company",
    "location" : "http://my_online_depot.com",
    "enabled" : true
}
```

- The following example creates an offline depot, also called an offline bundle, that can be imported to the vSphere Lifecycle Manager depot.

```
POST https://{server}/api/esx/settings/depots/offline?vmw-task=true

{
    "file_id" : "string",
    "description" : "string",
    "ownerdata" : "string",
    "source_type" : "PULL",
    "location" : "http://myurl.com"
}
```

2 Synchronize the online depot to download the depot metadata and make the vSphere Lifecycle Manager aware of your solutions. See Synchronizing Software Depots.

```
POST https://<vcenter_server_ip_or_fqdn>/api/esx/settings/depots/online?vmw-
task=true&action=sync
```

3 Create a draft software specification. See Creating a Draft Software Specification.

```
POST https://<vcenter_server_ip_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/
drafts
```

4 Add your solution to the created draft software specification. See Editing a Draft Software Specification.

```
PUT https://<vcenter_server_ip_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/
drafts/<draft_id>/software/components/<component_id>
```

5 Save the created draft software specification to make it the desired state for the cluster. See Committing the Draft Software Specification.

```
POST https://<vcenter_server_ip_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software/
drafts/<draft_id>?action=commit&vmw-task=true
```

6 Remediate the ESXi cluster with the desired state that contains your solution to apply the desired state on all hosts in that cluster. See Remediating an ESXi Cluster and a Standalone Host.

```
POST https://<vcenter_server_ip_or_fqdn>/api/esx/settings/clusters/<cluster_id>/software?
action=apply&vmw-task=true
{
    "hosts" : [
        "obj-103",
        "obj-103"
    ],
    "commit" : "obj-103",
    "accept_eula" : true
}
```

Results

You set up an image for the cluster which includes your third-party solution. You now manage all hosts in the cluster collectively with a single image. Upon remediation, the image is installed on all hosts in the cluster.

What to do next

You can update, delete or add new solutions to the draft software specification and then commit the changes to make the draft the desired state for the cluster.

Enable vSphere Lifecycle Manager on a Cluster Managed with Baselines

For more information about how to convert a cluster to use vSphere Lifecycle Manager images instead of baselines, see Enabling an Existing Cluster to Use vSphere Lifecycle Manager.

vSphere Lifecycle Manager APIs Equivalent to the vSphere Host Patch Manager APIs

To manage the life cycle of a single host with baselines, you use the vSphere Host Patch Manager APIs which are part of the vSphere Web Services API. To manage the life cycle of all hosts in a cluster collectively with an image, you use the vSphere Lifecycle Manager APIs which are part of the vSphere Automation APIs. The following table compares the APIs for life cycle management of the hosts and clusters in your environment.

Operation	vSphere Lifecycle ManagerAPI	vSphere Host Patch Manager API
Check whether hosts can be remediated	`com.vmware.esx.settings.clusters.Software.check`	`vim.host.PatchManager.CheckHostPatch_Task`
Check the compliance of the cluster or host with the desired state	`com.vmware.esx.settings.clusters.Software.scan`	`vim.host.PatchManager.ScanHostPatchV2_Task`

Operation	vSphere Lifecycle ManagerAPI	vSphere Host Patch Manager API
Remediate the cluster or host with the desired state	`com.vmware.esx.settings.clusters.Software.apply`	`vim.host.PatchManager.InstallHostPatchV2_Task`
Retrieve information about the cluster or host that you want to remediate	■ `com.vmware.esx.settings.clusters.Software.get` - View information about the current desired state. ■ `com.vmware.esx.settings.clusters.software.Components.list` - View all components in the current desired state. ■ `com.vmware.esx.settings.clusters.software.Components.get` - View detailed information about a specific component.	`vim.host.PatchManager.QueryHostPatch_Task` - View information about the bulletins installed on an ESXi host.
Stage a desired state on a cluster or host	`com.vmware.esx.settings.clusters.Software.stage`	`vim.host.PatchManager.StageHostPatch_Task`
Uninstall a component	`com.vmware.esx.settings.clusters.Software.apply` - When a software component is removed from the desired software specification and then the cluster is remediated, the component will be uninstalled from all hosts in that cluster.	`vim.host.PatchManager.UninstallHostPatch_Task`

Virtual Machine Configuration and Management

5

A virtual machine is a software computer that, like a physical computer, runs an operating system and applications.

The virtual machine consists of a set of specification and configuration files and is backed by the physical resources of a host. Each virtual machine encapsulates a complete computing environment and runs independently of the underlying hardware.

Starting with vSphere 6.5, you can create virtual machines, configure virtual machine settings, and perform power operations on the virtual machines through the vSphere Automation APIs.

Starting with vSphere 7.0, you can use the vSphere Automation APIs to perform various virtual machine management operations. For example, you can deploy virtual machines by using several approaches, clone an existing virtual machine, create an instant clone of a running virtual machine. You can also install VMware Tools which enables you to manage the life cycle and customize the networking and identity settings of the guest operating system installed on the virtual machine.

Read the following topics next:

- Creating Virtual Machines
- Configuring Virtual Machines
- Managing Virtual Machines
- Managing Data Sets

Creating Virtual Machines

You can use the vSphere Automation APIs to create virtual machines depending on your needs and infrastructure setup.

You can create a basic virtual machine and then configure it according to your needs. You can also create a more comprehensive virtual machine and then edit its settings. To create a virtual machine, you must specify the datastore, resource pool, folder, or host where the virtual machine is placed. Later, you can customize the virtual machine by specifying the boot options, number of CPUs, the guest OS, and virtual NIC. See Creating a Virtual Machine Without a Clone or Template and Configuring Virtual Machines.

You can use a turned off virtual machine to create a VM template from which to deploy other virtual machines. See Create a VM Template in a Content Library from a Virtual Machine. You can also mark a virtual machine as template by calling the `VirtualMachine.MarkAsTemplate` method from the vSphere Web Services APIs. See *vSphere Web Services SDK Programming Guide*.

You can capture a virtual machine or a vApp in an OVF template and store the template in a content library. Then you can use the OVF template to deploy a virtual machine or a vApp in your environment. See Creating Virtual Machines and vApps from Templates in a Content Library.

Creating a Virtual Machine Without a Clone or Template

You can create a virtual machine by using the REST API.

When you create a virtual machine without a template or clone, you can configure the virtual hardware, including processors, hard disc, memory. To create a virtual machine, you must specify the virtual machine attributes by using the `VMCreateSpec` data structure. For example, you can specify a name, boot options, networking, and memory for the new virtual machine.

All attributes are optional except the virtual machine placement information that you must provide by using the `VMPlacementSpec` data structure. Use the virtual machine placement specification to set the datastore, cluster, folder, host, or resource pool of the created virtual machine. You must make sure that all these vSphere objects are located in the same data center in a vCenter Server instance.

For more information, refer to the *API Reference* documentation.

Configuring Virtual Machines

You configure a virtual machine in the process of creation.

You can later view and edit virtual machine settings by adding or changing the type of the storage controllers, configure the virtual disks, boot options, CPU and memory information, or networks. Virtual machine settings can be configured when cloning, registering and relocating an existing virtual machine.

Name and Location

You specify the display name and the location of the virtual machine by using the `VMCreateSpec` and `VMPlacementSpec` data structures.

When you create your virtual machine, use the `name` property of the `VMCreateSpec` data structure to set the display name of the virtual machine.

You must create also a `VMPlacementSpec` instance that describes the location of the virtual machine in regards to the resources of a given vCenter Server instance. Use the `placement` property of the `VMCreateSpec` data structure to set the placement information for the virtual machine. You can set one or all of the following vSphere resources: datastore, cluster, folder, host, and resource pool.

Hardware Version

The hardware version of a virtual machine reflects the virtual hardware features supported by a virtual machine. These features depend on the physical hardware available on the ESXi host on which the virtual machine is running.

Virtual hardware features include the BIOS and Extensible Firmware Interface (EFI), the maximum number of CPUs, the maximum memory configuration, and other hardware characteristics.

When you create a virtual machine, the default hardware version of the virtual machine is the most recent version available on the host where the virtual machine is created. For information about the latest VMware products and virtual hardware versions, see Virtual machine hardware versions (1003746).

To set a hardware version different than the default, use the `hardware_version` enumerated type of the `VMCreateSpec` data structure. For information about the hardware features available for the virtual hardware versions, see Hardware features available with virtual machine compatibility settings (2051652).

You can set a lower virtual hardware version of a virtual machine than the highest supported by the ESXi host on which the virtual machine is running. Setting a lower hardware version can provide flexibility and is useful in the following cases.

- To help you standardize testing and deployment in your environment.
- In case you do not need the hardware features of the latest hardware version of the host.
- To maintain compatibility with hosts with a lower hardware version.

Boot Options

You can configure the boot options of a virtual machine by using the `VmHardwareBootCreateSpec` data structure.

By specifying the properties of the `VmHardwareBootCreateSpec` data structure, you can select one of the following settings when booting the virtual machine.

- Delay - Indicates a delay in milliseconds before starting the firmware boot process when the virtual machine is powered on.
- Retry - Indicates whether the virtual machine automatically retries to boot after a failure.
- Retry delay - Indicates a delay in milliseconds before retrying the boot process after a failure.
- Enter setup mode - If set to `true`, indicates that the firmware boot process automatically enters BIOS setup mode the next time the virtual machine boots. The virtual machine resets this flag to `false` once it enters setup mode.
- EFI legacy boot - If set to `true`, indicates that the EFI legacy boot mode is used.

Guest Operating System

The guest operating system that you specify affects the supported devices and available number of virtual CPUs.

When you create a virtual machine, you specify the guest operating system by using the `VmGuestOS` enumerated type of the `VMCreateSpec` data structure. The `VmGuestOS` enumerated type defines the valid guest OS types that you can choose from for configuring a virtual machine.

After the create operation finishes successfully, you can install the guest operating system on the mew virtual machine in the same way as you install it on a physical machine. For further information on installing a guest operating system, refer to the *Guest Operating System Installation Guide* at http://partnerweb.vmware.com/GOSIG/home.html and the *vSphere Virtual Machine Administration* guide.

Starting with vSphere 7.0, you can use the vSphere Automation APIs to install the VMware Tool on the quest operating system and perform some guest OS customizations. See Installing VMware Tools.

CPU and Memory

The `VMCreateSpec` data structure allows you to specify the CPU and memory configuration of a virtual machine.

To change the CPU and memory configuration settings, use the `VmHardwareCpuUpdateSpec` and `VmHardwareMemoryUpdateSpec` data structures.

You can set the number of CPU cores in the virtual machine by using the `count` property of the `VmHardwareCpuUpdateSpec` data structure. The supported range of CPU cores depends on the guest operating system and virtual hardware version of the virtual machine. If you set the `hot_add_enabled` and `hot_remove_enabled` properties to `true`, you allow virtual processors to be added or removed from the virtual machine at runtime.

You can set the memory size of a virtual machine by using the `size_MiB` property of the `VmHardwareMemoryUpdateSpec` data structure. The supported range of memory sizes depends on the configured guest operating system and virtual hardware version of the virtual machine. If you set the `hot_add_enabled` property to `true` while the virtual machine is not powered on, you enable adding memory while the virtual machine is running.

Networks

You configure network settings so that a virtual machine can communicate with the host and with other virtual machines. When you configure a virtual machine, you can add network adapters (NICs) and specify the adapter type.

You can add virtual Ethernet adapters to a virtual machine by using the `VmHardwareEthernetCreateSpec` data structure. The `VmHardwareEthernetEmulationType` enumerated type defines the valid emulation types for a virtual Ethernet adapter. The `VmHardwareEthernetMacAddressType` enumerated type defines the valid MAC address origins for a virtual Ethernet adapter. You can set the MAC address type to `MANUAL`, `GENERATED`, or `ASSIGNED`. To specify the MAC address explicitly, select `MANUAL`.

You can specify also the physical resources that back a virtual Ethernet adapter by using the `VmHardwareEthernetBackingSpec` data structure. The `VmHardwareEthernetBackingType` enumerated type defines the valid backing types for a virtual Ethernet adapter. You can set the backing type to `STANDARD_PORTGROUP`, `HOST_DEVICE`, `DISTRIBUTED_PORTGROUP`, or `OPAQUE_NETWORK`.

Managing Virtual Machines

Virtual machines can be configured like physical computers.

You can change the guest operating system settings after installing VMware Tools. You can add and remove virtual machines from the vCenter Server inventory. You can also move virtual machines from one host or storage location to another.

Filtering Virtual Machines

You can retrieve commonly used information about virtual machines that match specific criteria. You can retrieve information for up to 4000 virtual machines in a single vCenter Server instance.

You can retrieve a list of virtual machines in a single vCenter Server instance by filtering the results based on a specific requirement. For example, you can use as filter criteria the power state of the virtual machines, or the host, cluster, data center, folder, or resource pool that must contain the virtual machines. In case you specify multiple filter criteria, only virtual machines that match all filter criteria are returned.

Installing VMware Tools

VMware Tools is a set of drivers and utilities that you install on the guest operating system of a virtual machine to enhance the performance of the guest OS. VMware Tools also improves the management of the virtual machine. For each guest OS, VMware provides a specific binary-compatible version of VMware Tools.

Before you install VMware tools, you must install and boot the guest operating system.

To mount and unmount the VMware Tools installer CD as a CD-ROM for the guest operating system, use the connect and disconnect operations of the `tools/installer` service.

To mount the VMware Tools, use the `POST https://<api_host>/api/vcenter/vm/<vm>/tools/installer?action=connect` HTTP request. On Windows guest operating systems with activated Autorun feature, this request automatically initiates the installation of the VMware Tools and requires a user input to complete. On other guest operating systems, this method only mounts the VMware Tools disk image on the virtual CD/DVD drive and the user is required to do some guest OS-specific actions. For example, for some Linux distributions, the user is required to extract the contents of the VMware Tools installation archive and run the installer.

To unmount the VMware Tools installer CD image from the virtual CD/DVD drive, use the `POST https://<api_host>/api/vcenter/vm/<vm>/tools/installer?action=disconnect` HTTP request.

To monitor the status of the VMware Tools installation, you can check the `Tools.Info.version-status` and `Tools.Info.run-state` from `Tools.get`.

To upgrade the VMware Tools, use the `POST https://<api_host>/api/vcenter/vm/<vm>/tools?action=upgrade` HTTP request. The request takes as arguments the ID of the virtual machine on which the VMware Tools is installed and running. Use the `command_line_options` string to specify the command-line options that you want to pass to the installer to modify the tools installation procedure. You can monitor the upgrade operation in the same way you monitor the installation operation.

To update the properties of the VMware Tools, use the `PATCH https://<api_host>/api/vcenter/vm/<vm>/tools` HTTP request. Pass as argument a `VmToolsUpdateSpec` data structure and define the tools upgrade policy settings for the virtual machine by using the `VmToolsUpgradePolicy` enumerated type.

Performing Virtual Machine Power Operations

You can start, stop, reboot, and suspend virtual machines by using the operations of the `Power` service.

A virtual machine can have one of the following power states:

- `POWERED_ON` - Indicates that the virtual machine is running. If a guest operating system is not currently installed, you can perform the guest OS installation in the same way as for a physical machine.
- `POWERED_OFF` - Indicates that the virtual machine is not running. You can still update the software on the physical disk of the virtual machine, which is impossible for physical machines.
- `SUSPENDED` - Indicates that the virtual machine is paused and can be resumed. This state is the same as when a physical machine is in standby or hibernate state.

To perform a power operation on a virtual machine, you can use one of the operations of the `Power` service. Before you use one of the operations to change the power state of a virtual machine, you must first check the current state of the virtual machine by using the `GET https://<api_host>/api/vcenter/vm/<vm>/power` HTTP request. Pass as argument the virtual machine identifier.

Following is a list if the power operations, which you can use with a `POST` HTTP request:

- `power?action=start` - Powers on a powered off or suspended virtual machine. The method takes as argument the virtual machine identifier.
- `power?action=stop` - Powers off a powered on or suspended virtual machine. The method takes as argument the virtual machine identifier.
- `power?action=suspend` - Pauses all virtual machine activity for a powered on virtual machine. The method takes as argument the virtual machine identifier.
- `power?action=reset` - Shuts down and restarts the guest operating system without powering off the virtual machine. Although this method functions as a `stop` operation that is followed by a `start` operation, the two operations are atomic with respect to other clients, meaning that other power operations cannot be performed until the `reset` operation completes.

Registering and Unregistering Virtual Machines

When you create a virtual machine, it becomes part of the vCenter Server inventory and is registered to the host and vCenter Server. If you remove a virtual machine from the vCenter Server inventory, it becomes unusable. Virtual machine files remain in the same datastore but you cannot power on the virtual machine when it is not registered in the inventory.

You can temporarily remove a virtual machine from vCenter Server by unregistering it. Virtual machine files are not deleted from the datastore. Though all high-level information about the virtual machine such as statistics, resource pool association, permissions, and alarms, is removed from the host and the vCenter Server instance. To remove a virtual machine from the inventory, use the `POST https://<`*`api_host`*`>/api/vcenter/vm/<`*`vm`*`>?action=unregister` HTTP request and specify the ID of the virtual machine.

To restore a virtual machine to the vCenter Server inventory, and make it usable again, use the `POST https://<`*`api_host`*`>/api/vcenter/vm/<`*`vm`*`>?action=register` HTTP request. You pass as argument a `VMRegisterSpec` data structure that contains information about the current location of the virtual machine files on the datastore. You can also define the location within the vCenter Server inventory, for example, the cluster, folder, or the host, where you want to register the virtual machine. After registration, the virtual machine takes its resources (CPU, memory, and so on) from the resource pool or host to which it is registered.

If you no longer need a virtual machine and you want to free up datastore space, you can permanently delete a virtual machine from the inventory. Use the `DELETE https://<`*`api_host`*`>/api/vcenter/vm/<`*`vm`*`>` HTTP request and specify the ID of the virtual machine. Upon a successful completion of the operation, the virtual machine files are removed from the datastore, including the configuration file and the virtual disk files.

Managing Data Sets

You can use data sets to share information between a virtual machine and its guest operating system.

Data sets provide a communication infrastructure that you can use to share information between the host and guest. This information can then be used by guest agents or applications to configure the guest itself or guest applications. The information is grouped into data sets, each of which contains key-value entries comprising the data. Each application that uses the service should have at least one unique data set in which to store its data to avoid conflict with other applications. Each data set has attributes that define its access control and interoperability configuration.

Note You should not store sensitive data, such as passwords or private keys, in plain text.

To perform data set operations, the virtual machine must be running virtual hardware version 20 or later.

You should modify each data set by using the application that originally created that data set. If your application modifies a data set owned by another application, the other application might stop working.

For more information about data sets, see *vSphere Virtual Machine Administration* and the *VMware Guest SDK Programming Guide*.

Data Set Operations

You can use the vSphere Automation API to manipulate entries in data sets.

You can use data set operations to share information between a virtual machine and its guest operating system. The available operations include listing data sets, retrieving data set information, creating, modifying, and deleting a data set. You can also manipulate individual entries in a data set. The available operations include listing entry keys in a data set, retrieving the value of an entry in a data set, modifying and deleting an entry in a data set.

Table 5-1. User Operations

Operation	Description
List data sets	You can list all available data sets of a virtual machine.
Get data set	You can retrieve information about a specific data set.
Create data set	You can create a new data set.
Update data set	You can modify the attributes of a data set.
Delete data set	You can delete a data set.
List data set entry keys	You can list all entry keys in a data set.
Get data set entry key value	You can retrieve the value of a specific entry in a data set.

Table 5-1. User Operations (continued)

Operation	Description
Set data set entry key value	You can create or update an entry in a data set. If the new entry does not contain a key that matches an existing key, the operation creates a new entry. If the new entry contains a key that matches an existing key, the new key overwrites the value of the existing key.
Delete data set entry key value	You can delete an entry in a data set.

HTTP Requests for Data Set Operations

You can use HTTP requests to retrieve, create, update, or delete data sets and date set entries.

HTTP Requests

The following HTTP requests show the syntax that you can use to perform the available user operations.

- List data sets

  ```
  GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets
  ```

- Get data set

  ```
  GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets/<dataSet>
  ```

- Create data set

  ```
  POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets
  ```

- Update data set

  ```
  PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets/<dataSet>
  ```

- Delete data set

  ```
  DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets/<dataSet>?
  force={true}
  ```

- List data set entry keys

  ```
  GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets/<dataSet>/entries
  ```

- Get data set entry key value

  ```
  GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets/<dataSet>/entries/
  <key>
  ```

- Set data set entry key value

  ```
  PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets/<dataSet>/entries/
  <key>
  ```

- Delete data set entry key value

  ```
  DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm/<vm>/data-sets/<dataSet>/
  entries/<key>
  ```

For information about the content and syntax of the HTTP query parameters, path parameters, and request body parameters, see the *API Reference* documentation.

Using Content Libraries

6

Use the vSphere Automation API to access the Content Library service and use it to store and share different types of content in your vSphere environment. Content libraries are vSphere container objects for storing and sharing OVF and OVA packages, VM templates, vApp templates, and other types of files.

You can use content libraries to share VM and vApp templates, and other types of files, such as ISO images, text files, and so on, across the vCenter Server instances in the same or different locations. Sharing templates across your virtual environment promotes consistency, compliance, efficiency, and automation in deploying workloads at scale.

You use library items to store and manage content in a content library. A single library item can contain one or more files. For example, an OVF package consists of a `.vmdk` file, manifest file, and others, but it is represented by a single library item.

Starting with vSphere 7.0 Update 3, you can add a security policy to a local or subscribed content library and thus protect your system when you synchronize or download library content from third party providers.

- What Is a Content Library

  A content library instance represents a container for a set of library items. A content library item instance represents the logical object stored in the content library, which might be one or more usable files.

- Querying Content Libraries

  You can create queries to find libraries that match your criteria. You can also retrieve a list of all libraries or only the libraries of a specific type.

- Working with Content Libraries

  The Content Library API provides services that you can use to create and manage content libraries.

- Working with Library Items

  A library item groups multiple files within one logical unit. You can perform various tasks with the items in a content library.

- Content Library Support for OVF and OVA Packages

  You can use the objects and methods provided by the Content Library API to manage OVF and OVA packages.

- Creating Virtual Machines and vApps from Templates in a Content Library

  You can create VM and OVF templates from virtual machines and vApps in your inventory. You can then deploy virtual machines and vApps from the templates that are stored in a content library.

What Is a Content Library

A content library instance represents a container for a set of library items. A content library item instance represents the logical object stored in the content library, which might be one or more usable files.

You create and manage the content of a content library on a single vCenter Server instance, but you can distribute the content to other vCenter Server instances. Depending on your needs, you can maintain two types of content libraries: local and subscribed. You can shape the contents of a library item and then combine several library items in a local content library. Furthermore, you can publish the library to make its content available to other users.

What to read next

- Content Library Types

  You can create two types of libraries, local and subscribed.
- Content Library Items

  Library items are VM templates, vApp templates, or other VMware objects that can be contained in a content library.
- Content Library Storage

  When you create a local library, you can store its contents on a datastore managed by the vCenter Server instance or on a remote file system.

Content Library Types

You can create two types of libraries, local and subscribed.

Local library.

You can create a local library as the source for content you want to save or share. Create the local library on a single vCenter Server instance. You can add items to a local library or remove them. You can publish a local library and as a result this content library service endpoint can be accessed by other vCenter Server instances in your virtual environment. When you publish a library, you can configure a password for authentication.

Subscribed library.

You can create a subscribed library and populate its content by synchronizing to a local library. A subscribed library contains copies of the local library files or just the metadata of the library items. The local library can be located on the same vCenter Server instance as the subscribed library, or the subscribed library can reference a local library on a different vCenter Server instance. You cannot add library items to a subscribed library. You can only add items to the source library. After synchronization, both libraries will contain the same items.

Content Library Items

Library items are VM templates, vApp templates, or other VMware objects that can be contained in a content library.

VMs and vApps have several files, such as log files, disk files, memory files, and snapshot files that are part of a single library item. You can create library items in a specific local library or remove items from a local library. You can also upload files to an item in a local library so that the libraries subscribed to it can download the files to their NFS or SMB server, or datastore.

For information about the tasks that you can perform by using the content library service, see Working with Content Libraries.

Content Library Storage

When you create a local library, you can store its contents on a datastore managed by the vCenter Server instance or on a remote file system.

Depending on the type of storage that you have, you can use Virtual Machine File System (VMFS) or Network File System (NFS) for storing content on a datastore.

For storing content on a remote file system, you can enter the path to the NFS storage that is mounted on the Linux file system of the vCenter Server instance. For example, you can use the following URI formats: `nfs://<server>/<path>?version=4` and `nfs://<server>/<path>`. If you have a vCenter Server instance that runs on a Windows machine, you can specify the Server Massage Block (SMB) URI to the Windows shared folders that store the library content. For example, you can use the following URI format: `smb://<unc-server>/<path>`.

Querying Content Libraries

You can create queries to find libraries that match your criteria. You can also retrieve a list of all libraries or only the libraries of a specific type.

Listing All Content Libraries

You can retrieve a list of all content library IDs in your virtual environment, regardless of their type, by using the `Library` service. You can use the `List Library` operation to retrieve all local and subscribed libraries in your system.

Listing Content Libraries of a Specific Type

You can use the vSphere Automation API to retrieve content libraries of a specific type. For example, you can list only the local libraries in your virtual environment.

If you want to retrieve only a list of the local libraries, you must retrieve the `LocalLibrary` service and use the `List Local Library` operation on the `LocalLibrary` service. To list only subscribed libraries, you must retrieve the `SubscribedLibrary` service and call the `List Subscribed Library` operation on the `SubscribedLibrary` service.

Listing Content Libraries by Using Specific Search Criteria

You can filter the list of content libraries and retrieve only the libraries that match your specific criteria. For example, you might want to publish all local libraries with a specific name.

To filter with specific search criteria, use the `Find Library` operation of the `Library` service. Specify the properties of the `LibraryFindSpec` data structure that contains your search criteria. Upon a successful completion of the operation, you receive a list of all content libraries that match your search criteria.

Working with Content Libraries

The Content Library API provides services that you can use to create and manage content libraries.

You can create a local library and publish it for the entire virtual environment. You can also subscribe to use the contents of a local library and enable automatic synchronization to ensure that you have the latest content.

You can perform content library operations by using HTTP requests. Some operations require you to specify parameters in the body of the HTTP request according to your environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Creating a Local Content Library

You can create a local content library by using the `POST https://<vcenter_ip_address_or_fqdn>/api/content/local-library` HTTP request.

Publishing an Existing Content Library

To make the library content available for other vCenter Server instances across the vSphere Automation environment, you must publish the library. Depending on your workflow, select a method for publishing the local library. You can publish a local library that exists in your vSphere Automation environment by using the `POST https://<vcenter_ip_address_or_fqdn>/api/content/local-library/<`*`library_id`*`>?action=publish` HTTP request.

Subscribing to a Content Library

You can subscribe to local content libraries. When you subscribe to a library, you must specify the backing storage for the library content. If the library requires basic authentication, you must also provide the correct user name and password. You can create a subscribed library by using the `POST https://<vcenter_ip_address_or_fqdn>/api/content/subscribed-library` HTTP request.

Synchronizing a Subscribed Content Library

When you subscribe to a published library, you can configure the settings for downloading and updating the library content. You can force the synchronization of a subscribed library by using the `POST https://<vcenter_ip_address_or_fqdn>/api/content/subscribed-library/<library_id>?action=sync` HTTP request.

Editing the Settings of a Content Library

You can update the settings of content library types in your virtual environment. You can update the settings of a local library by using the `PATCH https://<vcenter_ip_address_or_fqdn>/api/content/local-library/<library_id>` HTTP request. You can update the settings of a subscribed library by using the `PATCH https://<vcenter_ip_address_or_fqdn>/api/content/subscribed-library/<library_id>` HTTP request.

Removing the Content of a Subscribed Library

You can free storage space in your virtual environment by removing the subscribed library content that you no longer need.

You can create a subscribed library with the option to download the library content on demand. As a result, only the metadata for the library items is stored in the associated with the subscribed library storage. When you want to deploy a virtual machine from a VM temple in the subscribed library, you must synchronize the subscribed library to download the entire published library content. When you no longer need the VM template, you can use the `POST https://<vcenter_ip_address_or_fqdn>/api/content/subscribed-library/<library_id>?action=evict` HTTP request. You must provide the subscribed library ID to this function. As a result, the subscribed library content that is cached on the backing storage is deleted.

If the subscribed library is not configured to synchronize on demand, an exception is thrown. In this case the subscribed library always attempts to have the most recent published library content.

Delete a Content Library

When you no longer need a content library, you can use the `Delete` operation on either the `Local Library` or the `Subscribed Library` service depending on the library type. You can delete a local library by using the `DELETE https://<vcenter_ip_address_or_fqdn>/api/content/local-library/<library_id>` HTTP request. You can delete a subscribed library by using the `DELETE https://<vcenter_ip_address_or_fqdn>/api/content/subscribed-library/<library_id>` HTTP request.

Working with Library Items

A library item groups multiple files within one logical unit. You can perform various tasks with the items in a content library.

You can upload files to a library item in a local library and update existing items. You can download the content of a library item from a subscribed library and use the item, for example, to deploy a virtual machine. You can remove the content of a library item from a subscribed library to free storage space and keep only the metadata of the library item. When you no longer need local library items, you can delete them and they are removed from the subscribed library when a synchronization task is completed.

You can create a library item from a specific item type, for example, **`.ovf`** and VM template. The Content Library service must support the library item type to handle the item correctly. If no support is provided for a specified type, the Content Library service handles the library item in the default way, without adding metadata to the library item or guiding the upload process. For information about the supported VM template types, see the *vSphere Virtual Machine Administration* documentation.

Creating an Empty Library Item

You can create as many library items as needed and associate them with a local content library. You can create a library item by using the `POST https://<vcenter_ip_address_or_fqdn>/api/content/library/item` HTTP request.

Querying Library Items

You can perform numerous query operations on library items.

You can retrieve a list of all items in a library, retrieve a library item that has a specific type or name, and find a library item that is not cached on the disk. You can then update the library item content from the subscribed library.

You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/content/library/item?library_id` HTTP request to retrieve a list of all items in a particular library.

You can filter the items contained in a library and retrieve only the items matching specific criteria. For example, you might want to remove or update only specific items in a library. You can use the `POST https://<vcenter_ip_address_or_fqdn>/api/content/library/item?action=find` HTTP request to filter items in a particular library.

Editing the Settings of a Library Item

You can use the `PATCH https://<vcenter_ip_address_or_fqdn>/api/content/library/item/<library_item_id>` HTTP request to edit the properties of a library item such as name, description, and type.

Uploading a File from a Local System to a Library Item

You can upload different types of files from a local system to a library item that you want to use in the vSphere Automation environment.

To upload a file, you must create an empty library item and verify that you have access to the `UpdateSession` and `File` services.

1 Use the `LibraryItemUpdateSessionModel` data structure to track the changes that you make to the library item.

2 Create an update session by using the `UpdateSession` service.

3 Use the `LibraryItemUpdatesessionFileAddSpec` data structure to describe the upload method and other properties of the file to be uploaded.

4 Create the request for changing the item by using the `File` service.

5 Upload the file that is on the local system.

6 Complete and delete the update session to apply the changes to the library item.

Upload a File from a URL to a Library Item

You can upload different types of files from a URL to a library item that you want to use in the vSphere Automation environment.

To upload a file, you must first create an empty library item and verify that you have access to the `UpdateSession` and `File` services.

1 Use the `LibraryItemUpdateSessionModel` data structure to track the changes that you make to the library item.

2 Create an update session by using the `UpdateSession` service.

3 Create a file specification to describe the upload method and other properties of the file to be uploaded.

4 Specify the location of the file you want to upload by using the `LibraryItemTransferEndpoint` data structure.

5 Add the file source endpoint to the file specification.

6 Create a request for changing the item by using the configured file specification.

7 Complete the update session to apply the changes to the library item.

Downloading Files to a Local System from a Library Item

You might want to download files to a local system from a library item and then make changes to the files before you use them.

1 Create a download session model to specify the item, which contains the file that you want to download.

2 Access the `File` service and retrieve the file that you want to export to your system within the new download session.

3 Prepare the files that you want to download and wait until the files are in the prepared state.

4 Retrieve the download endpoint URI of the files.

5 Download the files by using an HTTP GET request.

6 Delete the download session after all files are downloaded.

Synchronizing a Library Item in a Subscribed Content Library

The items in a subscribed library have features that are distinct from the items in a local library. Synchronizing the content and the metadata of an item in a subscribed library depends on the synchronization mechanism of the subscribed library. You can use the `PATCH https://<vcenter_ip_address_or_fqdn>/api/content/library/subscribed-item/<library_item_id>?action=sync` HTTP request to force the synchronization of an individual library item in a subscribed library.

Removing the Content of a Library Item

You can remove the content from a library item to free space on your storage.

If you create a subscribed library with the option to synchronize library content on demand, only the metadata for the library items is stored. When you want to use the items in the library, you must force synchronization on the items to download their content. When you no longer need the files in an item, you can remove the cached content of the library item and free storage space. You can use the `POST https://<vcenter_ip_address_or_fqdn>/api/content/library/subscribed-item/<library_item_id>?action=evict` HTTP request to evict the cached content of a library item in a subscribed library.

Deleting a Library Item

You can remove a library item from a local library when you no longer need it.

You can use the `DELETE https://<vcenter_ip_address_or_fqdn>/api/content/library/item/<library_item_id>` HTTP request to remove a library item from a library. The item content is asynchronously removed from the storage.

You cannot remove items from a subscribed library. If you remove an item from a local library, the item is removed from the subscribed library when you perform a synchronization task on the subscribed library item.

Content Library Support for OVF and OVA Packages

You can use the objects and methods provided by the Content Library API to manage OVF and OVA packages.

Open Virtualization Format (OVF) is an industry standard that describes metadata about a virtual machine image in an XML format. An OVF package includes an XML descriptor file and optionally disk images, resource files (such as ISO files), manifest files, and certificate files.

An OVA package is a single file that contains all OVF package files in an archived form. After you upload an OVA package to a content library, the OVA file is converted to the standard OVF package.

When you try to upload signed content to a content library, you might receive preview warnings. Signed content can be either OVF or OVA packages that contain manifest and certificate files. If you do not respond to the preview warnings, the upload fails. To complete an upload operation successfully, you must ignore any preview warnings by using the `LibraryItemUpdatesessionWarningBehavior` data structure.

With the vSphere Automation API, you can use the OVF package in a content library to deploy virtual machines and vApps on hosts, resource pools, and clusters. You can also use the API to create OVF packages in content libraries from vApps and virtual machines on hosts, resource pools, and clusters.

When you create library items to store OVF packages, you must set the item type to `ovf`. To comply with the specific standards of the OVF packages, the vSphere Automation API provides the `Item` service.

Working with OVF and OVA Packages in a Content Library

You can upload an OVF or OVA package to a library item by using the `UpdateSession` service. You can also download an OVF and OVA packages from a content library to your local file system.

In case you want to upload an OVF package, the location of the content determines whether you can pull the content from a URL or push the content directly to a content library. For information about uploading content to library items, see Uploading a File from a Local System to a Library Item and Upload a File from a URL to a Library Item.

To download the files that are included in an OVF or OVA package to your local file system, use the `DownloadSession` service. For more information, see Downloading Files to a Local System from a Library Item.

Upload an OVF or an OVA Package from a URL to a Library Item

You can upload an OVF or an OVA package from a Web server to a library item.

Note If you try to upload a signed OVF package and it returns preview warnings, you must ignore the preview warnings to complete the upload.

Prerequisites

- Create a new local content library or retrieve the desired existing content library.
- Required privileges: **Content library.Add library** item and **Content library.Update files** on the library.

Procedure

1 Create an empty library item.

2 Create an update session object.

3 Create an `LibraryItemUpdatesessionFileAddSpec` data structure to describe the properties and the upload location of the OVF descriptor file or of the OVA package file.

4 Link the `LibraryItemUpdatesessionFileAddSpec` data structure to the update session.

   All files that are included in the OVF package are automatically uploaded.

5 Complete the asynchronous transfer.

Upload an OVF or OVA Package from a Local File System to a Library Item

You can upload an OVF or OVA package from a local file system. This procedure describes how to use the `LibraryItemUpdatesessionFileAddSpec` data structure after you have created a library item and initiated an update session.

Note If you try to upload a signed OVF package and it returns preview warnings, you must ignore the preview warnings to complete the upload.

Prerequisites

- Create a new local content library or retrieve the desired existing content library.
- Required privileges: **Content library.Add library item** and **Content library.Update files** on the library.

Procedure

1 Create a library item.

2 Create an update session.

3 Create a `LibraryItemUpdatesessionFileAddSpec` data structure to describe the properties and the upload locations of the OVF descriptor file or of the OVA package file.

4 Link the `LibraryItemUpdatesessionFileAddSpec` data structure to the update session.

5 (Optional) Create a `LibraryItemUpdatesessionFileAddSpec` data structure for each VMDK file included in the OVF package.

6 Add all `LibraryItemUpdatesessionFileAddSpec` data structures to the update session.

   If you upload an OVF package and it has a VMDK file included, you must repeat steps 5 and 6. If you are uploading a signed OVF package, steps 5 and 6 must also be repeated for the manifest and certificate files included in the OVF package.

7 Initiate the upload operation.

8 Complete the update session.

9 Delete the session.

Creating Virtual Machines and vApps from Templates in a Content Library

You can create VM and OVF templates from virtual machines and vApps in your inventory. You can then deploy virtual machines and vApps from the templates that are stored in a content library.

Create a VM Template in a Content Library from a Virtual Machine

By using the vSphere Automation REST API, you can create a VM template in a content library from an existing virtual machine in your vCenter Server inventory.

When you call the `create` function of the `com.vmware.vcenter.vm_template.LibraryItems` service, a VM template is created as a library item in your local content library. If the operation is successful, the `LibraryItems` service returns the ID of the newly created library item.

To create a library item that contains a VM template, you can use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items` HTTP request. You can review the information about a VM template by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items/<vm_template_item_id>` HTTP request. For information about how to create a VM template by using the vSphere Client, see the *vSphere Virtual Machine Administration* documentation.

For information about the available and mandatory parameters, see the *API Reference* documentation.

Prerequisites

- Verify that you have administrative privileges on your vCenter Server instance.
- Verify that you created a vSphere Automation session to your vCenter Server.
- Verify that you created a local library by using the vSphere Client or the vSphere Automation APIs.

Procedure

1 Get the ID of your ESXi host on which you want to store the VM template.

You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/host` HTTP request.

2 Get the ID of the datastore on which you want to store the VM template files.

You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/datastore` HTTP request.

3 Get the ID of the virtual machine that you want to save as a VM template.

You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm` HTTP request.

4 Get the ID of your local library.

   You can get the list of the local libraries in your vCenter Server and review the information about each library by using the `GET https://<vcenter_ip_address_or_fqdn>/api/com/vmware/content/local-library` and `GET https://<vcenter_ip_address_or_fqdn>/api/com/vmware/content/library/id:<library_id>` HTTP requests.

5 Create a library item specification for the VM template.

   You can use the body of the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items` HTTP request.

   a Specify the local library, source virtual machine, and the name of the library item by using the `library`, `source_vm`, and `name` parameters. You must use the IDs of the local library and source virtual machine.

   b Specify the placement information for your VM template.

      You can use the `placement` parameters in the body of the HTTP request. To specify the host, resource pool, cluster, and folder, you must use their IDs.

   c Specify the datastore on which you want to store the log, configuration, and disk files of your VM template.

      To specify the storage backing for the VM template, you can use the `vm_home_storage` and `disk_storage` parameters in the body of the HTTP request. You must use the ID of the datastore.

   d Include the placement and storage specifications in the library item specification.

6 Create a library item for storing the VM template.

   You can use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items` HTTP request.

Results

If the operation is successful, the `LibraryItems` service returns the ID of the library item that contains the VM template. For information about the available responses, see the *API Reference* documentation.

What to do next

- Review the information stored in the library item by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items/<library_item_id>` HTTP request. If you did not save the ID of the library item holding the VM template, you can check the UUID by using the vSphere Client. The URN ends with the ID of the library item and has the following format: `urn:vapi:com.vmware.content.library.Item:<VMTemplateItemID>`.

Create an OVF Template in a Content Library from a Virtual Machine or vApp

You can create library items from existing virtual machines or vApp. Use those library items later to deploy virtual machines and vApps on hosts and clusters in your vCenter Server environment.

Procedure

1. Create a `LibraryItem.DeployableIdentity` instance to specify the source virtual machine or vApp to be captured in an OVF template.
2. Create a `LibraryItem.CreateTarget` instance to identify the content library where the OVF template is stored.
3. Create a `LibraryItem.CreateSpec` instance to specify the properties of the OVF template.
4. Initiate a synchronous create operation by using the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/ovf/library-item` HTTP request.
5. Verify the outcome of the create operation.

Deploy a Virtual Machine from a VM Template in a Content Library

By using the vSphere Automation APIs, you can deploy a virtual machine from a VM template stored in a content library.

To deploy a virtual machine from a VM template in a content library, use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items/<vm_template_item_id>?action=deploy` HTTP request. You can specify the power state and customize the guest operation system prior to the virtual machine deployment.

For information about the available and mandatory parameters, see the *API Reference* documentation.

Prerequisites

- Verify that you have administrative privileges on your vCenter Server instance.
- Verify that you created a vSphere Automation session to your vCenter Server instance.

Procedure

1. Review the information stored in the VM template library item.

   You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items/<vm_template_item_id>` HTTP request. If you did not save the ID of your item, you can select the UUID of your VM template item by using the vSphere Client. The URN ends with the ID of the item and has the following format: `urn:vapi:com.vmware.content.library.Item:<VMTemplateItemID>`.

2 Get the ID of the host on which you want to deploy the virtual machine.

You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/host` HTTP request.

3 Get the ID of the resource pool to which you want to add your virtual machine.

You can use the `https://<vcenter_ip_address_or_fqdn>/api/vcenter/resource-pool` HTTP request.

4 Get the ID of the `VIRTUAL_MACHINE` folder to which you want to add your virtual machine.

You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/folder` HTTP request.

5 Get the ID of the datastore on which you want to store log, configuration, and disk files of the virtual machine.

You can use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/datastore` HTTP request.

6 Create a deployment specification.

You can use the body of the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items/<vm_template_item_id>?action=deploy` HTTP request.

a Specify a name and description of the virtual machine that you want to deploy.

b Specify the place in your inventory on which you want to deploy the virtual machine such as an ESXi host, resource pool, and VM folder.

You can use the `placement` parameter in the body of the request. You must use the IDs of your inventory objects.

c Specify the datastore on which you want to store the log, configuration, and disk files of the virtual machine. You must use the ID of the datastore.

You can use the `vm_home_storage`, and `disk_storage` parameters in the body of the request.

d (Optional) Specify the guest operating system and hardware customization specifications that you want to apply to the virtual machine during the deployment process. Add this information to the deployment specification.

You can use the `guest_customization` and `hardware_customization` parameters in the body of the request. You can get a list of the guest operating system customization specifications that are available in your vCenter Server by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/guest/customization-specs` HTTP request.

e Include the placement and storage specifications in the deployment specification.

7 Deploy a virtual machine from your VM template.

You can use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm-template/library-items/<vm_template_item_id>?action=deploy` HTTP request.

Results

If the operation is successful, the ID of the deployed virtual machine is returned. For information about the possible exceptions, see the *API Reference* documentation.

Deploy a Virtual Machine or vApp from an OVF Template in a Content Library

You can use the `LibraryItem` service to deploy a virtual machine or vApp on a host, cluster, or resource pool from a library item.

Procedure

1 Create a `LibraryItem.DeploymentTarget` instance to specify the deployment location of the virtual machine or vApp.

2 Create a `LibraryItem.ResourcePoolDeploymentSpec` instance to define all necessary parameters for the deployment operation.

For example, you can assign a name for the deployed virtual machine or vApp, and accept the End User License Agreements (EULAs) to complete the deployment successfully.

3 (Optional) Retrieve information from the descriptor file of the OVF template and use the information during the OVF template deployment.

4 Use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/ovf/library-item/<ovf_library_item_id>?action=deploy` HTTP request.

5 Verify the outcome of the deployment operation.

vSphere Tag Service

7

The vSphere Automation Tag Service supports the definition of tags that you can associate with vSphere objects or vSphere Automation resources.

Starting with vSphere 6.5, the vSphere Automation APIs provide programmatic access to creating and managing vSphere tags in your vSphere inventory.

For example, if you want to tag your virtual machines by guest operating system type, you can create a category called **`operating system`**. You can specify that it applies to virtual machines only and that only a single tag can be applied to a virtual machines at any time. This category can include the following tags: **`Windows`**, **`Linux`**, and **`Mac OS`**.

Read the following topics next:

- Creating vSphere Tags
- Creating Tag Associations
- Updating a Tag
- Using Tags to Create and Manage Compute Policies

Creating vSphere Tags

You create a vSphere tag to add metadata to objects in the vSphere inventory.

Tags are grouped in categories and each tag must have at least one category related to it. After you create the tag, you can associate the tag with a vSphere object.

Tags and categories can span multiple vCenter Server instances.

- If multiple on-premises vCenter Server instances are configured to use Enhanced Linked Mode, tags and tag categories are replicated across all these vCenter Server instances.
- When you use Hybrid Linked Mode, tags and tag categories are maintained across your linked domain. That means the on-premises SDDC and the VMware Cloud on AWS SDDC share tags and tag attributes. For more information about Hybrid Linked Mode, see "Hybrid Linked Mode" in the *VMware Cloud on AWS Product Documentation*.

Creating a Tag Category

You create tags in the context of a tag category. You must create a category before you can add tags within that category.

A tag category has the following properties:

- name
- description
- cardinality, or how many tags it can contain
- the types of elements to which the tags can be assigned

You can associate tags with both vSphere API managed objects and VMware vSphere Automation API resources.

Creating a Tag

After you create a tag category, you can create tags within that category

A tag has the following properties:

- name
- description
- category ID

Creating Tag Associations

After you create a tag category and create a tag within the category, you can associate the tag with a vSphere managed object or a vSphere Automation resource.

An association is a simple link that contains no data of its own. You can enumerate objects that are attached to a tag or tags that are attached to an object.

Tag associations are local to a vCenter Server instance. When you request a list of tag associations from a vCenter Server system, it enumerates only the associations that it has stored.

When you associate a tag with an object, the object's type must match one of the associable types specified for the category to which the tag belongs.

Assign the Tag to a Content Library

After you create a tag, you can assign the tag to a vSphere Automation resource.

Procedure

1 Construct a dynamic object identifier for the library.

The dynamic identifier includes the type and ID of the object.

2 Attach the tag to the content library.

```
POST https://<vcenter_ip_address_or_fqdn>/api/cis/tagging/tag-association/<tag_id>?
action=attach
```

Assign a Tag to a Cluster

After you create a tag, you can assign the tag to a vSphere managed object. Tags make the inventory objects in your virtual environment more sortable and searchable.

This procedure describes the steps for applying a tag to a cluster object in your inventory.

Prerequisites

Obtain the managed object identifier for the specified cluster.

To get the managed object identifier of the `ClusterComputeResource`, you must access vCenter Server by using the vSphere Web Services API.

Procedure

1 Construct a dynamic object identifier for the cluster.

The dynamic identifier includes the type and ID of the managed object reference.

2 Attach the tag to the cluster.

```
POST https://<vcenter_ip_address_or_fqdn>/api/cis/tagging/tag-association/<tag_id>?
action=attach
```

Updating a Tag

To update a tag, you must create an update spec for the tag.

In the update spec, you set values for the fields to be changed, and omit values for the other fields. When you do an update operation using the update spec, only the fields that contain values are changed.

For example, you might use a timestamp in a tag description to identify a resource's last reconfiguration. After reconfiguring the resource, you update the tag description to contain the current time.

Using Tags to Create and Manage Compute Policies

Compute policies rely on tags to identify inventory objects on which to enforce a policy.

Create a Compute Policy

You can create a compute policy and check the compliance status of the policy for a specific virtual machine.

Procedure

1 Retrieve the object ID of a virtual machine.

2 Retrieve the object ID of the host you want the virtual machine to run on.

Note The host should be different from the host the virtual machine is already running on.

3 Tag the virtual machine.

For example, use the *tag-1* tag.

4 Tag the host.

For example, use the *tag-2* tag.

5 Create a compute policy.

For example, use the *tag-1* and *tag-2* tags.

6 Check the compliance status of the policy on this virtual machine.

Note The compliance status can take up to 3 minutes to update.

Using HTTP Requests to Manage the Compute Policy Workflow

You can use HTTP requests to manage compute policy capabilities, create tags and associations, create, list, and use policies.

The following sequence represents a typical workflow of discovering all compute policy capabilities available on a vCenter Server instance and creating policies based on such capabilities.

What to read next

Procedure

1 Manage Compute Policy Capabilities

You can list capabilities, retrieve details about a specific capability, and use the details to create a compute policy.

2 Create Tags and Associations

You must create a category and specify the associable types and cardinality for the tags that make up the category. After you create a category, you can create tags for the category.

3 Create, List, and Use Policies

You can create various types of policies, list all policies on a vCenter Server instance, list and filter tags used by policies.

Manage Compute Policy Capabilities

You can list capabilities, retrieve details about a specific capability, and use the details to create a compute policy.

The `capabilities` service provides operations to manage compute policy capabilities. The description of the capability provides information about the intent of a policy based on this capability. A capability provides a type to create a policy. A capability also provides a type that describes the information returned when retrieving information about a policy.

Procedure

1 List all available compute policy capabilities.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/compute/policies/capabilities
```

The following is an example response that contains details about the capabilities.

```
[
    {
        "capability":
"com.vmware.vcenter.compute.policies.capabilities.vm.placement.anti_affinity_with_vcls",
        "name": "Anti-affinity with vSphere Cluster Services (vCLS) VMs",
        "description": "vSphere Cluster Services (vCLS) VMs will not be placed on hosts
with virtual machines that share the tag."
    }
]
```

2 Retrieve details about a specific compute policy.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/compute/policies/capabilities/
com.vmware.vcenter.compute.policies.capabilities.vm.placement.anti_affinity_with_vcls
```

The following is an example response that contains details about the capability.

```
{
    "info_type":
"com.vmware.vcenter.compute.policies.capabilities.vm.placement.anti_affinity_with_vcls.info
",
    "name": "Anti-affinity with vSphere Cluster Services (vCLS) VMs",
    "description": "vSphere Cluster Services (vCLS) VMs will not be placed on hosts with
virtual machines that share the tag.",
    "create_spec_type":
"com.vmware.vcenter.compute.policies.capabilities.vm.placement.anti_affinity_with_vcls.crea
te_spec"
}
```

3 Use the details to construct a `create_spec_type` for a policy.

```
GET https://<vcenter_ip_address_or_fqdn>/api/com/vmware/vapi/metadata/metamodel/structure/
id:com.vmware.vcenter.compute.policies.capabilities.anti_affinity_with_vcls.create_spec
```

What to do next

You can create tags to identify inventory objects on which a compute policy should be enforced. See Create Tags and Associations.

Create Tags and Associations

You must create a category and specify the associable types and cardinality for the tags that make up the category. After you create a category, you can create tags for the category.

Procedure

1 Create a tagging category.

```
POST https://<vcenter_ip_address_or_fqdn>/api/com/vmware/cis/tagging/category
```

The following example `create_spec` specifies the associable types and cardinality for the tags that make up the category.

```
{
    "create_spec": {
        "associable_types": [
            "HostSystem"
        ],
        "cardinality": "SINGLE",
        "description": "cat-4-desc",
        "name": "cat-4"
    }
}
```

The following is an example response that contains the category ID.

```
{
    "value": "urn:vmomi:InventoryServiceCategory:ce8902ef-b5e0-4aab-8ffa-
ec67b15eb336:GLOBAL"
}
```

2 Create a tag for the tagging category.

```
POST https://<vcenter_ip_address_or_fqdn>/api/com/vmware/cis/tagging/tag
```

The following example `create_spec` specifies the ID of the category for which you create the tag.

```
{
    "create_spec": {
        "category_id": "urn:vmomi:InventoryServiceCategory:ce8902ef-b5e0-4aab-8ffa-
ec67b15eb336:GLOBAL",
        "description": "tag-1-desc",
        "name": "tag-1"
    }
}
```

The following is an example response that contains the tag ID.

```
{
    "value": "urn:vmomi:InventoryServiceTag:a8d0d278-edd1-4a79-86ac-fa9384595062:GLOBAL"
}
```

3 List all tags on a vCenter Server instance.

```
GET https://<vcenter_ip_address_or_fqdn>/api/com/vmware/cis/tagging/tag
```

The following is an example response that contains the ID of the available tag.

```
{
    "value": [
        "urn:vmomi:InventoryServiceTag:a8d0d278-edd1-4a79-86ac-fa9384595062:GLOBAL"
    ]
}
```

4 List all virtual machines in the inventory.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/vm
```

The following is an example response that contains details about the available virtual machine.

```
{
    "value": [
        {
            "memory_size_MiB": 40,
            "vm": "vm-21",
            "name": "a_vm",
            "power_state": "POWERED_OFF",
            "cpu_count": 1
        }
    ]
}
```

5 Attach a tag to a virtual machine.

```
POST https://<vcenter_ip_address_or_fqdn>/api/com/vmware/cis/tagging/tag-association/
id:<tag-id>?~action=attach
```

The following example specifies the details of the virtual machine you want to attach.

```
{
    "object_id": {
        "id": "vm-21",
        "type": "VirtualMachine"
    }
}
```

If the response is empty, the operation is successful.

6 List all tags attached to a virtual machine.

```
POST https://<vcenter_ip_address_or_fqdn>/api/com/vmware/cis/tagging/tag-association?
~action=list-attached-tags
```

The following example specifies the details of the virtual machine for which you want to list the attached tags.

```
{
    "object_id": {
        "id": "vm-21",
        "type": "VirtualMachine"
    }
}
```

The following is an example response that contains the ID of the attached tag.

```
{
    "value": [
        "urn:vmomi:InventoryServiceTag:a8d0d278-edd1-4a79-86ac-fa9384595062:GLOBAL"
    ]
}
```

What to do next

You can create and manage compute policies. See Create, List, and Use Policies.

Create, List, and Use Policies

You can create various types of policies, list all policies on a vCenter Server instance, list and filter tags used by policies.

Procedure

1 Create an `anti_affinity_with_vcls` policy.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/compute/policies
```

The following example `create_spec` specifies the policy details.

```
{
  "spec": {

"@class":"com.vmware.vcenter.compute.policies.capabilities.anti_affinity_with_vcls.create_s
pec",
    "name":"az_policy",
    "host_tag":"urn:vmomi:InventoryServiceTag:adf08d0c-a4e4-4c15-8798-c9a4d6873820:GLOBAL",
    "description":"az_policy description",
    "vm_tag":"urn:vmomi:InventoryServiceTag:adf08d0c-a4e4-4c15-8798-c9a4d6873820:GLOBAL"
  }
}
```

The following is an example response that contains the policy ID.

```
{
    "value": "178aaf62-06e7-4958-bff9-841674633a3f"
}
```

2. List all policies on a vCenter Server instance.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/compute/policies
```

The following is an example response that contains the available policies.

```
{
    "value": [
        {
            "capability": 
"com.vmware.vcenter.compute.policies.capabilities.anti_affinity_with_vcls",
            "name": "az_policy",
            "description": "az_policy description",
            "policy": "178aaf62-06e7-4958-bff9-841674633a3f"
        }
    ]
}
```

3. List tags used by one or all policies defined on a vCenter Server instance.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/compute/policies/tag-usage
```

The following is an example response that contains the available tags.

```
{
    "value": [
        {
            "category_name": "AZ_vms",
            "tag_type": "com.vmware.cis.tagging.Tag:VirtualMachine",
            "tag_name": "vm_us_west_a",
            "tag": "urn:vmomi:InventoryServiceTag:eefb3b11-afa2-4b44-a389-
e94065004548:GLOBAL",
            "policy": "178aaf62-06e7-4958-bff9-841674633a3f"
        },
        {
            "category_name": "AZ_hosts",
            "tag_type": "com.vmware.cis.tagging.Tag:HostSystem",
            "tag_name": "host_us_west_b",
            "tag": "urn:vmomi:InventoryServiceTag:5b3564f4-
a762-4480-9145-0f87d28aea78:GLOBAL",
            "policy": "178aaf62-06e7-4958-bff9-841674633a3f"
        }
    ]
}
```

4 Filter tags used by one or all policies defined on a vCenter Server instance by policy ID, tag UUID, and tag type.

```
GET https://<vcenter_ip_address_or_fqdn>/api/
vcenter/compute/policies/tag-usage?filter.policies=178aaf62-06e7-4958-
bff9-841674633a3f&filter.tags=urn:vmomi:InventoryServiceTag:5b3564f4-
a762-4480-9145-0f87d28aea78:GLOBAL&filter.tag_types=com.vmware.cis.tagging.Tag:HostSystem
```

The following is an example response that contains the filtered tag.

```
{
    "value": [
        {
            "category_name": VMC_AZ_hosts",
            "tag_type": "com.vmware.cis.tagging.Tag:HostSystem",
            "tag_name": "host_us_west_a",
            "tag": "urn:vmomi:InventoryServiceTag:5b3564f4-
a762-4480-9145-0f87d28aea78:GLOBAL",
            "policy": "178aaf62-06e7-4958-bff9-841674633a3f"
        }
    ]
}
```

vSphere Security

8

This chapter provides information about securing your vSphere environment for vCenter Server and ESXi.

Read the following topics next:

- Managing Certificates
- Managing vSphere Native Key Provider
- vSphere Trust Authority

Managing Certificates

Starting with vSphere 6.7 Update 2, you can use the vSphere Automation API to manage certificates in your vSphere environment. You can not only refresh default certificates that are issued by the VMware Certificate Authority (VMCA) but also add third-party or custom-made certificates to your environment.

Certificate Management Operations

Use the vSphere Automation API to manage trusted root certificate chains, VMware Certificate Authority (VMCA) root certificates, machine SSL (TLS) certificates, and Security Token Service (STS) signing certificates.

With the vSphere Automation API, you can refresh the VMCA-issued certificates but also add external and third-party certificates to your vSphere environment. For more information on vSphere certificate management, see the *vSphere Authentication* guide.

Certificate Management Interfaces

You can use the following interfaces to manage certificates with the vSphere Automation API:

- `Trusted Root Chains`
- `VMCA Root`
- `TLS CSR`
- `TLS`
- `Signing Certificate`

Certificate Management Operations

You can use the operations listed in the following table to manage certificates.

Table 8-1. Certificate Management Operations

Operation	Interface	HTTP Request	Description	Introduced in
List trusted root certificates	`Trusted Root Chains`	`GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/trusted-root-chains`	You can retrieve the identifiers of all trusted root certificates that are present in vCenter Server.	vSphere 6.7 U2
Get trusted root certificate information	`Trusted Root Chains`	`GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/trusted-root-chains/<chain>`	You can retrieve a root certificate chain by providing its identifier. You can retrieve the identifier by using the List trusted root certificates operation.	vSphere 6.7 U2
Add a trusted root certificate	`Trusted Root Chains`	`POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/trusted-root-chains`	Use this operation to add a trusted root certificate chain to your vCenter Server system.	vSphere 6.7 U2
Delete a trusted root certificate	`Trusted Root Chains`	`DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/trusted-root-chains/<chain>`	You can delete a root certificate by providing its unique identifier. You can retrieve the identifier by using the List trusted root certificates operation.	vSphere 6.7 U2
Replace the VMCA root certificate	`VMCA Root`	`POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/vmca-root`	You can replace the VMCA root certificate with a new VMCA-signed certificate. The operation triggers a restart of the services that are using this certificate.	vSphere 7.0
Generate a CSR	`TLS CSR`	`POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/tls-csr`	You can generate a CSR and use it to issue a custom certificate. If the operation is successful, you receive a CSR in PEM format.	vSphere 6.7 U2
Get the Machine SSL certificate	`TLS`	`GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/tls`	You can retrieve the Machine SSL certificate of your vCenter Server system.	vSphere 6.7 U2
Renew the Machine SSL certificate	`TLS`	`POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/tls?action=renew`	You can renew the validity of the machine SSL certificate for a specified period in days. The duration must be less than or equal to 730 days.	vSphere 6.7 U2

Table 8-1. Certificate Management Operations (continued)

Operation	Interface	HTTP Request	Description	Introduced in
Replace the Machine SSL certificate with a custom signed certificate	`TLS`	`PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/tls`	You can replace the vCenter Server Machine SSL certificate with a custom certificate signed by an external Certificate Authority (CA).	vSphere 6.7 U2
Replace the Machine SSL certificate with a VMCA-signed certificate	`TLS`	`POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/tls?action=replace-vmca-signed`	You can replace the vCenter Server Machine SSL certificate with a VMCA-signed certificate.	vSphere 7.0
Retrieve the STS signing certificate chains	`Signing Certificate`	`GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/signing-certificate`	You can retrieve the STS signing certificate chains, which are used for validating tokens signed by vCenter Server.	vSphere 7.0 U3
Replace the STS signing certificate	`Signing Certificate`	`PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/signing-certificate`	You can replace the current STS signing certificate with a certificate of your choice. The accepted file format is PEM.	vSphere 7.0 U3
Refresh the STS signing certificate	`Signing Certificate`	`POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/signing-certificate?action=refresh`	You can replace the current STS signing certificate with a new VMCA-signed certificate. The newly-generated certificate is set as the active STS signing certificate for the vCenter Server token service.	vSphere 7.0 U3

Add a Root Certificate to vCenter Server

You can use the `Certificate Management vCenter Trusted Root Chains` interface to add, delete and read trusted root certificate chains.

If you want to use an enterprise or third-party certificate authority (CA) for certificate management of your vSphere environment, you must first establish trust with that CA. You can do this by adding the root certificate of the external CA to the trusted root store of your vCenter Server system.

Adding a root certificate or certificate chain to the vCenter Server trusted certificate store establishes trust with an enterprise or third-party certificate authority. You can add a root certificate to vCenter Server as a prerequisite for other scenarios such as setting a third-party or enterprise machine SSL certificate.

Prerequisites

- Verify that you are connected to a vSphere Automation API server.
- Verify that the root certificate or certificate chain you want to add is available on your machine.
- Verify that you have the required privileges: **CertificateManagement.Manage** and **CertificateManagement.Administer**.

Procedure

1 (Optional) Retrieve the root certificates on your vCenter Server system.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
trusted-root-chains
```

2 Populate the `TrustedRootChains.CreateSpec` data structure.

Parameter	Type	Description
`cert_chain`	String	Certificate or certificate chain in base64 encoding. The input must be JSON string escaped for newline (\n).

3 Add the certificate or certificate chain.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
trusted-root-chains
```

If the operation is successful, the system returns the unique identifier of the trusted root certificate you added.

Delete a Root Certificate from vCenter Server

You can use the `Certificate Management vCenter Trusted Root Chains` interface to add, delete and read trusted root certificate chains. This use case demonstrates how to delete a root certificate or certificate chain from the trusted root store of your vCenter Server system.

Deleting certificates is not available through the vSphere Client and you can only do this by using the vSphere Automation API or the CLI tools.

Caution Deleting a root certificate or certificate chain that is in use might cause breakage of your systems. Proceed to delete a root certificate only if you are sure it is not in use by your vCenter Server or any connected systems.

Prerequisites

- Verify that you are connected to a vSphere Automation API server.
- Verify that you have the required privileges for executing the method: **CertificateManagement.Administer** and **CertificateManagement.Manage**.

Procedure

1 (Optional) Retrieve the root certificates from your vCenter Server system.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
trusted-root-chains
```

The system lists the unique identifiers (chains) of the certificates in the trusted root store.

2 Retrieve the certificate you want to delete.

To retrieve and verify a root certificate, use its unique identifier (chain).

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
trusted-root-chains/<chain>
```

3 Delete the certificate by providing its unique identifier (chain).

```
DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
trusted-root-chains/<chain>
```

The system returns a 204 error, which means that the request was processed but no content is returned.

4 (Optional) To verify you deleted the certificate, retrieve the root certificates from your vCenter Server system once again.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
trusted-root-chains
```

Change the Machine SSL Certificate of vCenter Server

You can change the machine SSL certificate of a vCenter Server system by using the TLS and the TLS CSR interfaces of the vSphere Automation API.

The machine SSL certificate is used for server verification and for secure communication such as HTTPS or LDAPS. The machine certificates are the human-facing certificates in vSphere. They are used to create an SSL socket on the server side to which SSL clients can then connect.

Changing the machine SSL certificate with one issued by an official or enterprise certificate authority is an essential part of the Hybrid Mode of vSphere certificate management. In this mode, you replace the machine SSL certificate and you leave the VMCA to manage all other certificates automatically. The VMCA is a just-enough internal certificate authority that comes integral with your vSphere deployment. It has been purpose-built to serve the certificate needs of your vSphere environment. For more information on vSphere certificate management, see the *vSphere Authentication* guide.

Prerequisites

- Verify that you are connected to a vSphere Automation API server.
- Verify that the root certificate of the CA you are going to use is available on your machine.
- Verify that you have the required privileges: **CertificateManagement.Administer** and **CertificateManagement.Manage**.

Procedure

1 (Optional) Retrieve the current Machine SSL certificate of your vCenter Server system.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/tls
```

2 Generate a certificate signing request (CSR) by using the `TLS CSR` functionality.

a Populate the `CertificateManagementVcenterTlsCsrSpec` data structure.

Parameter	Type	Description
`country`	String	Specifies the country in the certificate subject.
`state_or_province`	String	Specifies the state or province in the certificate subject.
`locality`	String	Specifies the locality in the certificate subject.
`organization`	String	Specifies the organization in the certificate subject.
`organization_unit`	String	Specifies the organization unit in the certificate subject.
`email_address`	String	Specifies the email address in the certificate subject.

b Make the request.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
tls-csr
```

The system returns the CSR in PEM format.

3 Save the CSR to your machine.

4 Send the CSR to the certificate authority of your choice.

Note The private key corresponding to the public key generated by the CSR is stored in the vCenter Server keystore and does not exit your system.

5 Save the issued Machine SSL certificate to your machine.

6 Set the new custom certificate to your vCenter Server system by using the `TLS` functionality.

a Populate the `CertificateManagementVcenterTlsSpec` data structure.

Parameter	Type	Description
`cert`	String	The Machine SSL certificate in PEM format. You must also paste the intermediate CA certificate, if you have one. The certificates must be JSON string escaped for newline (\n).
`root_cert`	String	The third-party root CA certificate in PEM format. You must also paste the intermediate CA certificate, if you have one. The certificates must be JSON string escaped for newline (\n).

Note You must not provide the private key as it was generated with the CSR and is already saved to your system.

b Make the request.

```
PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/tls
```

The system returns a 204 error, which means that the request was processed successfully but no content is returned.

Note Restart of vCenter Server services after the certificate change is not necessary. The certificate replacement is completed seamlessly and all your sessions remain active.

Refresh the vCenter Server STS Signing Certificate with a VMCA-Issued Certificate

You can refresh the vCenter Server Security Token Service (STS) signing certificate by using the `Certificate Management vCenter Signing Certificate` interface. The STS is an internal entity that issues and verifies tokens so that vSphere services can communicate with and trust each other.

You can refresh the current STS signing certificate of your vCenter Server system with a new VMCA-issued certificate.

There are two valid reasons for refreshing your STS signing certificate or certificate chain.

- If it is close to expiry. The standard lifespan of the vCenter Server STS signing certificate is 10 years. Your vCenter Server system will notify you in advance of STS certificate expiry. An alarm is triggered once per week when your STS certificate is 90 days away from expiry, and then daily when seven days away.
- If you already replaced your signing certificate with a third-party or enterprise one and now want to revert back to a default VMCA-issued certificate. This procedure replaces the custom or third-party STS signing certificates you added.

Prerequisites

- Verify that you are connected to a vSphere Automation API server.
- Verify that you have the **CertificateManagement.Administer** privilege.

Procedure

1 (Optional) Retrieve the current vCenter Server STS signing certificate chain.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
signing-certificate
```

2 Refresh the STS signing certificate.

a (Optional) Populate the `SigningCertificateRefreshRequestBody` data structure.

Parameter	Type	Description
`force`	boolean	The default is `false`. You can use this parameter to force a signing certificate refresh in environments that would otherwise prevent refresh from occurring such as mixed-version environments. Use `force` only when it is understood why the refresh fails or if you are instructed to do so by VMware® customer support.

b Request the refresh.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
signing-certificate?action=refresh
```

Results

If successful, the system returns the x509 certificate chain issued in accordance with the vCenter Server policies.

Caution If you used a forced refresh, you must restart your vCenter Server and all linked services.

Set a Custom STS Signing Certificate to vCenter Server

You can import and replace the vCenter Server STS signing certificate with a custom generated or third-party certificate by using the `Certificate Management vCenter Signing Certificate` interface.

In the usual case, you must not replace the vCenter Server STS signing certificate as it is not an external-facing certificate. The STS is an internal service that enables communication between various vSphere services. A fresh installation of vSphere 7.0 and later comes with a signing certificate that is issued with a default duration of 10 years. Replace the STS signing certificate with a custom or third-party certificate only if your company security policy requires you to do so.

Prerequisites

- Verify that you are connected to a vSphere Automation API server.
- Verify that the custom generated or third-party certificate chain and private key are available on your machine.
- Verify that you have the **CertificateManagement.Administer** privilege.

Procedure

1 Populate the `SigningCertificateSetSpec` data structure.

   a Populate the correspoding unencrypted PKCS#8 private key as a string in base64-encoded PEM format. It must be JSON string escaped for newline(\n).

   b Populate the `X509CertChain` data structure.

Parameter	Type	Description
cert_chain	String	The custom generated or third-party certificate chain in base64-encoded PEM format. It must be a valid certificate chain with the leaf cert marked for Digital Signature key usage. The leaf certificate must be first in the sequence and the root must be last. The certificates must be JSON string escaped for newline(\n).

2 Set the STS signing certificate.

```
PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/certificate-management/vcenter/
signing-certificate
```

The system returns a 204 error, which means that the request was processed successfully but no content is returned.

Results

Caution The change of the STS signing certificate might leave systems in the local vCenter Server domain in a non-functional state. To prevent system failure, restart your vCenter Server instance and all linked services.

Managing vSphere Native Key Provider

You can use the vSphere Automation REST API to perform VMware vSphere® Native Key Provider™ operations.

In vSphere 7.0 Update 2 and later, you can use the built-in vSphere Native Key Provider to enable encryption technologies, such as virtual TPMs (vTPM).

vSphere Native Key Provider is included in all vSphere editions and does not require an external key server (also called a Key Management Server (KMS) in the industry). You can also use vSphere Native Key Provider for vSphere Virtual Machine Encryption, but you must purchase the VMware vSphere® Enterprise Plus Edition™.

For details about vSphere Native Key Provider, see the *vSphere Security* documentation.

The procedures in this chapter are based on the REST API. For details, see the *vSphere Automation REST API Reference*.

Retrieve vSphere Native Key Providers

You can use HTTP requests to list vSphere Native Key Providers and retrieve details about a specific vSphere Native Key Provider.

Procedure

1 Retrieve a list of vSphere Native Key Providers.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/kms/providers
```

In the response body, you receive information about the health status, ID, and type of each provider.

2 Retrieve information about a vSphere Native Key Provider.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/kms/providers/
<provider>
```

In the response body, you receive detailed information about the health status, the type of the provider, and any constraints on the provider.

Manage vSphere Native Key Providers

You can use HTTP requests to perform vSphere Native Key Provider management operations.

You can add, update, and remove vSphere Native Key Providers. Some operations require you to specify parameters in the body of the HTTP request according to your environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Procedure

1 Add a vSphere Native Key Provider by specifying the provider ID in the request body.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/kms/providers
```

The following request body contains example syntax.

```
{
    "native_spec": {
    },
    "provider": Test_Provider
}
```

2 Update a vSphere Native Key Provider by specifying the new provider configuration in the request body.

```
PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/kms/providers/
<provider>
```

3 Remove a vSphere Native Key Provider.

```
DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/kms/providers/
<provider>
```

Export and Import vSphere Native Key Provider Configuration

You can use HTTP requests to export and import vSphere Native Key Provider configuration.

You can set a vSphere Native Key Provider as the default for a vCenter Server instance by using the `SetDefaultKmsCluster` method of the `CryptoManagerKmip` managed object from the vSphere Management SDK.

The operations require you to specify parameters in the body of the HTTP request according to your environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Procedure

1 Export a vSphere Native Key Provider configuration by specifying the provider you want to export in the request body.

Note You must export the vSphere Native Key Provider configuration at least once before you can use a vSphere Native Key Provider.

a Initiate the export by providing the name identifier of the key provider and a password to encrypt the export data in the request body.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/kms/providers?
action=export
```

The following request body contains example syntax.

```
{
    "password": <your_password>,
    "provider": Test_Provider
}
```

In the response body, you receive the URL of the exported configuration and an authentication token. The following response body contains example syntax.

```
{
    "location": {
        "download_token": {
            "expiry": 2023-09-20T21:04:50.000Z,
            "token": eyJhbGciOiJIUzI1NjKXn21y01CM
        },
        "url": https://vc8.home/cryptomanager/kms/test
    },
    "type": LOCATION
}
```

b Send an HTTP request to the URL with the authentication token to fetch the exported PKCS12 data by providing the URL and token values received in the response body of the previous step.

```
POST <url>
Authorization: Bearer <token>
```

The following example HTTP request contains the values from the example response in the previous step.

```
POST https://vc8.home/cryptomanager/kms/test
Authorization: Bearer eyJhbGciOiJIUzI1NjKXn21y01CM
```

2. Import a vSphere Native Key Provider configuration by specifying the configuration you want to import in the request body.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/kms/providers?
   action=import
   ```

   The following request body contains example syntax.

   Note You must provide a Base64-encoded string that is a representation of the PKCS12 data exported in the previous step as the value of the `config` parameter. The other parameters are optional.

   ```
   {
       "config": <string>,
       "constraints": {
           "tpm_required": false
       },
       "dry_run": false,
       "password": <your_password>
   }
   ```

   In the response body, you receive information about the imported provider and the time when that provider was exported.

Retrieve vSphere Native Key Providers on a Host

You can use HTTP requests to list vSphere Native Key Providers on a host and retrieve details about a specific vSphere Native Key Provider.

Procedure

1. Retrieve a list of vSphere Native Key Providers on a host.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/hosts/<host>/kms/
   providers
   ```

   In the response body, you receive information about the health status, ID, and type of each provider.

2. Retrieve information about a vSphere Native Key Provider on a host.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/crypto-manager/hosts/<host>/kms/
   providers/<provider>
   ```

   In the response body, you receive detailed information about the health status and the type of the provider.

vSphere Trust Authority

You can use the vSphere Automation REST API to perform vSphere Trust Authority operations.

vSphere Trust Authority is a foundational technology that enhances workload security. vSphere Trust Authority establishes a greater level of trust in your organization by associating an ESXi host hardware root of trust to the workload itself. For details about vSphere Trust Authority, see the *vSphere Security* documentation.

The procedures in this chapter are based on the REST API. For details, see the *vSphere Automation REST API Reference*.

Configure a vSphere Trust Authority Cluster

You can use HTTP requests to perform vSphere Trust Authority Cluster management operations.

You can retrieve details about vSphere Trust Authority Clusters, update the state of a cluster, and check the result of the update operation. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1. Retrieve a list of clusters for a vCenter Server instance that are configured as Trust Authority Clusters.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters
   ```

   You receive the list in the response body.

2. Update the state of a cluster.

   ```
   PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

3. Check the result of the last update operation for the same cluster.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>
   ```

   You receive the ID and current state of the cluster in the response body.

Configure Key Providers

You can use HTTP requests to perform Key Provider management operations.

You can retrieve, add, update, remove, and retrieve details about Key Providers. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 Retrieve a list of Key Providers to see which Key Providers the cluster is using.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/kms/providers?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

2 Add a new Key Provider which all hosts in the cluster can use.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/kms/providers?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

3 Retrieve information about a Key Provider to verify the configuration.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/kms/providers/<provider>?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

4 Update an existing Key Provider to modify the connection details and primary key for it.

```
PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/kms/providers/<provider>?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

5. Remove a Key Provider.

```
DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/kms/providers/<provider>?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation completes successfully, the cluster can no longer use that Key Provider.

Establish Trust Between Key Provider and Key Server

You can use HTTP requests to perform trust management operations.

You can list and update server certificates, retrieve, generate, and update client certificates, generate a CSR, and set the key server credential. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1. List the remote server certificates on the configured key servers to verify the trusted key servers.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/
trusted-infrastructure/trust-authority-clusters/<cluster>/kms/providers/<provider>/peer-
certs/current?server_names=<value-1>&server_names=<value-2>&trusted=<true>&vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

2. Retrieve the list of trusted server certificates.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/kms/providers/<provider>/peer-certs/trusted?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

3. Update the trusted server certificates.

   Note This operation overwrites the existing list of trusted certificates.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/kms/providers/<provider>/peer-certs/trusted?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

4. Retrieve the existing client certificate.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/kms/providers/<provider>/client-certificate?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

   If the operation is successful, you receive the client certificate in PEM format.

5. Generate a new self-signed client certificate, used to establish a secure connection to the key server.

   Note This operation overwrites the existing client certificate.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/kms/providers/<provider>/client-certificate?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

   If the operation is successful, you can provide the newly generated self-signed client certificate to the key server to establish trust with the Key Provider.

6 Update the client certificate to specify what Key Provider should use to authenticate with the key server.

Note If a client certificate exists, this operation overwrites it.

```
PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-authority-clusters/<cluster>/kms/providers/<provider>/client-certificate?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

7 Generate a certificate signing request (CSR) for the client certificate.

Note If a CSR exists, this operation overwrites it.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-authority-clusters/<cluster>/kms/providers/<provider>/client-certificate/csr?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, you receive the client CSR in PEM format and the host ID which issued it. The generated CSR can later be signed by a third party. The signed CSR should be replicated and set on each host.

8 Set the key server credential for key servers that require a password.

```
PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-authority-clusters/<cluster>/kms/providers/<provider>/credential?vmw-task=true

"secret string"
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

Configure Trusted TPMs of Attested ESXi Hosts on a Cluster Level

You can use HTTP requests to manage remote attestation configuration for TPM trust.

You can add, list, remove, and retrieve details about TPM CA certificates and TPM endorsement keys. You can also set and retrieve TPM 2.0 attestation settings. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 Add a new TPM CA certificate to a Trusted Cluster to specify a trusted platform OEM.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/attestation/tpm2/ca-certificates?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

2 Retrieve a list of configured TPM CA certificates on a Trusted Cluster to identify the trusted platform OEMs.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/attestation/tpm2/ca-certificates?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

3 Remove a TPM CA certificate from a Trusted Cluster because a platform OEM is no longer trusted.

```
DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/attestation/tpm2/ca-certificates/<name>?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

4 Retrieve details about a specific TPM CA certificate on a Trusted Cluster to get more information about the trusted platform OEM.

```
GET https://<vcenter_ip_address_or_fqdn>//api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/attestation/tpm2/ca-certificates/<name>?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

5. Add a new TPM endorsement key to a Trusted Cluster to specify a trusted ESXi host.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/tpm2/endorsement-keys?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

6. Retrieve a list of configured TPM endorsement keys in a Trusted Cluster to identify the trusted ESXi hosts.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/tpm2/endorsement-keys?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

7. Remove a TPM endorsement key from a Trusted Cluster because an ESXi host is no longer trusted.

   ```
   DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/tpm2/endorsement-keys/<name>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

8. Retrieve details about a specific TPM endorsement key on a Trusted Cluster to get more information about the trusted ESXi host.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/tpm2/endorsement-keys/<name>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

9. Set the TPM 2.0 attestation settings by specifying that TPM endorsement keys on a Trusted Cluster do not need to be signed because the trusted OEM does not sign endorsement keys.

   ```
   PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/tpm2/settings?vmw-task=true
   ```

```
{
    "require_endorsement_keys" : false,
    "require_certificate_validation" : true
}
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

10. Determine the TPM 2.0 attestation settings in a Trusted Cluster.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-clusters/<cluster>/attestation/tpm2/settings?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

Configure Trusted ESXi Builds on a Cluster Level

You can use HTTP requests to manage trusted instances of ESXi software on a cluster level.

You can import, list, remove, and retrieve details about ESXi base images. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1. Import ESXi metadata as a new trusted base image to each host in a vSphere Trust Authority Cluster.

```
POST https://<vcenter_ip_address_or_fqdn>/api/
vcenter/trusted-infrastructure/trust-authority-clusters/<cluster>/attestation/os/esx/base-
images?action=import-from-imgdb&vmw-task=true

"YmluYXJ5"
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

2. Retrieve a list of trusted ESXi base images in a vSphere Trust Authority Cluster.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/os/esx/base-images?
   version=<value-1>&version=<value-2>&display_name=<value-1>&display_name=<value-2>&health=<v
   alue-1>&health=<value-2>&vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

3. Remove an ESXi base image that should no longer be trusted from a vSphere Trust Authority Cluster.

   ```
   DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/os/esx/base-images/<version>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

4. Retrieve details about a trusted ESXi base image version in a vSphere Trust Authority Cluster.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/attestation/os/esx/base-images/<version>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

Retrieve vSphere Trust Authority Components Information

You can use HTTP requests to retrieve information about Attestation Service and Key Provider Service instances running on hosts.

You can use the retrieved information to connect to the hosts running the vSphere Trust Authority components. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 Retrieve detailed information, including the certificates, about the Attestation Service instance running on a Trust Authority Host.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-hosts/<host>/attestation/
```

You receive the details in the response body. You can use the retrieved information to import the Attestation Service instance into a Workload vCenter Server.

2 List Trust Authority Hosts running an Attestation Service instance by using filters.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-hosts/attestation?projection=<value>
```

You receive the results that match your criteria in the response body. You can use the retrieved information to review the Attestation Service instances.

3 Retrieve detailed information, including the certificates, about the Key Provider Service instance running on a Trust Authority Host.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-hosts/<host>/kms/
```

You receive the details in the response body. You can use the retrieved information to import the Key Provider Service instance into a Workload vCenter Server.

4 List Trust Authority Hosts running a Key Provider Service instance by using filters.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
authority-hosts/attestation?projection=<value>kms?projection=<value>&action=query
```

You receive the results that match your criteria in the response body. You can use the retrieved information to review the Key Provider Service instances.

Configure vSphere Trust Authority Components

You can use HTTP requests to perform Key Provider Service and Attestation Service management operations.

You can register, list, remove, and retrieve details about Key Provider Service and Attestation Service instances. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 Register a Key Provider Service instance in a Workload vCenter Server.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/kms/services
```

The Key Provider Service instance is propagated to all Workload ESXi hosts that the Workload vCenter Server manages.

2 Register an Attestation Service instance in a Workload vCenter Server.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/attestation/
services
```

The Attestation Service instance is propagated to all Workload ESXi hosts that the Workload vCenter Server manages.

3 List Key Provider Service instances registered in a Workload vCenter Server by using filters.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/kms/services?
action=query
```

You receive the results that match your criteria in the response body. You can use the filtered list to retrieve the health status of the Key Provider Service instances.

4 List Attestation Service instances registered in a Workload vCenter Server by using filters.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/attestation/
services?action=query
```

You receive the results that match your criteria in the response body. You can use the filtered list to retrieve the health status of the Attestation Service instances.

5 Remove a registered Key Provider Service instance.

```
DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/kms/
services/<service>
```

The Workload ESXi hosts can no longer retrieve keys by using that Key Provider Service instance.

6 Remove a registered Attestation Service instance.

```
DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/attestation/
services/<service>
```

The Workload ESXi hosts can no longer attest that their configuration is secure by using that Attestation Service instance.

7 Retrieve detailed information, including the certificates, for a registered Key Provider Service instance.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/kms/services/
<service>
```

You receive the details in the response body. You can use the retrieved information to verify the Key Provider Service instance.

8 Retrieve detailed information, including the certificates, for a registered Attestation Service instance.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/attestation/
services/<service>
```

You receive the details in the response body. You can use the retrieved information to verify the Attestation Service instance.

Configure vSphere Trust Authority Components for Trusted Clusters

You can use HTTP requests to manage Key Provider Service and Attestation Service instances that a Trusted Cluster is configured to use.

You can configure, list, remove, and retrieve details about Key Provider Service and Attestation Service instances. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 Configure a cluster in a Workload vCenter Server to use a registered Key Provider Service instance.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
clusters/<cluster>/kms/services?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, the Key Provider Service instance is propagated to all Trusted ESXi hosts in the cluster.

2. Configure a cluster in a Workload vCenter Server to use a registered Attestation Service instance.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/attestation/services?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

   If the operation is successful, the Attestation Service instance is propagated to all Trusted ESXi hosts in the cluster.

3. List Key Provider Service instances used by a cluster by using filters.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/kms/services?action=query
   ```

   You receive the results that match your criteria in the response body. You can use the filtered list to retrieve the health status of the Key Provider Service instances.

4. List Attestation Service instances used by a cluster by using filters.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/attestation/services?action=query
   ```

   You receive the results that match your criteria in the response body. You can use the filtered list to retrieve the health status of the Attestation Service instances.

5. Remove a Key Provider Service instance from the configuration of a Trusted Cluster.

   ```
   DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/kms/services/<service>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

   If the operation is successful, the Trusted ESXi hosts can no longer retrieve keys by using that Key Provider Service instance.

6. Remove a registered Attestation Service instance from the configuration of a Trusted Cluster.

   ```
   DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/attestation/services/<service>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

If the operation is successful, the Trusted ESXi hosts can no longer attest that their configuration is secure by using that Attestation Service instance.

7. Retrieve detailed information, including the certificates, for a configured Key Provider Service instance used by a Trusted Cluster.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-clusters/<cluster>/kms/services/<service>
   ```

   You receive the details in the response body. You can use the retrieved information to verify the Key Provider Service instance.

8. Retrieve detailed information, including the certificates, for a registered Attestation Service instance used by a Trusted Cluster.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-clusters/<cluster>/attestation/services/<service>
   ```

   You receive the details in the response body. You can use the retrieved information to verify the Attestation Service instance.

Establish Trust Between Hosts in a vSphere Trust Authority Cluster and a Workload vCenter Server

You can use HTTP requests to perform trust management operations.

You can establish and remove trust between a Workload vCenter Server and the hosts in a vSphere Trust Authority Cluster. You can also list all Workload vCenter Server instances that have established trust with the host in a vSphere Trust Authority Cluster. Some operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1. Establish trust between a vSphere Trust Authority Cluster and a Workload vCenter Server by creating a profile, so that the Workload vCenter Server can retrieve the health status of the vSphere Trust Authority components.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-authority-clusters/<cluster>/consumer-principals?vmw-task=true
   ```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

2. Remove the trust between a Workload vCenter Server and the hosts in the vSphere Trust Authority Cluster, so that the Workload vCenter Server stops using the hosts for attestation.

   ```
   DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/consumer-principals/<profile>?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

3. List all profiles which the vSphere Trust Authority Cluster trusts.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trust-
   authority-clusters/<cluster>/consumer-principals?action=query&vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

Check Trusted Cluster Health

You can use HTTP requests to retrieve information about the health of the applied vSphere Trust Authority component configurations in a Trusted Cluster.

You can retrieve basic and detailed information about the health of Key Provider Service or Attestation Service configurations applied to a Trusted Cluster with respect to the desired state. You can also retrieve detailed information about the health of all applied vSphere Trust Authority component configurations in a Trusted Cluster. The operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1. Retrieve a summary about the health status of all Key Provider Service instances configured for use in a Trusted Cluster.

   ```
   POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/kms/services-applied-config?action=query&vmw-task=true
   ```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, you can verify whether all Key Provider Service configurations of the Trusted Cluster are applied successfully and every host in the cluster is consistent with the desired state.

2 Retrieve detailed information about the health status of a specific Key Provider Service instance configured for use in a Trusted Cluster.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
clusters/<cluster>/kms/services-applied-config?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, you can verify whether the specified Key Provider Service configuration of the Trusted Cluster is applied successfully and every host in the cluster is consistent with the desired state.

3 Retrieve a summary about the health status of all Attestation Service instances configured for use in a Trusted Cluster.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
clusters/<cluster>/attestation/services-applied-config?action=query&vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, you can verify whether all Attestation Service configurations of the Trusted Cluster are applied successfully and every host in the cluster is consistent with the desired state.

4 Retrieve detailed information about the health status of a specific Attestation Service instance configured for use in a Trusted Cluster.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
clusters/<cluster>/attestation/services-applied-config?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, you can verify whether the specified Attestation Service configuration of the Trusted Cluster is applied successfully and every host in the cluster is consistent with the desired state.

5 Retrieve detailed information about the health status of all vSphere Trust Authority components configured for use in a Trusted Cluster.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
clusters/<cluster>/services-applied-config?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, you can verify whether the vSphere Trust Authority component configuration is applied successfully and every host in the cluster is consistent with the desired state.

What to do next

If there are errors, you can try to remediate the Trusted Cluster. See Remediate a Trusted Cluster.

Remediate a Trusted Cluster

You can use HTTP requests to remediate vSphere Trust Authority component configurations in a Trusted Cluster or remove the configurations.

You can update the applied Key Provider Service or Attestation Service configurations in a Trusted Cluster to become consistent with the desired state or you can remove the applied Key Provider Service or Attestation Service configurations. You can also update all applied vSphere Trust Authority component configurations in a Trusted Cluster or remove the configurations. By removing the configurations, you can move hosts from a Trusted Cluster to another cluster. The operations require you to specify parameters in the body of the HTTP request according to your vSphere Trust Authority environment. For details about the syntax of each HTTP request body, see the *API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 Remediate all Key Provider Service instances configured for use in a Trusted Cluster.

```
PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
clusters/<cluster>/kms/services-applied-config?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, the Key Provider Service configuration of every host in the cluster is consistent with the desired state.

2. Remove all Key Provider Service configurations from a Trusted Cluster.

   ```
   DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/kms/services-applied-config?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

   If the operation is successful, the applied Key Provider Service configurations are removed from the configuration of every host in the cluster without affecting the desired state.

3. Remediate all Attestation Service instances configured for use in a Trusted Cluster.

   ```
   PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/attestation/services-applied-config?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

   If the operation is successful, the Attestation Service configuration of every host in the cluster is consistent with the desired state.

4. Remove all Attestation Service configurations from a Trusted Cluster.

   ```
   DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/attestation/services-applied-config?vmw-task=true
   ```

   You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

   ```
   GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
   ```

   If the operation is successful, the applied Attestation Service configurations are removed from the configuration of every host in the cluster without affecting the desired state.

5. Remediate all vSphere Trust Authority components configured for use in a Trusted Cluster.

   ```
   PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
   clusters/<cluster>/services-applied-config?vmw-task=true
   ```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, the vSphere Trust Authority component configuration of every host in the cluster is consistent with the desired state.

6 Remove all vSphere Trust Authority component configurations from a Trusted Cluster.

```
DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/trusted-
clusters/<cluster>/services-applied-config?vmw-task=true
```

You receive the task ID in the response body. You can use the task ID to check the status of the task by running the following HTTP request.

```
GET https://<vcenter_ip_address_or_fqdn>/api/cis/tasks/<task_ID>
```

If the operation is successful, the applied vSphere Trust Authority component configurations are removed from the configuration of every host in the cluster without affecting the desired state.

What to do next

You can recheck the Trusted Cluster health after the remediation. See Check Trusted Cluster Health.

Retrieve Host Hardware TPM Information

You can use HTTP requests to retrieve a list of configured TPM devices on a host and information about each TPM device.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 List configured TPM devices on a host.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/hosts/<host>/
hardware/tpm
```

You receive the results in the response body. You can use the retrieved information to review the basic information of the available TPM devices.

2 Retrieve detailed information about a specific TPM device, including the manufacturer, model, and firmware version.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/hosts/<host>/
hardware/tpm/<tpm>
```

You receive the results in the response body. You can use the retrieved information to review the detailed information of the TPM device.

Manage Host Hardware TPM Endorsement Keys

You can use HTTP requests to retrieve a list of configured TPM endorsement keys on a host and information about each endorsement key. You can also retrieve the TPM event log and unseal a secret that is bound to an endorsement key.

You can retrieve the TPM event log for different purposes, such as configuring firmware trust with an attestation service or validating the boot time TPM measurements. You can unseal a secret that is bound to an endorsement key to verify reported measurements. For example, you can verify measurements from the TPM event log. For details about the unsealing operation, see the *REST API Reference* documentation.

Prerequisites

- Verify that you have access to a working vSphere Trust Authority environment.
- Verify that you have Trusted Infrastructure administrative privileges.

Procedure

1 List configured TPM endorsement keys on a host.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/hosts/<host>/
hardware/tpm/<tpm>/endorsement-keys
```

You receive the results in the response body. You can use the retrieved information to review the basic information of the available TPM endorsement keys.

2 Retrieve detailed information about a specific TPM endorsement key.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/hosts/<host>/
hardware/tpm/<tpm>/endorsement-keys/<key>
```

You receive the results in the response body. You can use the retrieved information to review the detailed information of the TPM endorsement key.

3 Retrieve the event log associated with a TPM device.

```
GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/hosts/<host>/
hardware/tpm/<tpm>/event-log
```

You receive the information in the response body.

4 Unseal a secret that is bound to an endorsement key.

```
POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/trusted-infrastructure/hosts/<host>/
hardware/tpm/<tpm>/endorsement-keys/<key>?action=unseal
```

You receive a string that contains the unsealed secret.

vSphere with Tanzu Configuration and Management

9

Starting with vSphere 7.0, you can use the vSphere Automation REST APIs to enable and configure vSphere with Tanzu on a vSphere cluster. Then you can run Kubernetes workloads directly on ESXi hosts.

You can use the vSphere Automation to automate the enabling and disabling of vSphere with Tanzu on a vSphere cluster, creating and managing namespaces.

The information in this chapter is intended for vSphere administrators who want to use the vSphere Automation REST APIs to configure their environment to run Kubernetes workloads in vSphere. To take fully advantage of these vSphere Automation REST APIs, you must have basic knowledge about the Kubernetes technology and containers.

For more information about how to configure and manage vSphere with Tanzu through the vSphere Client, see the *Installing and Configuring vSphere with Tanzu* and *vSphere with Tanzu Configuration and Management* documentation.

Read the following topics next:

vSphere with Tanzu Terminology

You must understand the basic terminology in this chapter to be able use the vSphere with Tanzu automation APIs effectively.

vSphere with Tanzu Basic Terms

Term	Description
Supervisor	A vSphere cluster that has the vSphere with Tanzu enabled.
Tanzu Kubernetes cluster	An upstream Kubernetes cluster provisioned and managed by using the VMware Tanzu™ Kubernetes Grid™. A Tanzu Kubernetes resides in a vSphere Namespace. You can deploy workloads and services to such clusters in the same way as you do with standard Supervisor.
vSphere Namespace	A namespace that is created within a Supervisor. Each namespace sets the resource boundaries for CPU, memory, storage, and also the number of Kubernetes objects that can run within the namespace. After a namespace is configured, you can run Kubernetes workloads within the namespace.
vSphere Pod	A virtual machine with a small footprint that runs one or more Linux containers. A vSphere Pod is equivalent to a Kubernetes pod. vSphere Pods are compatible with the Open Container Initiative (OCI) and can run OCI compatible containers regardless the operating system.
Spherelet	A spherelet is an implementation of the kubelet functionality ported natively on each host in the Supervisor.
Kubernetes Workload	Workloads are applications that are deployed in one of the following ways: ■ As containers running inside vSphere Pods. ■ Workloads provisioned through the VM service. ■ Tanzu Kubernetes Gridclusters deployed by using the Tanzu Kubernetes Grid. ■ Applications running inside the Tanzu Kubernetes Grid clusters.
Supervisor control plane	vSphere with Tanzu creates a Kubernetes control plane directly on the hypervisor layer. The control plane manages the worker nodes and the vSphere Pods in the Supervisor.
Supervisor worker nodes	ESXi hosts that are part of a Supervisor are considered as worker nodes. You run your Kubernetes workloads on the worker nodes.
Container Runtime Executive (CRX)	CRX is an isolated Linux execution environment similar to a VM that works together with ESXi.
VM Service	The VM Service functionality allows DevOps engineers to deploy and manage virtual machines in their Kubernetes environment through standard Kubernetes APIs. vSphere administrators are responsible for providing VM Classes and VM Images for the DevOps engineers to choose from, as well as managing resource allocations to self-service provisioned VMs.

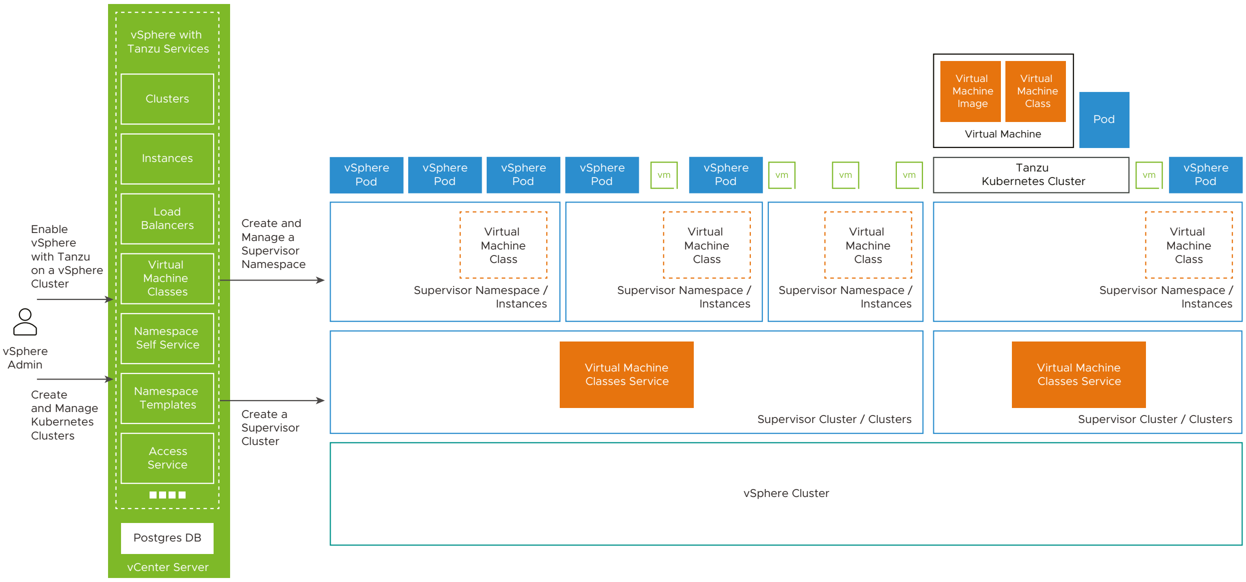
Term	Description
Self-Service Namespace	vSphere administrators can activate the Self-Service Namespace service on a Supervisor and create namespace templates for DevOps engineers to create a vSphere Namespace themselves.
vSphere Zones	vSphere Zones provide high availability against clusters-level failures to workloads deployed on vSphere with Tanzu. You can configure a three-zone Supervisor mapped to three vSphere clusters or a one-zone Supervisor mapped to a single vSphere cluster. In a single cluster deployment, the high availability is provided by vSphere HA and is only on a host level.

vSphere with Tanzu Components and Services

Before you can automate some of the administrative tasks for using vSphere with Tanzu, you must first familiarize yourself with the high-level system architecture and components involved.

The vSphere with Tanzu API consists of two packages, `namespace_management` and `namespaces`. In the `namespace_management` package, you can find APIs for enabling a vSphere cluster with vSphere with Tanzu, configuring the network and storage policies of the Supervisor, upgrading a cluster to the desired version of vSphere with Tanzu, and so on. In the `namespaces` package, you can find APIs for creating, configuring, and deleting a vSphere Namespace, and also for setting the necessary permissions for accessing the namespace.

Figure 9-1. Services and Components Involved in Using vSphere with Tanzu



The vSphere Kubernetes Services component runs on vCenter Server and communicates the vSphere admin requests to the Supervisor control plane. The component comprises of several services which vSphere Automation endpoints you can use to enable vSphere with Tanzu on a vSphere cluster and create Kubernetes workloads.

You can use the Cluster Compatibility service to query a vCenter Server instance about the available clusters that meet the requirements for enabling vSphere with Tanzu.

You can use the Clusters service to enable or disable vSphere with Tanzu on a cluster. You can also reconfigure the settings of a Supervisor.

You can use the Instances service to create, edit, and delete a vSphere Namespace from a Supervisor. You can also change all or some of the settings of an existing namespace.

Starting with vSphere 7.0 Update 1, a Supervisor backed by a vSphere Distributed Switch uses the HAProxy load balancer to provide connectivity to DevOps and external service. The Load Balancer service represents the user provisioned load balancers.

Starting with vSphere 7.0 Update 2a, vSphere administrators can use the VM Service functionality to enable DevOps engineers to deploy and run VMs and containers in one shared Kubernetes environment through a single Kubernetes native interface. Use the vSphere with Tanzu APIs to define VM Classes and content libraries to allocate resources to virtual machines provisioned by DevOps engineers.

As of vSphere 7.0 Update 2a, vSphere administrators can also configure a vSphere Namespace as a template on a cluster. Then the DevOps engineers can use it to self-service the creation of vSphere Namespaces and deploy workloads within them.

How to Configure and Manage a Supervisor

You use the `Clusters` service to enable and disable a Supervisor, or edit the configuration of an existing Supervisor. The `Clusters` service is provided within the `namespace_management` package.

You can enable a vSphere cluster to manage Kubernetes workload objects, only after you enable vSphere DRS in a fully automated mode and enable HA on the cluster.

Before you enable a vSphere with Tanzu on a vSphere cluster, you must prepare your environment to meet the specific networking, storage, and infrastructure requirements. See the *Installing and Configuring vSphere with Tanzu* documentation.

For more information about how to configure the storage settings to meet the requirements of vSphere with Tanzu, see Creating Storage Policies for vSphere with Tanzu.

For more information about how to configure the networking settings for Supervisors that are configured with the VMware NSX-T™ Data Center as the networking stack, see Configuring NSX for vSphere with Tanzu.

Starting with vSphere 7.0 Update 1, you can enable a Supervisor with vSphere networking or NSX-T Data Center, to provide connectivity between control planes, services, and workloads. A Supervisor that is configured with vSphere networking uses a vSphere Distributed Switch to provide connectivity to Kubernetes workloads and control planes. The cluster also requires a third-party load balancer that provides connectivity to DevOps users and external services. You can install in your vSphere environment the HAProxy load balancer implementation that VMware provides. See Configuring the vSphere Networking Stack for vSphere with Tanzu and Installing and Configuring the HAProxy Load Balancer.

Staring with vSphere 7.0 Update 2, if you are using vSphere networking, you can use the VMware NSX® Advanced Load Balancer™ to support Tanzu Kubernetes clusters provisioned by the Tanzu Kubernetes Grid. See Using the NSX Advanced Load Balancer with vSphere Networking.

Persistent Storage in vSphere with Tanzu

Some Kubernetes workloads use persistent storage to store data permanently. vSphere with Tanzu integrates with Cloud Native Storage (CNS) to provision persistent storage.

To understand how vSphere with Tanzu uses persistent storage, refer to the *vSphere with Tanzu Concepts and Planning* and *vSphere Storage* documentations.

Creating Storage Policies for vSphere with Tanzu

Before you enable vSphere with Tanzu, you must set up the storage to provision the Kubernetes infrastructure. You achieve this task by creating storage policies to be used in the Supervisor and namespaces.

To automate the creation of a tag-based storage policy, use the VMware® vSphere Management SDK. For more information about how to create a tag-based storage policy through the Web Services API, see the *VMware Storage Policy SDK Programming Guide* and *vSphere Web Services SDK Programming Guide* documentations.

Optionally, you can use the vSphere Automation REST APIs to create and add a tag to the datastore. See the Chapter 7 vSphere Tag Service chapter. Currently, you can create a tag-based storage policy only through the Web Services APIs.

Use the vSphere Automation REST APIs to retrieve the default storage policy of a specific datastore by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/datastore/datastore_id/default-policy` request and submitting the ID of the datastore as a path parameter. You can also retrieve commonly used information about the storage policies available in the vCenter Server instance by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/compute/policies` request.

You can use the storage policies retrieved through the vSphere Automation APIs to perform the following tasks:

- Assign the storage policies to the Supervisor. The storage policies set within the Supervisor enable specification ensure that the Supervisor control plane, the ephemeral disks of all vSphere Pods, and the container images are placed on the datastores that the policies represent. See Configuring NSX for vSphere with Tanzu.
- Assign the storage policies to the vSphere Namespace. The storage policies associated with a namespace determine which datastores the namespace can access and use for persistent volumes for the vSphere Pod and the pods inside a Tanzu Kubernetes cluster. See Create a vSphere Namespace.

Enabling ReadWriteMany Support

You can enable the ReadWriteMany support in vSphere with Tanzu and allow multiple pods and applications to mount simultaneously a single persistent volume.

In vSphere 7.0 Update 3, only Tanzu Kubernetes clusters support persistent volumes in ReadWriteMany mode. When you enable file volume support for vSphere with Tanzu, be aware of the potential security weaknesses:

- The volumes are mounted without encryption. The unencrypted data might be accessed while the data transits the network.
- Access Control List (ACL) is used for the file shares to isolate file share access within a supervisor namespace. It might have risk of IP spoofing.

Follow these guidelines for networking:

- Make sure the vSAN File Services is routable from the Workload network and there is no NAT between the Workload network and vSAN File Services IP addresses.
- Use common DNS server for vSAN File Services and the vSphere cluster.
- If your vSphere with Tanzu has NSX networking, use the SNAT IP of the Supervisor namespace and the SNAT IP of the Tanzu Kubernetes cluster for ACL configuration.
- If you have vSphere with Tanzu with vSphere Distributed Switch (VDS) networking, use the Tanzu Kubernetes cluster VM IP or the IP of the Supervisor namespace for ACL configuration.

Before you activate the file volume support on a Supervisor, you must set up a vSAN cluster with enabled vSAN File Service. To configure a vSAN cluster with enabled vSAN File Service in the vSphere Client, see the *Configure File Services* topic in the *Administering VMware vSAN* documentation. For more information about how to programmatically achieve this task, see the *vSAN SDKs Programming Guide* documentation.

You activate the ReadWriteMany support on a cluster when you enable vSphere with Tanzu on it, or reconfigure an existing Supervisor. See Enable vSphere with Tanzu on a Cluster with NSX as the Networking Stack, Enable vSphere with Tanzu on a Cluster with the vSphere Networking Stack, and Reconfiguring a Supervisor. Pass the list of vSAN clusters to be used for provisioning file volumes by using the `cns_file_config` property of respective data structure. Currently, you can use only the current vSphere cluster for provisioning file volumes if it is a vSAN cluster with enabled vSAN File Service.

To deactivate the persistent volumes support on a Supervisor, pass an empty list when you set the Cloud Native Storage persistent storage support for the cluster. After that existing ReadWriteMany persistent volumes provisioned in the cluster remain unaffected and usable.

Supervisor Networking

You can enable a Supervisor with vSphere networking or NSX to provision connectivity to Kubernetes control planes, services, and workloads.

A Supervisor that uses the vSphere networking stack is backed by a vSphere Distributed Switch and requires a load balancer to provide connectivity to DevOps users and external services. The NSX Advanced Load Balancer and the HAProxy load balancers are supported for vSphere 7.0 Update 2.

A Supervisor that is configured with NSX, uses the software-based networks of the solution and an NSX Edge load balancer to provide connectivity to external services and DevOps users.

Configuring NSX for vSphere with Tanzu

vSphere with Tanzu requires specific networking configuration to allow you to connect to the Supervisors, vSphere Namespaces, and all objects that run inside the namespaces.

Follow the instructions for installing and configuring the NSX for managing Kubernetes workloads documented in the *Installing and Configuring vSphere with Tanzu* guide.

First, you need to create a vSphere Distributed Switch and a distributed port group for each NSX Edge uplink. To automate this step, use the Web Services APIs as described in the *vSphere Web Services SDK Programming Guide*. Then, you can use the NSX REST APIs to add a compute manager, create transport zones, and perform other steps required for configuring the NSX for vSphere with Tanzu.

Configuring the vSphere Networking Stack for vSphere with Tanzu

To configure a Supervisor with the vSphere networking stack, you must connect all hosts from the cluster to a vSphere Distributed Switch. Depending on your topology, you must create one or more distributed port groups on the switch and configure them as workload networks to the vSphere Namespaces on the cluster.

Workload networks provide connectivity to the nodes of Tanzu Kubernetes clusters and to the Supervisor control planes. The workload network that provides connectivity to Supervisor control planes is called primary workload network. Each Supervisor must have one primary workload network represented by a distributed port group.

The Supervisor control planes on the cluster use three IP addresses from the IP address range that is assigned to the primary workload network. Each node of a Tanzu Kubernetes cluster has a separate IP address assigned from the address range of the workload network that is configured with the namespace where the Tanzu Kubernetes cluster runs.

To create a vSphere Distributed Switch and port groups for configuring the vSphere networking stack of a Supervisor, you can use the vSphere Web Services APIs as described in the *vSphere Web Services SDK Programming Guide* documentation. When you create a distributed virtual switch, vCenter Server automatically creates one distributed virtual port group. You can use this port group as the primary workload network and use it to handle the traffic for the Supervisor control planes. Then you can create as many distributed port groups for the workload networks as your topology requires. For a topology with one isolated workload network, create one distributed port group that you will use as a network for all namespaces on the Supervisor.
For a topology with isolated networks for each vSphere Namespace, create the same number of distributed port groups as the number of namespaces.

To list all workload networks available for a Supervisor and retrieve information about the configuration of a specific workload network, use the `Networks` service from the vSphere Automation REST APIs. To associate a vSphere Distributed port group to a workload network, set the necessary information through the `PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/cluster_id/networks/network_id` request and submit a `NamespaceManagementNetworksSetSpec` JSON object in the request body. The `NamespaceManagementNetworksSetSpec` data structure holds the `NamespaceManagementNetworksVsphereDVPGNetworkSetSpec` structure through which you define the parameters of the existing vSphere Distributed port group.

If you want to retrieve a list of the distributed switches compatible with vSphere with Tanzu on a vCenter Server system, use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/distributed-switch-compatibility` request and pass the **`VSPHERE_NETWORK`** value for the `network_provider` query parameter.

Installing and Configuring the HAProxy Load Balancer

You can use the vSphere Automation APIs to customize the HAProxy control plane VM after you install the HAProxy in your vSphere with Tanzu environment.

If you use the vSphere networking stack in your vSphere with Tanzu environment, you need to supply your own load balancer. You can use the open source implementation of the HAProxy load balancer that VMware provides.

For more information about the prerequisites for installation and the deployment procedure through the vSphere Client, see the *Installing and Configuring vSphere with Tanzu* documentation.

You can use the vSphere Automation REST APIs to install and configure the HAProxy load balancer. You can download the latest version of the HAProxy OVA file from the VMware-HAProxy site to a content library item. For more information about how to achieve this task, see Upload an OVF or OVA Package from a Local File System to a Library Item . Then you can create a new VM from the OVA template in the content library as described in Deploy a Virtual Machine or vApp from an OVF Template in a Content Library.

To configure the HAProxy load balancer, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/<cluster_id>/load-balancers/<loadbalancer_id>` request and submit a `LoadBalancers.SetSpec` JSON object in the request body. You can define the following HAProxy load balancer settings within the `LoadBalancers.HAProxyConfigSetSpec` data structure which is an optional property of the `LoadBalancers.SetSpec` data structure.

Parameter	Description
`servers`	A list of `Server`s that represent the endpoints for configuring the HAProxy load balancers. Each endpoint is described by a load balancer IP address and a Data Plane API management port. Each endpoint must be described with the port on the HAProxy VM on which the Data Plane API service listens. The Data Plane API service controls the HAProxy server and runs inside the HAProxy VM. The default port is *5556*. Port *22* is reserved for SSH.
`username`	The administrator user name that is configured with the HAProxy OVA file and is used to authenticate to the HAProxy Data Plane API server.
`password`	The password for the administrator user name.
`certificate_authority_chain`	The certificate in PEM format that is signed or is a trusted root of the server certificate that the Data Plane API server presents.

Using the NSX Advanced Load Balancer with vSphere Networking

If you use the vSphere networking stack for workload management, you can install and configure the NSX Advanced Load Balancer, also known as Avi Load Balancer, Essentials Edition, to support the Tanzu Kubernetes clusters.

For more information about how to install and configure the NSX Advanced Load Balancerthrough the vSphere Client, see the *Installing and Configuring vSphere with Tanzu* documentation.

You can use the vSphere Automation APIs to deploy the Avi Controller on your vSphere Management network. You can upload the latest version of the NSX Advanced Load Balancer to a library item from your local file system or from a URL. For more information about how to achieve this task, see Upload an OVF or OVA Package from a Local File System to a Library Item . Then you can deploy the Controller VM on your vSphere Management network from the OVA template in the content library as described in Deploy a Virtual Machine or vApp from an OVF Template in a Content Library.

To configure the NSX Advanced Load Balancer, send a `PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/<cluster_id>/load-balancers/<loadbalancer_id>` request and submit a `LoadBalancers.SetSpec` JSON object in the request body. You can define the following NSX Advanced Load Balancer settings within the `LoadBalancers.AviConfigSetSpec` data structure which is an optional property of the `LoadBalancers.SetSpec` data structure.

Parameter	Description
`server`	The address of the Avi Controller that is used to configure virtual services.
`username`	The administrator user name that is used for accessing the Controller VM of the NSX Advanced Load Balancer.

Parameter	Description
`password`	The password for the administrator user name.
`certificate_authority_chain`	The certificate in PEM format that is used by the Controller. You can use the certificate that you assigned during the configuration of the NSX Advanced Load Balancer.

Enable vSphere with Tanzu on a Cluster with NSX as the Networking Stack

Through the vSphere Automation APIs, you can enable a vSphere cluster for managing Kubernetes workloads. A cluster configured with NSX supports running vSphere Pod and Tanzu Kubernetes clusters.

To enable a vSphere cluster for Kubernetes workload management, you use the services under the `namespace_management` package.

Prerequisites

- Verify that your environment meets the system requirements for enabling vSphere with Tanzu on the cluster. For more information about the requirements, see the *vSphere with Tanzu Concepts and Planning* documentation.
- Verify that the NSX is installed and configured. See Configuring NSX for vSphere with Tanzu.
- Create storage policies for the placement of pod ephemeral disks, container images, and Supervisor control plane cache.
- Verify that DRS is enabled in fully automated mode and HA is also enabled on the cluster.
- Configure shared storage for the cluster. Shared storage is required for vSphere DRS, HA, and storing persistent volumes of containers.
- Verify that the user who you use to access the vSphere Automation services has the **Modify cluster-wide configuration** privilege on the cluster.
- Create a subscribed content library on the vCenter Server system to accommodate the VM image that is used for creating the nodes of the Tanzu Kubernetes clusters.

Procedure

1 Retrieve the IDs of the tag-based storage policies that you configured for vSphere with Tanzu.

   Use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/storage/policies` request to retrieve a list of all storage policies and then filter the policies to get the IDs of the policies that you configured for the Supervisor.

2 Retrieve the IDs of the vSphere Distributed Switch and the NSX Edge cluster that you created when configuring the NSX for vSphere with Tanzu.

   Use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/networks/nsx/distributed-switches?action=check_compatibility` request to list all vSphere Distributed Switches associated with the specific vSphere cluster and then retrieve the ID of the Distributed Switch that you configured to handle overlay networking for the Supervisor.

   Use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/networks/nsx/edges?action=check_compatibility` request to retrieve a list of the created NSX Edge clusters for the specific vSphere cluster and associated with the specific vSphere Distributed Switch. Retrieve the ID of the NSX Edge cluster that has the tier-0 gateway that you want to use for the namespaces networking.

3 Retrieve the ID of the port group for the management network that you configured for the management traffic.

   To list the visible networks available on the vCenter Server instance that match some criteria, use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/<cluster_id>/networks` request and then retrieve the ID of the management network you previously configured.

4 Create a `Clusters.EnableSpec` JSON object and define the parameters of the Supervisor that you want to create.

   You must specify the following required parameters of the enable specification:

   - Storage policies settings and file volume support. The storage policy you set for each of the following parameters ensures that the respective object is placed on the datastore referenced in the storage policy. You can use the same or different storage policy for the different inventory objects.

Property	Description
`ephemeral_storage_policy`	Specify the ID of the storage policy that you created to control the storage placement of the vSphere Pods.
`image_storage`	Set the specification of the storage policy that you created to control the placement of the cache of container images.
`master_storage_policy`	Specify the ID of the storage policy that you created to control the placement of the Supervisor control plane cache.

     Optionally, you can activate the file volume support by using `cns_file_config`. See Enabling ReadWriteMany Support.

- Management network settings. Configure the management traffic settings for the Supervisor control plane.

Property	Description
`network_provider`	Specify the networking stack that must be used when the Supervisor is created. To use the NSX as the network solution for the cluster, set `NSXT_CONTAINER_PLUGIN`.
`master_management_network`	Enter the cluster network specification for the Supervisor control plane. You must enter values for the following required properties: ■ `network`- Use the management network ID retrieved in Step 3. ■ `mode` - Set `STATICRANGE` or `DHCP` for the IPv4 address assignment mode. The `DHCP` mode allows an IPv4 address to be automatically assigned to the Supervisor control plane by a DHCP server. You must also set the floating IP address used by the HA primary cluster by using `floating_IP`. Use the DHCP mode only for test purposes. The `STATICRANGE` mode, allows the Supervisor control plane to have a stable IPv4 address. You can use it in a production environment. ■ `address_range`- Optionally, you can configure the IPv4 addresses range for one or more interfaces of the management network. Specify the following settings: ■ The starting IP address that must be used for reserving consecutive IP addresses for the Supervisor control plane. Use up to 5 consecutive IP addresses. ■ The number of IP addresses in the range. ■ The IP address of the gateway associated with the specified range. ■ The subnet mask to be used for the management network.
`master_DNS`	Enter a list of the DNS server addresses that must be used from the Supervisor control plane. If your vCenter Server instance is registered with an FQDN, you must enter the IP addresses of the DNS servers that you use with the vSphere environment so that the FQDN is resolvable in the Supervisor. The list of DNS addresses must be specified in the order of preference.

Property	Description
`master_DNS_search_domains`	Set a list of domain names that DNS searches when looking up for a host name in the Kubernetes API server. Order the domains in the list by preference.
`master_NTP_servers`	Specify a list of IP addresses or DNS names of the NTP server that you use in your environment, if any. Make sure that you configure the same NTP servers for the vCenter Server instance, all hosts in the cluster, the NSX, and vSphere with Tanzu. If you do not set an NTP server, VMware Tools time synchronization is enabled.

- Workload network settings. Configure the settings for the networks for the namespaces. The namespace network settings provide connectivity to vSphere Pods and namespaces created in the Supervisor.

Property	Description
ncp_cluster_network_spec	Set the specification for the Supervisor configured with the NSX networking stack. Specify the following cluster networking configuration parameters for NCPClusterNetworkEnableSpec: ■ cluster_distributed_switch - The vSphere Distributed Switch that handles overlay networking for the Supervisor. ■ nsx_edge_cluster - The NSX Edge cluster that has tier-0 gateway that you want to use for namespace networking. ■ nsx_tier0_gateway- The tier-0 gateway that is associated with the cluster tier-1gateway. You can retrieve a list of NSXTier0Gateway objects associated with a particular vSphere Distributed Switch and determine the ID of the tier-0 gateway you want to set. ■ namespace_subnet_prefix - The subnet prefix that defines the size of the subnet reserved for namespaces segments. Default is 28. ■ routed_mode - The NAT mode of the workload network. If set to false: ■ The IP addresses of the workloads are directly accessible from outside the tier-o gateway and you do not need to configure the egress CIDRs. ■ File Volume storage is not supported. Default is true. ■ egress_cidrs - The external CIDR blocks from which the NSX Manager assigns IP addresses used for performing source NAT (SNAT) from internal vSphere Pods IP addresses to external IP addresses. Only one egress IP address is assigned for each namespace in the Supervisor. These IP ranges must not overlap with the IP ranges of the vSphere Pods, ingress, Kubernetes services, or other services running in the data center. ■ ingress_cidrs- The external CIDR blocks from which the ingress IP range for the Kubernetes services is determined. These IP ranges are used for load balancer services and Kubernetes ingress. All Kubernetes ingress services in the same namespace share a common IP address. Each load balancer service is assigned a unique IP address.

Property	Description
	The ingress IP ranges must not overlap with the IP ranges of the vSphere Pods, egress, Kubernetes services, or other services running in the data center. ■ pod_cidrs - The internal CIDR blocks from which the IP ranges for vSphere Pods are determined. The IP ranges must not overlap with the IP ranges of the ingress, egress, Kubernetes services, or other services running in the data center. All vSphere Pods CIDR blocks must be of at least /23 subnet size.
worker_DNS	Set a list of the IP addresses of the DNS servers that must be used on the worker nodes. Use different DNS servers than the ones you set for the Supervisor control plane.
service_cidr	Specify the CIDR block from which the IP addresses for Kubernetes services are allocated. The IP range must not overlap with the ranges of the vSphere Pods, ingress, egress, or other services running in the data center. For the Kubernetes services and the vSphere Pods, you can use the default values which are based on the cluster size that you specify.

- Supervisor size. You must set a size to the Supervisor which affects the resources allocated to the Kubernetes infrastructure. The cluster size also determines default maximum values for the IP addresses ranges for the vSphere Pods and Kubernetes services running in the cluster. You can use the GET https://*<vcenter_ip_address_or_fqdn>*/api/vcenter/namespace-management/cluster-size-info calls to retrieve information about the default values associated with each cluster size.
- Optional. Associate the Supervisor with the subscribed content library that you created for provisioning Tanzu Kubernetes clusters. See Creating, Securing, and Synchronizing Content Libraries for Tanzu Kubernetes Releases.

  To set the library, use default_kubernetes_service_content_library and pass the subscribed content library ID.

5 Enable vSphere with Tanzu on a specific cluster by passing the cluster enable specification to the Clusters service.

Results

A task runs on vCenter Server for turning the cluster into a Supervisor. Once the task completes, Kubernetes control plane nodes are created on the hosts that are part of the cluster enabled with vSphere with Tanzu. Now you can create vSphere Namespaces.

What to do next

Create and configure namespaces on the Supervisor. See Create a vSphere Namespace.

Enable vSphere with Tanzu on a Cluster with the vSphere Networking Stack

Starting with vSphere 7.0 Update 1, you can select between creating a Supervisor with the vSphere networking stack or with NSX as the networking solution. A Supervisor that is configured with the vSphere networking stack only supports Tanzu Kubernetes clusters. vSphere Pods are not supported.

To enable a cluster configured with the vSphere networking stack for Kubernetes workloads management, you must use the services under the `namespace_management` package.

Prerequisites

- Verify that your environment meets the system requirements for enabling vSphere with Tanzu on the cluster. For more information about the requirements, see the documentation.
- Verify that DRS is enabled in fully automated mode and HA is also enabled on the cluster.
- Configure shared storage for the cluster. Shared storage is required for vSphere DRS, HA, and storing persistent volumes of containers.
- Create storage policies for the placement of Kubernetes control planes.
- Create a subscribed content library on the vCenter Server system to accommodate the VM image that is used for creating nodes of Tanzu Kubernetes clusters. See Creating, Securing, and Synchronizing Content Libraries for Tanzu Kubernetes Releases.
- Add all hosts from the cluster to a vSphere Distributed Switch and create port groups for workload networks. See Configuring the vSphere Networking Stack for vSphere with Tanzu.
- Configure an HAProxy load balancer instance that is routable to the vSphere Distributed Switch that is connected to the hosts from the vSphere cluster.
- Verify that the user who you use to access the vSphere Automation services has the **Namespaces.Manage** privilege on the cluster.

Procedure

1 Retrieve the ID of the cluster which hosts were added to the vSphere Distributed Switch.

   Use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/cluster-compatibility` request to filter the clusters by using their network providers. To retrieve a list of all clusters in the vCenter Server system which are configured with the vSphere networking stack, set the network provider in the filter specification to `VSPHERE_NETWORK`.

2 Retrieve the IDs of the tag-based storage policies that you configured for vSphere with Tanzu.

   Use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/storage/policies` request to retrieve a list of all storage policies and then filter the policies to get the IDs of the policies that you configured for the Supervisor.

3 Retrieve the ID of the port group for the management network that you configured for the management traffic.

To list the visible networks available on the vCenter Server instance that match some criteria and then retrieve the ID of the management network you previously configured, use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/<cluster_id>/networks` request.

4 Create a Supervisor enable specification and define the parameters of the Supervisor that you want to enable.

You must specify the following required parameters of the enable specification:

- Supervisor size. You must set a size to the Supervisor which affects the resources allocated to the Kubernetes infrastructure. The cluster size also determines default maximum values for the IP addresses ranges for the vSphere Pod and Kubernetes services running in the cluster. You can use the `GET https://<server>/api/vcenter/namespace-management/cluster-size-info` request to retrieve information about the default values associated with each cluster size.
- Storage policy settings and file volume support. To specify the ID of the storage policy that you created to control the placement of the Supervisor control plane cache, use the `master_storage_policy` property . Optionally, you can activate the file volume support by using the `cns_file_config` property. See Enabling ReadWriteMany Support.
- Load balancer. To specify the user-provisioned load balancer configuration for the cluster, use the `load_balancer_config_spec` parameter of the enable specification. You must specify the following parameters of the `LoadBalancersTypes.ConfigSpec` specification:

Parameter	Description
`id`	A user-friendly name of the load balancer. The name must be an alphanumeric string with a maximum length of 63 characters which is unique across the namespaces in the vCenter Server instance.
`provider`	The type of the load balancer that you want to use. In vSphere 7.0 Update 2, you can choose between the HAProxy load balancer and the NSX Advanced Load Balancer. Pass as a value to this parameter one of the following constants: `HA_PROXY` or `AVI`.
`address_ranges`	The IP address ranges in CIDR format from which HAProxy allocates the IP addresses for the virtual servers. You must provide at least one IP range which is reserved by HAProxy. The CIDR range specified with this parameter must not overlap with the IPs allocated for the Kubernetes control planes and workloads. The IP range that you configure must be on a separate subnet.

Parameter	Description
ha_proxy_config_create_spec	The HAProxy runtime configuration. See Installing and Configuring the HAProxy Load Balancer.
avi_config_create_spec	The NSX Advanced Load Balancer configuration. See Using the NSX Advanced Load Balancer with vSphere Networking.

- Management network settings. Configure the network parameters for the Kubernetes control planes.

Parameter	Description
network_provider	Specify the networking stack that must be used when the Supervisor is created. To use the vSphere network as the solution for the cluster, select VSPHERE_NETWORK.
master_management_network	Enter the cluster network specification for the Supervisor control plane. You must enter values for the following required properties: ■ network- Use the management network ID retrieved in Step 3. ■ mode- Set STATICRANGE or DHCP for the IPv4 address assignment mode. The DHCP mode allows an IPv4 address to be automatically assigned to the Supervisor control plane by a DHCP server. You must also set the floating IP address used by the HA primary cluster by using floating_IP. Use the DHCP mode only for test purposes. The STATICRANGE mode, allows the Supervisor control plane to have a stable IPv4 address and can be used in a production environment.
master_DNS	Enter a list of the DNS server addresses that must be used from the Supervisor control plane. If your vCenter Server instance is registered with an FQDN, you must enter the IP addresses of the DNS servers that you use with the vSphere environment so that the FQDN is resolvable in the Supervisor. The list of DNS addresses must be specified in the order of preference.
master_DNS_search_domains	Set a list of domain names that DNS searches inside the Kubernetes control plane nodes, so that the DNS server can resolve them. Order the domains in the list by preference.
master_NTP_servers	Specify a list of IP addresses or DNS names of the NTP server that you use in your environment, if any. Make sure that you configure the same NTP servers for the vCenter Server instance, all hosts in the cluster, and vSphere with Tanzu. If you do not set an NTP server, VMware Tools time synchronization is enabled.

- Workload network settings. Configure the settings for the network that will handle the networking traffic for Kubernetes workloads running on the Supervisor.

Parameter	Description
`service_cidr`	Specify the CIDR block from which the IP addresses for Kubernetes services are allocated. The IP range must not overlap with the ranges of the vSphere Pods, ingress, egress, or other services running in the data center. For the Kubernetes services and the vSphere Pods, you can use the default values which are based on the cluster size that you specify.
`workload_networks_spec`	Enter the workload networks specifications for the cluster. To configure the primary workload network that is used to expose the Supervisor control plane to DevOps and other workloads, create a `NetworksType.CreateSpec` / `NetworksTypes.CreateSpec` instance. Enter the following parameters of the vSphere Distributed Switch: ■ `network`. The name of the vSphere Distributed Switch that is associated with the hosts in the cluster. The name must be a unique alphanumeric string that does not exceed 63 characters. ■ `network_provider`. Pass `VSPHERE_NETWORK` as value to this parameter. ■ `vsphere_network`. Optionally, you can create a `vsphere_DVPG_network_create_spec` instance to describe the configuration of the namespace network backed by the vSphere Distributed port group. You must define the following parameters for the vSphere Distributed port group specification: ■ `portgroup`. Specify the port group that serves as the primary network to the Supervisor. ■ `address_ranges`. Set the IP range for allocating IP addresses for the Kubernetes control planes and workloads. You must use unique IP ranges for each workload network. ■ `gateway`. Set the gateway for the primary network. ■ `subnet_mask`. Specify the subnet mask of the network.

- Content library settings. Add the subscribed content library that contains the VM images for deploying the nodes of Tanzu Kubernetes clusters. See Creating, Securing, and Synchronizing Content Libraries for Tanzu Kubernetes Releases.

  To set the library, use `default_kubernetes_service_content_library` and pass the subscribed content library ID.

5 Enable the Supervisor by passing the enable specification to the `Clusters` service.

Results

A task runs on vCenter Server for enabling vSphere with Tanzu on the cluster. Once the task completes, three Kubernetes control planes are created on the hosts that are part of the cluster.

What to do next

Create and configure namespaces on the Supervisor.

Upgrading a Supervisor

You can use the vSphere with Tanzu REST APIs to upgrade a single or a group of clusters to a specific version.

vSphere with Tanzu supports rolling upgrades through the vSphere Automation REST APIs for Supervisors and for the infrastructure supporting these clusters. This model ensures that there is minimal downtime for the cluster workloads during the upgrade process.

To retrieve a list of all available vSphere with Tanzu upgrade versions for a specific vCenter Server system, use the Cluster Available Versions service. You can get information about the release version, name, description, release date, and release notes for each available upgrade.

You must use the Software Clusters service for upgrading a Supervisor. You can retrieve upgrade information about all Supervisors enabled on a vCenter Server system by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/software/clusters` HTTP request. You receive a list of basic upgrade-related information for each cluster, such as the current software version, the date of the last successful upgrade, the upgrade status of the cluster, and so on. In case some of the clusters are in the process of upgrading, you can retrieve also information about their desired upgrade version. If you want to view a more detailed upgrade-related information about a cluster, you must use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/software/clusters/<cluster_id>` HTTP request.

After you view the details about the upgrade versions that you can apply on a single or multiple Supervisors, you can create upgrade specifications that define the versions you want to upgrade to. When you upgrade a batch of Supervisor and for some reason one of the clusters fails to upgrade, you receive information about the pre-check exceptions that led to that cluster upgrade failure.

Monitoring the Enable and Upgrade Supervisor Operations

When you run the Supervisor enable and upgrade operations, the status of the tasks is not returned. Since these operations might be time-consuming but critical when automating second- and third-party products with vSphere with Tanzu, you can write a logic to track the task status.

To monitor the status of the enable and upgrade Supervisor operations, use the `Clusters.Info` data structure and query the Kubernetes and configuration status of the cluster. Track the status every two minutes or so, until you receive `READY` for `kubernetes_status` and `RUNNING` for `config_status`. These statuses indicate that the Supervisor reached the desired configuration status and is ready for running Kubernetes workloads.

Reconfiguring a Supervisor

You can change some or all the predefined settings of a Supervisor through the vSphere AutomationREST APIs.

To update only some of the Supervisor settings, use the `PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/<cluster_ID>` HTTP request and submit the updated specification in the request body. To reconfigure entirely the Supervisor, use the `PUT https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/<cluster_ID>` HTTP request and submit a `Clusters.SetSpec` data structure in the request body. The specification replaces entirely all existing Supervisor settings.

The settings you can configure with the `UpdateSpec` and `SetSpec` specifications are the same as the ones that you used for enabling the Supervisor. For example, you can change the storage settings on the Supervisor. Note that the changes that you make to the storage settings after the initial cluster configuration, apply only to the newly created Supervisor control planes.

Deactivating a Supervisor

You can programmatically deactivate vSphere with Tanzu on a vSphere cluster by using the vSphere Automation REST APIs.

When you deactivate a Supervisor, the vSphere Kubernetes Service forcefully deletes from the cluster all objects and configurations part of the Kubernetes infrastructure. To deactivate Kubernetes workloads on a cluster, Use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters/<cluster_id>?action=disable` request and pass the cluster ID as a path parameter.

Content Libraries in vSphere with Tanzu

vSphere with Tanzu uses content libraries as centralized repositories for templates, VM images, Tanzu Kubernetes release distributions, and other files related to their deployment.

Creating, Securing, and Synchronizing Content Libraries for Tanzu Kubernetes Releases

VMware Tanzu distributes Kubernetes software versions as Tanzu Kubernetes releases. To obtain and use these releases on your Tanzu Kubernetes clusters, you create subscribed or local content libraries.

A Tanzu Kubernetes release provides the VMware Kubernetes distribution which can be used with Tanzu Kubernetes clusters. Each Tanzu Kubernetes release is distributed as an OVA package. The Tanzu Kubernetes Grid uses the OVA package to deploy the virtual machine nodes for Tanzu Kubernetes clusters.

A Tanzu Kubernetes release is supported on Photon OS. The virtual machine nodes that are built from the OVA package have a 16 GB disk size. You specify the CPU and RAM resource reservations when you use a virtual machine class to size the Tanzu Kubernetes cluster.

Depending on your need for synchronization frequency and on the access to the published content libraries storing the Tanzu Kubernetes releases, you can use two approaches for storing Tanzu Kubernetes releases.

Note Starting with vSphere 7.0 Update 3, you can protect your content library by a security policy. In such case, make sure that all library items are compliant. If a protected library includes a mix of compliant and non-compliant library items, DevOps engineers are not able to retrieve the list of VM images provided with the library.

Automated Synchronization of Tanzu Kubernetes Releases

VMware publishes a content library that contains the latest VMware distributions of Kubernetes as an OVA package. If you want to provision Tanzu Kubernetes clusters, you can create a subscribed content library on the vCenter Server instance where vSphere with Tanzu is enabled. When configuring the content library subscription, use the following subscription URL of the publisher : https://wp-content.vmware.com/v2/latest/lib.json. For more information about how to create a subscribed content library, see Subscribing to a Content Library.

When you create the subscription, you configure the synchronization mechanism for downloading the content of the published library. You can select between on demand and automatic download of the virtual machine image for the Tanzu Kubernetes cluster nodes. If you choose to synchronize the subscribed library on demand, only the metadata for the library content is updated and as a result storage space is saved. This approach is an important consideration as more images containing different Kubernetes versions are published. However, the first time you decide to use a new virtual machine image version, you have to wait for it to download.

Starting with vSphere 7.0 Update3, you can secure a subscribed content library. The Content Library service verifies the library signing certificate during the synchronization process. If the certificate verification fails, only the library metadata is synchronized and the library content is not downloaded. For more information how to apply a security policy when you update a subscribed content library, see Editing the Settings of a Content Library.

You associate the subscribed content library with the Supervisor on which you want to create a Tanzu Kubernetes cluster, when you first enable vSphere with Tanzu on a cluster. See Enable vSphere with Tanzu on a Cluster with NSX as the Networking Stack.

The size of the content library can grow over time as new Kubernetes versions and images are published. If the underlying storage runs out of space, you will need to move to a new subscribed content library. After you create a new subscribed content library that has sufficient capacity for the target cluster, update the library association of the Supervisor. See Reconfiguring a Supervisor.

Manual Synchronization of Tanzu Kubernetes Releases

In an air-gapped network environment, you can use the storing functionality provided by a local content library for the needed Tanzu Kubernetes releases. You must first create a local content library, then download the OVA package for each Tanzu Kubernetes release that you want to import to the library. See Creating a Local Content Library.

Starting with vSphere 7.0 Update3, you can secure a local content library. The Content Library service verifies the library signing certificate during the synchronization process. If the certificate verification fails, only the library metadata is synchronized and the library content is not downloaded. For more information how to apply a security policy when you update a local content library, see Editing the Settings of a Content Library.

You can find the latest versions of the Kubernetes distribution by navigating to the https://wp-content.vmware.com/v2/latest URL. You must download the `photon-ova.ovf` and `photon-ova-disk1.vmdk` for each distribution you want and then upload these files from your local file system to your local content library. See Upload an OVF or OVA Package from a Local File System to a Library Item .

Note Make sure that you use as a name for each library item the Photon image version and the Kubernetes version from the directory where you downloaded the files. For example: `photon-3-k8s-v1.20.2---vmware.1-tkg.1.1d4f79a`.

Creating and Managing Content Libraries for VM Provisioning in vSphere with Tanzu

To provision new virtual machines in a vSphere with Tanzu environment, the DevOps engineers rely on VM templates and images. Your role is to make sure the DevOps engineers have access to these VM templates and images by using the Content Library service.

You can create a local content library and populate it with VM templates in OVF or OVA file format, or other types of files. For more information and a sample of how to create a local content library, see Creating a Local Content Library.

You can also create a subscription to download the content of a published local content library as described in the following topic: Subscribing to a Content Library.

Starting with vSphere 7.0 Update3, you can secure the content library. The Content Library service verifies the library signing certificate during the synchronization process. If the certificate verification fails, only the library metadata is synchronized and the library content is not downloaded. For more information how to apply a security policy when you update a local or subscribed content library, see Editing the Settings of a Content Library.

After you create the content library, you must populate it with content either from your local file system or from a Web server. You must use only the VM images available on the VMware Cloud Marketplace web site. For example, download or subscribe to VM Service Image for Ubuntu if you want to enable a DevOps engineer to deploy a VM using this image. For more information about the available ways to populate a content library with content, see Working with Library Items.

You must give the DevOps engineers access to the VM templates stored in the content libraries, so that they can use these templates to provision VMs through the VM Service functionality. To give access, you must associate one or more content libraries to the namespace where the VM Service is present. See Associating a Content Library with a Namespace and Virtual Machines in vSphere with Tanzu.

Associating a Content Library with a Namespace

You must give access to a source of VM templates, so that the DevOps engineers can use them to provision VMs in a self-service manner. To give access, you associate a content library with VM templates to the namespace used by the DevOps engineers.

You can add multiple content libraries to a namespace that has the VM Service enabled or the same content library to several namespaces. You associate a content library to a namespace when you create a new namespace, update or reconfigure an existing one.

To make the VM Service aware of the content libraries in your environment that the DevOps engineers can use to self-service VMs, you must use the `NamespacesInstancesVMServiceSpec` data structure which is part of the namespace create specification. The instance contains a list of content libraries that will be used by the VM Service. You can specify this list with the `content_libraries` property.

You can also associate one or more VM classes with the namespace. See Associating a VM Class with a vSphere Namespace.

Managing Namespaces on a Supervisor

You can use the vSphere Automation APIs to create namespaces on a Supervisor and configure them with resource limits and permissions for the DevOps users.

To create and configure a namespace, use the `Instances` service from the `namespaces` package. You can configure the access control to the objects in a namespace by using the `Access` service.

Create a vSphere Namespace

You can use the vSphere with Tanzu automation REST APIs to create namespaces on a Supervisor. You can set resource quotas, storage, as well as permissions for the DevOps users.

Prerequisites

- Enable vSphere with Tanzu on a vSphere cluster.

- Create users and groups for the DevOps engineers who will use the namespace. For more information about how to create users and groups through the Web Services APIs, see the *vSphere Web Services SDK Programming Guide*.
- Create storage policies for persistent storage used by the vSphere Pods and the pods inside a Tanzu Kubernetes cluster.
- Create VM Classes and content libraries for DevOps provisioned VMs. See Create a VM Class in vSphere with Tanzu and Creating and Managing Content Libraries for VM Provisioning in vSphere with Tanzu.
- Required privileges on the Supervisor:
  - **Namespaces.Modify cluster-wide configuration**
  - **Namespaces.Modify namespace configuration**
  - **Virtual Machine Classes.Manage Virtual Machine Classes**

Procedure

1 Retrieve the Supervisor ID by filtering the clusters available in the vCenter Server system.

Use the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/clusters` request and retrieve the ID of the cluster on which you want to create a namespace from the list of cluster summary JSON objects.

2 Retrieve the ID of the storage policy that you configured for placement of the persistent volumes from vSphere Pods and Tanzu Kubernetes clusters.

3 Configure the access control to the objects in the namespace.

Populate the properties of the `Instances.Access` data structure with appropriate values:

Property	Description
`domain`	Set the domain name of the vCenter Server system on which the namespace is created.
`subject_type`	Set the type of the user accounts that are associated with the specific role for the namespace. You must select between the `USER` and `GROUP` options.
`subject`	Set the name of the user or group that have permissions to access the namespace objects.
`role`	Set the role that is associated with the predefined set of privileges that you want to grant the specific user or group. You can select between the `EDIT`, `VIEW` and `OWNER` roles. The owner role is introduced in vSphere 7.0 Update 2a. When a DevOps engineer creates a namespace in a self-service manner, the Namespace Self-Service grants the owner role to the namespace creator. See Self-Service Namespace Management.

4 Populate the `Instances.CreateSpec` data structure with the appropriate namespace specification information.

The namespace specification can contain the following information:

Property	Description
`cluster`	Set the ID of the Supervisor on which the namespace is created.
`namespace`	Set a name of the namespace following the DNS label standard defined in RFC 1123. The name must be unique across all namespaces in the current vCenter Server system.
`networks`	Optional. You can set the workload networks used by the vSphere Namespace. Pass `null` as a value of this parameter, if the Supervisor is configured to use NSX as networking solution. The workload networking support for such namespaces is provisioned by NSX. If the Supervisor uses the vSphere networking stack, pass the workload network to be associated with the namespace. If you pass `null` as a value of this parameter, the vSphere Namespaces on the cluster are automatically associated with the cluster primary workload network. See Configuring the vSphere Networking Stack for vSphere with Tanzu.
`description`	Optional. You can set a description of the namespace.
`access_list`	Optional. You can set the access control that is associated with the namespace in Step 3.
`storage_specs`	Optional. You can set the amount of storage dedicated to each storage policy associated with the namespace and the maximum amount of storage that is used by the namespace. Use the `StorageSpec` specification to configure the storage quotas on the namespace.
`resource_spec`	Optional. You can set resource limitations to the namespace. You can limit the CPU, memory, the maximum number of pods that can exist on the namespace, and so on.
`creator`	Optional. The Namespace Self-Service populates this parameter with information about the DevOps user who created the namespace with `cubectl`. The user name and domain of the namespace creator are stored with this parameter.
`vm_service_spec`	Optional. The VM Service specification for the Dev-Ops provisioned virtual machines.

5 Create a namespace object on the Supervisor by using the namespace create specification.

Use the `POST https://`*`<vcenter_ip_address_or_fqdn>`*`/api/vcenter/namespaces/instances` request and submit an `Instances.CreateSpec` JSON object in the request body.

What to do next

Share the namespace with DevOps engineers and provide them with the user or group configured for accessing the namespace.

Updating the Namespace Configuration

You can change the whole namespace configuration or only some of the namespace settings.

To change the configuration of an existing namespace, you must have the **Namespaces.Configure** privilege on the Supervisor.

Note Before deleting a storage policy from vCenter Server or a vSphere Namespace, or changing the storage policy assignment, make sure that no persistent volume claim with the corresponding storage class runs in the namespace. Also, ensure that no Tanzu Kubernetes cluster is using the storage class.

To patch a namespace configuration, use the `PATCH https://`*`<vcenter_ip_address_or_fqdn>`*`/api/vcenter/namespaces/instances/`*`<namespace_id>`* request and submit an `UpdateSpec` JSON object in the request body. You set new values only to the configuration settings that you want to change. The parameters of the update specification are the same as the ones you configured during the namespace creation. When you issue the update request only the settings that you configured in the update specification are applied, the other settings are left as they are.

To reconfigure a namespace entirely, you must issue a `PUT https://`*`<vcenter_ip_address_or_fqdn>`*`/api/vcenter/namespaces/instances/`*`<namespace_id>`* request and submit a `SetSpec` JSON object in the request body. You can change the description, access controls, storage settings, and resource limitations of the specific namespace.

Configuring the Access to a Namespace

You can use the vSphere with Tanzu APIs to grant access permissions to DevOps engineers on the vSphere Namespaces.

Use the `Access` service to retrieve information about the access control of the DevOps engineers on a specific namespace. You can also set up or remove an access control for a specific user or group on a specific namespace, and add another access control on the namespace. You set up each access control to allow a user or group to access a namespace in a specific vCenter Server system. You can grant access to a DevOps engineer to more than one namespace.

You must have the **Namespaces.Configure.** privilege to grant permissions to a user. You assign the view and edit access role on the namespace for the user or group.

Starting with vSphere 7.0 Update 2a, you can also assign the owner role to a DevOps engineer. These roles allow the user to deploy workloads, share the namespace with other DevOps engineers, and delete it when it is no longer needed.

Self-Service Namespace Management

You can use the vSphere with Tanzu automation REST APIs to create a vSphere Namespace with specific resource quotas, set permissions, and assign storage policies. DevOps engineers can then use the namespace as a template for self-provisioning namespaces on the cluster.

Starting with vSphere 7.0 Update 2a, the Namespace Self-Service feature is available in vSphere with Tanzu. The service enables Kubernetes users to create vSphere Namespaces from templates configured through the automation REST APIs or vSphere Client. To activate the Namespace Self-Service on a cluster, use one of the following options:

- Create a self-service namespace template and then activate the Namespace Self-Service on the cluster.
- Create or update a self-service namespace template simultaneously with activating the Namespace Self-Service on the cluster.

Currently, only one namespace self-service template is allowed per vSphere Namespace. After a DevOps engineer creates a namespace from the template, the namespace can also be deleted through `kubectl`. You can verify whether a namespace is created from a template by retrieving the value of the `self_service_namespace` property of the `Instances.Info` JSON object that you receive in the request body of the `GET https://`*`<vcenter_ip_address_or_fqdn>`*`/api/vcenter/namespaces/instances/`*`namespace_id`* request.

To create a template for a self-service namespace, use the `POST https://`*`<vcenter_ip_address_or_fqdn>`*`/api/vcenter/namespaces/namespace-templates/clusters/`*`cluster_id`* HTTP request. You use as path parameter the cluster ID and submit in the request body the namespace template create specification.

You define the following configuration settings and resource limitations of the template:

Property	Description
`template`	The identifier of the namespace template must be a unique name across all clusters on the vCenter Server instance. The name must be compliant with DNS.
`resource_spec`	The resource quotas, such as CPU and memory, that are reserved for the namespace on the vCenter Server instance. The CPU limit is set in MHz and the minimum value is 10 MHz. The memory and the storage limits are set in MiB. For more options to configure resource limits for the namespace, see the `ResourceQuotaOptionsV1` class in the API Reference documentation.
`storage_specs`	The amount of storage in MiB utilized for each storage policy that you associate with the namespace. You must specify at least one policy.

Property	Description
`networks`	Optional. The networks associated with the namespace. Currently, you can set only one network for the namespace. Pass `null` as argument if the Supervisor is configured with NSX-T Data Center support. If you pass `null` for a namespace template on a cluster configured with a vSphere networking stack, the namespace is automatically associated with the Supervisor management workload network.
`permissions`	Optional. The permissions that allow DevOps engineers to use the template to self-provision namespaces through `kubectl`. If unset, only users with the Administrator role can use the template.

Once you have the template created, you can activate the Namespace Self-Service on the cluster by issuing the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespaces/namespace-self-service/<cluster_id>?action=activate` HTTP request. If you want to restrict DevOps users to use the namespace template on a cluster, you can deactivate the Namespace Self-Service feature. Then users are able to delete only the namespaces already created from the template.

You can activate the Namespace Self-Service on the cluster after configuring the namespace template by using the `NamespaceSelfService` service. You issue the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespaces/namespace-self-service/<cluster_id>?action=activateWithTemplate` HTTP request and submit the `NamespaceSelfService.ActivateTemplateSpec` JSON object in the request body. Depending on the availability of a template on the cluster, this method either creates a namespace template or activates the deactivated service and at the same time updates the existing template.

Virtual Machines in vSphere with Tanzu

vSphere with Tanzu offers the VM Service functionality to enable DevOps engineers to provision and manage VMs on a namespace in a self-service manner. You use the vSphere with Tanzu automation APIs to create VM classes that specify the deployment policy and resource reservations of such VMs.

Starting with vSphere 7.0 Update 2a, DevOps engineers can use the VM Service functionality to deploy and run VMs on a namespace through the `kubectl` commands. You can use the vSphere with Tanzu automation APIs to manage the two VM Service components: VM classes and content libraries. For more information about managing content libraries in the context of vSphere with Tanzu, see Content Libraries in vSphere with Tanzu.

You can use the automation APIs to create and manage VM classes. A VM class specification defines the number of CPUs, memory capacity, and resource reservation settings of the desired virtual machine. vSphere with Tanzu currently offers twelve ready-to-use VM classes (T-shirt sizes) that are derived from the most popular VMs in Kubernetes. Based on the resource reservation that a VM specification requests, each predefined VM class has two editions: guaranteed and best effort. The guaranteed VM class fully reserves the configured resources. A best effort VM class does not guarantee any resource reservations and allows their overcommitment.

You associate a VM class with a specific namespace to make it available to the DevOps engineers who have access to that namespace. You can assign any number of existing VM classes or create a custom one. Note that VMs deployed by the DevOps engineers through the VM Service can only be managed with the `kubectl` commands. A VM provisioned by DevOps engineers shares the same resources in a namespace as containers.

Use the `VirtualMachineClasses` interface to create and manage a specification of a VM class object. Through these objects you predefine the number of CPUs, memory capacity, and reservation settings. See Create a VM Class in vSphere with Tanzu. To make a VM class available to the DevOps engineers for self-service VM deployment, you must associate it with a specific namespace. See Associating a VM Class with a vSphere Namespace.

Create a VM Class in vSphere with Tanzu

You can use the vSphere Automation Kubernetes REST APIs to create custom VM classes to be used for VM deployment in vSphere with Tanzu.

A VM class specifies the CPU, memory, and resource reservations for a VM. vSphere with Tanzu offers several preconfigured VM classes which you can use as is, edit, or delete. You can also create a custom VM class in your vCenter Server instance and it will be available to all Supervisors and the namespaces created in these clusters. Note that even though a VM class is available to all namespaces, a DevOps user can only use the VM classes associated with the namespaces that he/she can access.

Prerequisites

Required privileges:

- **Namespaces.Modify cluster-wide configuration**
- **Namespaces.Modify namespace configuration**
- **Virtual Machine Classes.Manage Virtual Machine Classes**

Procedure

1 Create the specification of the VM class object by defining the following options.

Option	Description
`id`	The identifier of the VM class must follow these DNS requirements: ■ A unique name in the current vCenter Server instance. ■ An alphanumeric name with maximum 63 characters. ■ No uppercase letters of spaces. ■ A dash can be used anywhere except as a first or last character. Note that after a VM class is created, you cannot edit its ID.
`cpu_count`	The number of virtual CPUs (vCPUs) configured for a VM that are deployed with this VM class.
`memory_MB`	The memory in MB configured for a VM that are deployed with this VM class. The value must be between 4 MB and 24 TB and a multiple of 4.
`description`	Optional. The description of the VM class.
`cpu_reservation`	Optional. The percentage of total available CPU resources reserved for the VM deployed with the VM class. The percentage you specify with this attribute is multiplied by the minimum CPU available among all cluster nodes to get the CPU resources guaranteed by vSphere for a VM. The resulting value is in MHz.
`memory_reservation`	Optional. The percentage of available memory that is reserved for a VM deployed with this VM class. The value can be from 0 through 100%.

2 Create the VM class object.

Use the `POST https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/virtual-machine-classes` HTTP request and pass the created VM class specification in the request body.

What to do next

After you create the custom VM class, you can edit its parameters or delete it from your environment. See Editing or Removing a VM Class from Your Environment.

You can make your VM class available to DevOps engineers by associating it with a namespace. See Associating a VM Class with a vSphere Namespace.

Editing or Removing a VM Class from Your Environment

You can use the automation REST APIs to edit the configuration of a VM class that you created or a predefined VM classes that vSphere with Tanzu offers. When you no longer need an existing VM class, you can remove it from your vCenter Server instance.

Note that editing a VM class specification does not affect the VMs that are already deployed by the DevOps engineers from this class. Only newly deployed VMs will use the reconfigured VM class.

Deleting a VM class results in its removal from all related namespaces. All VMs deployed with this VM class remain unchanged but DevOps engineers can no longer use it.

You can list all VM classes available for a vCenter Server instance by using the `GET https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/virtual-machine-classes` HTTP request. You receive a list of all VM classes and a detailed information about each one of them. Based on the detailed information, you can narrow the list and retrieve the IDs of the VM classes that you want to edit or delete.

To edit a VM class configuration, use the `PATCH https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/virtual-machine-classes/vm_class_id` HTTP request and pass the update specification in the request body. You can edit all VM class attributes except the ID of the class. Only the attributes modified within the update specification will be edited.

To delete a VM class from your environment, use the `DELETE https://<vcenter_ip_address_or_fqdn>/api/vcenter/namespace-management/virtual-machine-classes/vm_class_id` HTTP request.

Associating a VM Class with a vSphere Namespace

You must associate a VM class with a namespace to make it available for DevOps engineers to deploy VMs in a self-service manner.

You can associate one or more VM classes with a single namespace or you can add one VM class to several namespaces. You can use the predefined VM classes that vSphere with Tanzu provides or you can create custom ones. See Create a VM Class in vSphere with Tanzu.

Since VM Service is the feature responsible for handling VM classes, you must make that service aware of the VM classes available to the engineers using a specific namespace. You can achieve this when you create a new namespace or edit an existing one. See Managing Namespaces on a Supervisor.

When you create the VM Service specification set the list of VM classes that must be used to create VMs on the specific namespace. You achieve this by using the `vm_classes` property of the `Instances.VMServiceSpec` data structure.

You can also associate one or more content libraries with a namespace that has the VM Service enabled. See Associating a Content Library with a Namespace.

vCenter Server Management

10

You can use the API to configure, monitor, maintain, and update vCenter Server.

Read the following topics next:

Authorization Model for Administration of vCenter Server

There are three types of authorization levels in vCenter Server.

Table 10-1. Authorization Levels

Authorization Level	Description
operator	A user has read access to configuration settings.
administrator	A user has read and write access to configuration settings, but cannot manage user accounts.
super administrator	A user has all the capabilities of the other roles, and has the additional capabilities of creating local user accounts and accessing the local Bash shell.

This model applies to the API and all other interfaces to vCenter Server except when you use SSH and log in by using a local account.

Authorization Model Mapping to the vCenter Single Sign-On Domain

The three-level authorization model of vCenter Server maps to local roles and to vCenter Single Sign-On groups, depending on how the user authenticated. This model allows consistent security control regardless of operational context.

The authorization levels map to group and role.

Table 10-2. Authorization Mapping

Authorization Level	vCenter Single Sign-On Group	vCenter Server Local Role
operator	SystemConfiguration.Administrators	operator
administrator	SystemConfiguration.Administrators	admin
superAdministrator	SystemConfiguration.BashShellAdministrators	superAdmin

When a super administrator adds user accounts, the options available include a choice of the role to assign to the new user.

Using the Operator Role

The **operator** role is the most restricted of the authorization levels available to users who work with vCenter Server.

Operators are allowed to view information about vCenter Server. They are not allowed to alter its configuration. The **operator** role is suited for monitoring and reporting functions. For example, the **operator** role provides access to the following methods.

- `resources.system.health.get`
- `resources.storage.stats.list`
- `services.status.get`

Using the Admin Role

The **administrator** role provides an intermediate authorization level for users who manage vCenter Server.

An **administrator** role is required for users who alter the vCenter Server configuration, exercise control functions, or other operations that can affect regular users.

For example, the **administrator** role provides access the following methods.

- `networking.ip4v.renew`
- `networking.firewall.addr.inbound.add`
- `services.control`
- `shutdown.reboot`

Using the SuperAdmin Role

The **superAdmin** role is the most expansive authorization level for users who manage vCenter Server.

The **superAdmin** role allows unrestricted access to vCenter Server. This role is required for adding or altering user accounts and for using the Bash shell.

Performing Privilege Checks Operations

Privilege checks recorder is a feature which allows you to monitor and subsequently query the privileges that were checked. You can use the recordings to create scripts that automatically create roles with minimum required privileges to run specific workflows of operations.

Currently, it is very hard to find out the minimal set of privileges that are required to run a specific workflow of operations. The `PrivilegeChecks` service provides operations for retrieving a list of the latest privilege checks along with the corresponding sessions, users, managed objects, and operation IDs (`opIDs`).

The following table lists the available privilege checks operations.

Operation	Description
List privilege checks	Returns a list of privilege checks that match criteria which you specified.
Get latest privilege check	Returns a marker to the last recorded privilege check. This allows retrieving all privilege checks up to or after a specified moment in time.

When you retrieve a list of privilege checks, the results are returned in pages due to the possibly large number of privilege checks that can be returned. You can control the page size either by using the `AuthorizationPrivilegeChecksIterationSpec` data structure, or with the `config.vpxd.privCheck.pageSize` advanced setting.

Filtering Privilege Checks

When you retrieve lists of privilege checks, you can refine your search by using the `AuthorizationPrivilegeChecksFilterSpec` data structure. The following table lists the available filtering options.

Filtering Option	Description
`Objects`	IDs of the managed objects on which the privilege check was performed. If unset, all objects match.
`OpIDs`	`OpIDs` of the requests for which the check was performed. If unset, all `opIDs` match.
`Principals`	Principals for which the privilege check was performed. The unset `PrivilegeChecks.Principal` value matches privilege checks for anonymous sessions. If unset, all principals match.
`Privileges`	Privileges that were checked. If unset, all privileges match.
`Sessions`	Sessions for which the check was performed. If unset, all sessions match.

Using Advanced Settings

You can configure advanced settings by using the vSphere Client. Some of the advanced settings are not available in the API, but are required for the privilege checks recorder to function properly. For information about configuring advanced settings, see *vCenter Server Configuration*.

Advanced Setting Name	Description
`config.vpxd.privCheck.pageSize`	Specifies the default page size for privilege checks lists.
`config.vpxd.privCheck.bufferSize`	Specifies the count of privileges to be kept in memory. The default value is 0. If you do not change the default value, the privilege checks recorder does not record any data.
`config.vpxd.privCheck.cleanupInterval`	Specifies the interval on which privilege checks for unused sessions are cleaned up. The default value is 30 minutes.

vCenter Server Installation and Setup

You can use the API to perform operations related to stage 2 of the installation process. You can also perform backup, restore, and troubleshooting operations.

Install Stage 2

The vCenter Server API provides methods for performing stage 2 deployment operations on a newly installed vCenter Server instance.

The vCenter Server instance is deployed in two stages. With stage 1 of the deployment process, you deploy the OVA file, which is included in the installer. With stage 2 of the deployment process, you set up and start the services of the newly deployed vCenter Server instance.

To complete stage 1 of the deployment process, you can use the GUI installer or perform a CLI deployment. For details, see *vCenter Server Installation and Setup*. Alternatively, you can perform a deployment by using the VMware OVF Tool. See *OVF Tool User's Guide*.

Setting Up a Newly Installed vCenter Server Instance

You can use the API to set up a newly deployed vCenter Server instance.

After stage 1 of the deployment process completes successfully, the vCenter Server instance enters in an `INITIALIZED` state. If the instance is not initialized, you cannot run stage 2 of the deployment process. You can get the state of the vCenter Server instance by using the `vcenter deployment` service. The vCenter Server instance can enter six states during the deployment process.

Figure 10-1. Install Stage 2 State Diagram

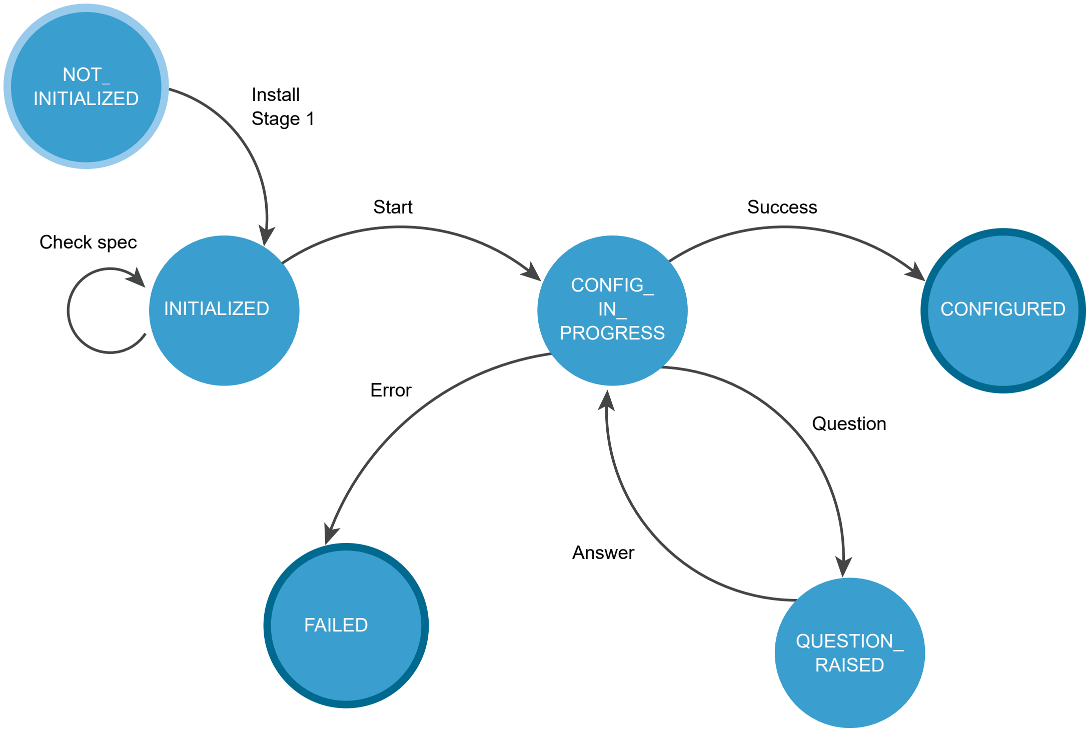


Table 10-3. vCenter Server Instance States During Install Stage 2

State	Description
`NOT_INITIALIZED`	The install stage 1 phase is in progress, not started, or failed.
`INITIALIZED`	The vCenter Server instance is deployed and ready for setup.
`CONFIG_IN_PROGRESS`	The setup process is in progress.
`QUESTION_RAISED`	You must answer the question to continue the setup process. The vCenter Server instance stays in the `QUESTION_RAISED` state until it receives the correct answer.
`FAILED`	Errors occurred during the setup process. You can check the errors, warnings, and info data structures.
`CONFIGURED`	The vCenter Server instance is installed and configured successfully.

`FAILED` and `CONFIGURED` are final states.

Table 10-4. User Operations lists operations that you can perform to set up your newly deployed vCenter Server instance.

Table 10-4. User Operations

Operation	Description
Get deployment information	You can retrieve information about the current deployment status. This operation is useful both before initiating stage 2 of the deployment and for monitoring the progress of the setup process.
Validate the configuration document	You can optionally verify whether your install spec is valid before starting the setup process.
Configure the vCenter Server instance	You can initiate the setup process by providing an install spec that defines the values for the settings that you want to configure.
Get question	You can retrieve a question raised during the setup process.
Answer question	You can provide an answer to the question raised during the setup process. The available answer values are `YES`, `NO`, `OK`, `CANCEL`, `ABORT`, `RETRY`, and `IGNORE`. The possible answer values depend on the type of the question. **Note** Each question has a default answer value. If you set questions to receive automatic answers in the install spec and a question is raised during the setup process, the default answer value is automatically provided as the answer to the question.

For information about the HTTP requests that you can use to perform the user operations, see HTTP Requests for Install Stage 2.

Workflows for Install Stage 2

You can use the `vcenter deployment` API to run the install stage 2 process of your vCenter Server instance.

Figure 10-2. Install Workflow and Figure 10-3. Install Stage 2 Workflow show example install workflows.

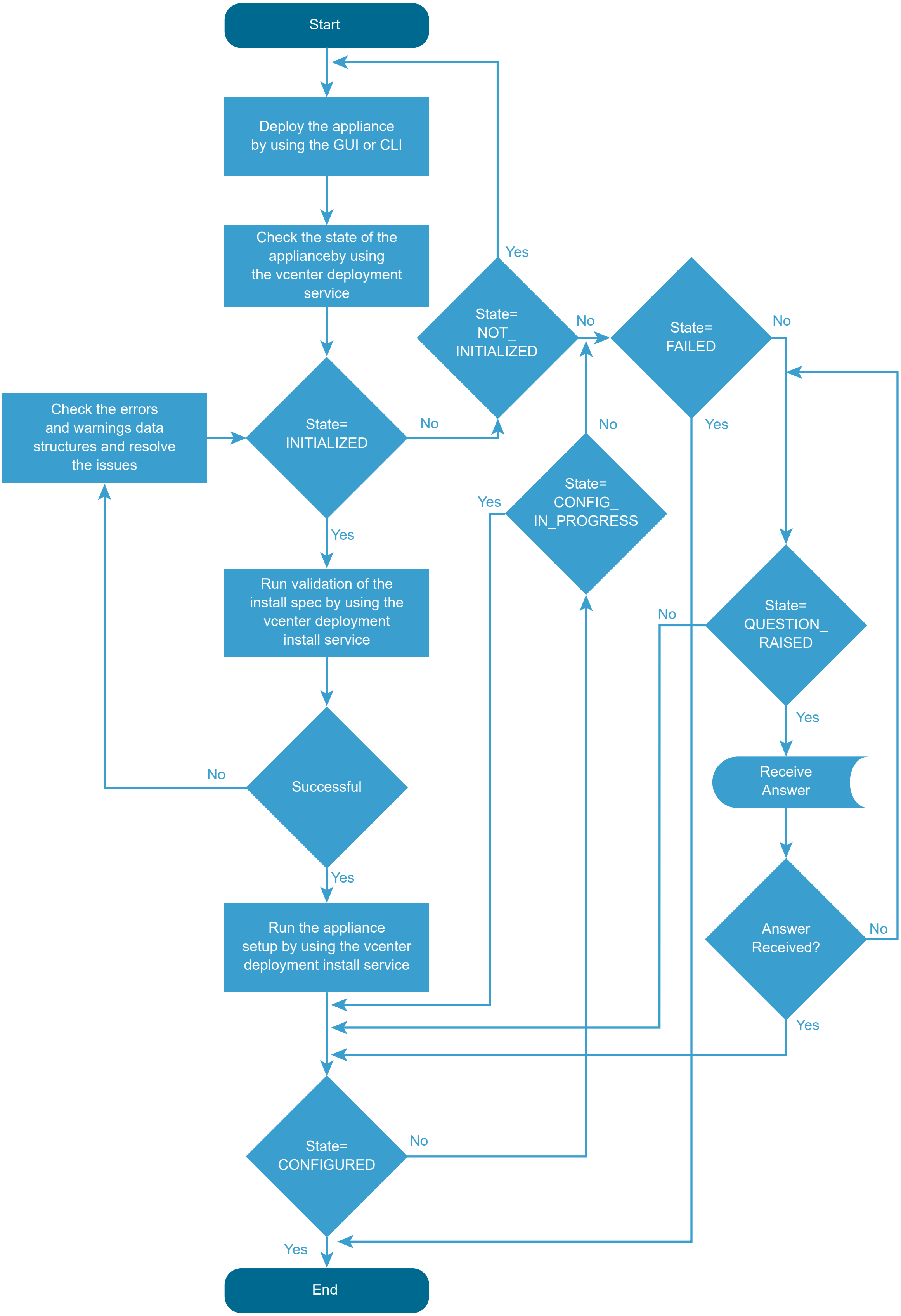
During stage 1, the vCenter Server instance is in a `NOT_INITIALIZED` state. After a successful deployment, the vCenter Server instance enters in an `INITIALIZED` state. If there are errors during stage 1, the vCenter Server instance stays in a `NOT_INITIALIZED` state and you must redeploy it.

You can check the state of the vCenter Server instance before, during, and after the setup process. You can run the install stage 2 process if the vCenter Server instance is initialized. You can check the setup configuration before you initiate stage 2 by running pre-checks. If errors or warnings appear during the validation of the install specification, you must remove the causes and correct the specification.

During the setup process, the regular vCenter Server instance state is `CONFIG_IN_PROGRESS`. The vCenter Server instance can also enter in a `FAILED` or `QUESTION_RASED` state. If a question appears during the setup, the vCenter Server instance enters in a `QUESTION_RAISED` state and stays in it until you provide an answer. You can set questions to receive automatic answers in the install spec and if a question is raised during the setup process, the default answer value is automatically provided as the answer to the question.

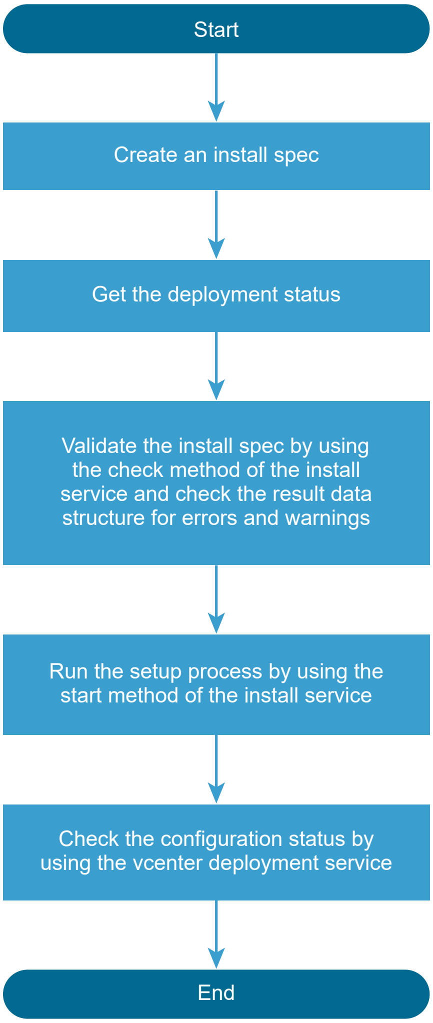
If errors occur during the setup process, the vCenter Server instance enters in a `FAILED` state and you must restart the setup after the causes are removed. If the setup is successful, the vCenter Server instance enters in a `CONFIGURED` state.

Figure 10-2. Install Workflow


Start
Deploy the appliance by using the GUI or CLI
Check the state of the applianceby using the vcenter deployment service
State= INITIALIZED
Check the errors and warnings data structures and resolve the issues
Yes
No
State= NOT_ INITIALIZED
Yes
No
State= FAILED
No
Yes
State= CONFIG_ IN_PROGRESS
Yes
No
Run validation of the install spec by using the vcenter deployment install service
Successful
No
Yes
State= QUESTION_ RAISED
No
Yes
Receive Answer
Answer Received?
No
Yes
Run the appliance setup by using the vcenter deployment install service
State= CONFIGURED
No
Yes
End

For information about the states of the vCenter Server instance and available operations, see Setting Up a Newly Installed vCenter Server Instance.

Figure 10-3. Install Stage 2 Workflow


Start
Create an install spec
Get the deployment status
Validate the install spec by using the check method of the install service and check the result data structure for errors and warnings
Run the setup process by using the start method of the install service
Check the configuration status by using the vcenter deployment service
End

You can run the setup pre-checks and the install stage 2 process by creating and passing an `InstallSpec`. In `InstallSpec`, you define the setup configuration. See Figure 10-3. Install Stage 2 Workflow . You can run the setup in silent mode by setting the `InstallSpec.auto_answer` to `true`. The default value of `InstallSpec.auto_answer` is `false` and the setup is in interactive mode, in which you must provide answers to the raised questions.

For information about the classes, variables, and default values, see the *API reference* documentation.

HTTP Requests for Install Stage 2

You can use HTTP requests to set up a newly deployed vCenter Server instance.

HTTP Requests

After stage 1 of the deployment process completes successfully, you can perform setup by sending HTTP requests.

Note When you send the requests, you must authenticate with vCenter Server **root** credentials.

The following HTTP requests show the syntax that you can use to perform the available user operations.

- Get deployment information

  ```
  GET https://<vcenter_ip_address_or_fqdn>:5480/rest/vcenter/deployment
  ```

- Validate the install spec

  ```
  POST https://<vcenter_ip_address_or_fqdn>:5480/rest/vcenter/deployment/install?action=check
  ```

- Configure the vCenter Server instance

  ```
  POST https://<vcenter_ip_address_or_fqdn>:5480/rest/vcenter/deployment/install?action=start
  ```

- Get question

  ```
  GET https://<vcenter_ip_address_or_fqdn>:5480/rest/vcenter/deployment/question
  ```

- Answer question

  ```
  POST https://<vcenter_ip_address_or_fqdn>:5480/rest/vcenter/deployment/question?
  action=answer
  ```

For information about the content and syntax of the HTTP request body, see the *API reference* documentation.

HTTP Status Codes and Errors

Table 10-5. HTTP Status Codes and Errors lists the status codes that you can receive when you send HTTP requests.

Table 10-5. HTTP Status Codes and Errors

HTTP Status Code	Description	Operations that Return the Status Code
200	The operation is successful.	All operations. You can check the returned data in the results data structure.
400	You cannot perform the operation because the vCenter Server instance is in the current state. For information about the states of the vCenter Server instance, see Setting Up a Newly Installed vCenter Server Instance.	■ Validate the install spec ■ Start the setup ■ Get the install spec ■ Get the raised question ■ Answer the question ■ Get the state of the vCenter Server instance
401	You use an invalid user name or password, or authentication has failed.	■ Validate the install spec ■ Start the setup ■ Get the install spec ■ Get the raised question ■ Answer the question ■ Get the state of the vCenter Server instance
404	The state of the vCenter Server instance cannot be determined .	Get the vCenter Server instance state.
500	There is a `vapi std` error. For information about the types of `vapi std` errors, see `vapi.std.errors` in the *API reference* documentation.	Get the raised question.

If errors occur during the setup process, you can check the `results` data structure, the API log file, and download the vCenter Server instance support bundle from `https://<vcenter_ip_address_or_fqdn>:443/appliance/support-bundle`.

Use HTTP Requests to Set Up a Newly Deployed vCenter Server Instance

You can send HTTP requests to complete stage 2 of the deployment process of a newly deployed vCenter Server instance.

You set up a newly deployed vCenter Server instance by providing configuration settings in the body of the HTTP request.

Prerequisites

- Verify that the newly deployed vCenter Server instance is reachable.
- Verify that you have the correct credentials for sending HTTP requests.

Procedure

1 Check whether the vCenter Server instance state is set to `INITIALIZED`.

```
GET https://<server>:5480/rest/vcenter/deployment
```

If the vCenter Server instance is in the correct state, you receive a message body that contains the following line and you can continue with the setup process.

```
...
    "state": "INITIALIZED",
...
```

2 (Optional) Validate the configuration settings that you provide in the body of the HTTP request.

```
POST https://<server>:5480/rest/vcenter/deployment/install?action=check
```

The following example shows syntax that you can use in the body of the HTTP request.

```
{
  "spec": {
    "vcsa_embedded": {
        "ceip_enabled": true,
        "standalone": {
        "sso_domain_name": "vsphere.local",
        "sso_admin_password": "<your_password>"
      }
    },
    "auto_answer": true
  }
}
```

If the input is valid, you receive the following response.

```
{
    "status": "SUCCESS"
}
```

3 Initiate the setup process by providing valid input in the body of the HTTP request.

```
POST https://<server>:5480/rest/vcenter/deployment/install?action=start
```

4 Monitor the progress of the setup process.

```
GET https://<server>:5480/rest/vcenter/deployment
```

The following example shows part of the response body when the setup process is ongoing.

```
{
    "progress": {
        "completed": 2,
        "message": {
            "id": "install.ciscommon.component.starting",
            "args": [
                "VMware Authentication Framework"
            ],
            "default_message": "Starting VMware Authentication Framework..."
        },
        "total": 3
    },
    "status": "RUNNING",
    "state": "CONFIG_IN_PROGRESS",
    "operation": "INSTALL",
...
```

The following example shows part of the response body when the setup process has completed successfully.

```
{
    "subtask_order": [
        "rpminstall",
        "validate",
        "firstboot"
    ],
    "cancelable": false,
    "progress": {
        "completed": 3,
        "total": 3,
        "message": {
            "default_message": "Task has completed successfully.",
            "id": "com.vmware.vcenter.deploy.task.complete.success",
            "args": []
        }
    },
    "status": "SUCCEEDED",
    "description": {
        "default_message": "Install vCenter Server.",
        "id": "com.vmware.vcenter.deploy.task.description.op.install",
        "args": []
    },
    "state": "CONFIGURED",
...
```

Results

You successfully configured the newly deployed vCenter Server instance.

File-Based Backup and Restore of vCenter Server

You can back up a vCenter Server instance and later restore the instance from the backup copy.

Backing up vCenter Server

The vCenter Server Management API supports backing up key parts of the vCenter Server instance. This allows you to protect vCenter Server data and to minimize the time required to restore data center operations.

The backup process collects key files into a tar bundle and compresses the bundle to reduce network load. To minimize storage impact, the transmission is streamed without caching in the vCenter Server instance. To reduce total time required to complete the backup operation, the backup process handles the different components in parallel.

You have the option to encrypt the compressed file before transmission to the backup storage location. When you choose encryption, you must supply a password which can be used to decrypt the file during restoration.

The backup operation always includes the vCenter Server database and system configuration files, so that a restore operation has all the data needed to re-create an operational vCenter Server instance. Current Alarms are included as well. You also have the option to specify additional data sets, called parts. In this release, you can specify a data set that includes Statistics, Events, and Tasks.

Backup and Restore Protocols for vCenter Server

The vCenter Server backup and restore feature supports a number of plug-in communication protocols.

Choose one of these protocols as the backup location type when you invoke the operation.

- FTP
- FTPS
- SCP
- HTTP
- HTTPS
- NFS
- SMB

The value `PATH` for the location type field indicates a locally mounted volume.

Note If you specify the `SCP` protocol, you must specify an absolute path as the value of the location type field when you create the backup job.

Calculate the Size Needed To Store the Backup File

When you prepare to do a backup of a vCenter Server instance, you can use the API to calculate the storage space needed for the backup file.

You can do this task when you are choosing a backup storage location or whenever your existing storage location might be approaching full capacity.

Prerequisites

- Verify that you have a vCenter Server instance running.
- Verify that you are familiar with authentication methods. See Chapter 2 Authentication Mechanisms.

Procedure

1. Authenticate to the vSphere Automation API endpoint and establish a session.
2. Request a list of backup parts available.
3. For each available backup part, request the size of the backup file.

   The backup process calculates the compressed size of each backup part.
4. Choose which parts to include in the backup, and sum their sizes.

   The backup storage server must have sufficient space to contain the chosen parts.

What to do next

After you choose which backup parts you will store, and verify that the backup storage server has sufficient free space, you can launch a backup job. For information, see Back up a vCenter Server Instance by Using the API.

Back up a vCenter Server Instance by Using the API

You can use the Management API to create a backup of the vCenter Server database and key components of the vCenter Server instance.

This procedure explains the sequence of operations you use to create a backup file of the vCenter Server instance. You can do this as part of a regular maintenance schedule.

Prerequisites

- Verify that the vCenter Server instance is in a ready state. All processes with start-up type automatic must be running.
- Verify that no other backup or restore jobs are running.
- Verify that the destination storage location is accessible to the backup process.
- Verify that the path to the destination directory exists, as far as the parent directory.
- If the destination directory does not exist, the backup process creates it. If the directory does exist, verify that it is empty.
- Verify that the destination storage device has sufficient space for the backup file. For information about how to calculate the space needed for the backup file, see Calculate the Size Needed To Store the Backup File.

Procedure

1. Authenticate to the vSphere Automation API endpoint and establish a session.

2 Use the `RecoveryBackupJobBackupRequest` data structure to describe the backup operation.

   The data structure specifies several properties, especially the backup location, the protocol used to communicate with the storage server, the necessary authorization, and which optional parts of the database you want to back up. The core inventory data and Alarms are always backed up, but you can choose whether or not to back up Statistics, Events, and Tasks. Collectively, this optional part of the backup is referred to as `seat`.

3 Issue a request to start the backup operation.

4 From the response, save the unique job identifier of the backup operation.

5 Monitor the progress of the job until it is complete.

6 Report job completion.

Schedule a Backup Job

You can automate the backup process by creating a schedule that runs backup jobs at specific times.

You can keep existing backups on the backup server. The retention policy defines the maximum number of backups that the server keeps. You can also specify whether the backup job should run once, or on a recurring basis. The recurrence policy defines the days of the week and specific times at which the backup job is scheduled to run.

Prerequisites

- Verify that you can access the backup server and you have read and write permissions.
- Verify that you have established a connection to the vAPI services.

Procedure

1. Create a schedule by using the `POST https://<vCenter_Server_IP>/api/appliance/recovery/backup/schedules` HTTP request.
2. Specify the retention and recurrence information.
3. Create a schedule by specifying the backup location, user credentials to access the location, retention, and recurrence information.
4. Use the `RecoveryBackupSchedulesUpdateSpec` data structure and pass the updated information.
5. Get a backup schedule by passing a schedule ID.

What to do next

Run the backup job by using the schedule.

Restoring vCenter Server

The vCenter Server Management API supports restoring a vCenter Server instance from a backup copy. The API simplifies the process by unifying the handling of various components of vCenter Server in a single operation.

The process of restoring a vCenter Server instance from a backup has two phases.

1. Deploy a new vCenter Server instance. OVF deployment is described in the *vSphere Automation SDKs Programming Guide*.
2. Invoke the `restore` operation from the Management API to apply configuration settings and load the vCenter Server database from the backup file.

Note You cannot specify optional parts for the restore operation. The restore operation includes all optional parts, such as Events and Tasks, that were specified at the time when the backup file was created.

Authentication When Restoring a vCenter Server Instance

During the process of restoring a vCenter Server instance from a backup image, you cannot use token-based authentication. You must use basic authentication until the vCenter Server instance is fully configured.

When you restore your vCenter Server instance from a backup file, it begins in an unconfigured state. During this time, you must use basic authentication to access the Management API. When you use basic authentication, do not use the vSphere Automation API endpoint. Instead, you must connect your client to port 5480 of the vCenter Server instance.

When you use basic authentication, you must pass user name and password credentials with each request. Use credentials that are known to the guest operating system of the vCenter Server instance.

Availability of Services While Restoring a vCenter Server Instance

During the process of restoring the vCenter Server backup file, services in the vCenter Server instance must restart. While they are restarting, your API client receives an error message.

You can write your client to trap the error, but you have no way to know when the vCenter Server services are running again. To determine when the restore process is complete, you must retry the API connection until it succeeds, then request the status of the job.

Restore a vCenter Server Instance by Using the API

You can use the Management API of to restore a vCenter Server instance from a backup file containing the vCenter Server database and key components of the vCenter Server instance.

Prerequisites

- Verify that the backed up vCenter Server instance is powered off.
- A new vCenter Server instance must be deployed in an unconfigured state, except that it must have a fully qualified domain name or IP address that matches the old one.

- Verify that the new vCenter Server instance has the same build number as the one in the backup file.
- Verify that the new vCenter Server instance has a size equal to or greater than the old one. If the old vCenter Server instance was customized to exceed the largest template size, the new one must be customized to the same size.
- Verify that no other backup or restore jobs are running.
- Verify that the destination storage location is accessible to the vCenter Server restore process.

Procedure

1. Use the `RecoveryRestoreJobRestoreRequest` data structure to describe the restore operation.
2. Issue a request to start the restore operation.
3. Monitor the progress of the job until it is complete.
4. Report job completion.

What to do next

After the vCenter Server instance is fully configured by the restore operation, you can resume using the vSphere Automation API endpoint for subsequent operations.

Reconcile a vCenter Server Instance with Nodes in Embedded Linked Mode

You can run the reconciliation process after you successfully restored your vCenter Server instance. By using the API or HTTP requests, you can reconcile vCenter Server nodes that work in an embedded linked mode and are connected in a ring or daisy-chain.

Reconciliation is a post-restore process that checks whether the vCenter Server partners in embedded linked mode are available, synchronizes the vCenter Server data and services with the partners, and runs the vCenter Server services. The processes of restore and reconciliation depend on the topology and if there are changes in the topology between the backup and restore, you cannot restore the embedded linked mode. If the replication partners are not available and you try to restore the first node, you must ignore the warnings. In this case, any changes that are made in the topology or infrastructure after the backup will be lost. If you restore a node different from the first one, you must add it to the domain of the first node. If you use a daisy-chain topology, you must first restore the first node, and after that to restore the second, link it to the first one, and apply the same to the following nodes.

You can use the reconciliation API after a file-based and an image-based restore. After an image-based restore, you can run the reconciliation process by using the API or UI. After a file-based restore, you can monitor the reconciliation process by using the `GET https://<vcenter_ip_address_or_fqdn>/api/appliance/recovery/reconciliation/job` HTTP request. For information about how to restore a vCenter Server instance from an image or a file by using the UI, see the *vCenter Server Installation and Setup* documentation.

Prerequisites

- Verify that you successfully restored your node from an image.
- Verify that the replication partners are available.
- Verify that you restored your nodes in the correct order, if you use a daisy-chain topology.
- Verify that you have administrator's credentials to your Single Sign-On domain.
- Verify that there is no running or failed reconciliation job.

Procedure

1. Use the `RecoveryReconciliationJobCreateSpec` data structure, specify user name and password of Single Sign-On administrator, and set the `ignore_warnings` field to `true`.

   The default value of `ignore_warnings` is `false`. If you do not set `ignore_warnings` to `true`, the reconciliation fails due to the validation warnings.

2. Run a reconciliation job by using the `POST https://<vcenter_ip_address_or_fqdn>/api/appliance/recovery/reconciliation/job` HTTP request.

   You can check the result of the operation by reading the `RecoveryReconciliationJobInfo` data structure. `RecoveryReconciliationJobInfo` contains information about the job such as description, status, progress, error, start and end time.

3. Get the status of the job by using the `GET https://<vcenter_ip_address_or_fqdn>/api/appliance/recovery/reconciliation/job` HTTP request.

   The possible states are `NONE`, `RUNNING`, `FAILED`, and `SUCCEEDED`.

Troubleshooting for vCenter Server Installation or Deployment

You can use the API to perform troubleshooting operations related to the installation and deployment of vCenter Server.

Managing System Logs

You can automate the forwarding of vCenter Server system log messages to remote logging servers by using the vCenter Server Management API.

You can configure the syslog forwarding by using the API or user interface. For information about how to manage the syslog by using the user interface, see the *vSphere Monitoring and Performance* documentation.

Configuring Syslog Forwarding

You can use the vCenter ServerManagement API or HTTP requests to configure the forwarding of vCenter Server syslog messages and test the connection between the vCenter Server instance and remote servers.

The following table lists operations that you can perform to manage the forwarding of syslog messages to remote logging servers.

Table 10-6. User Operations

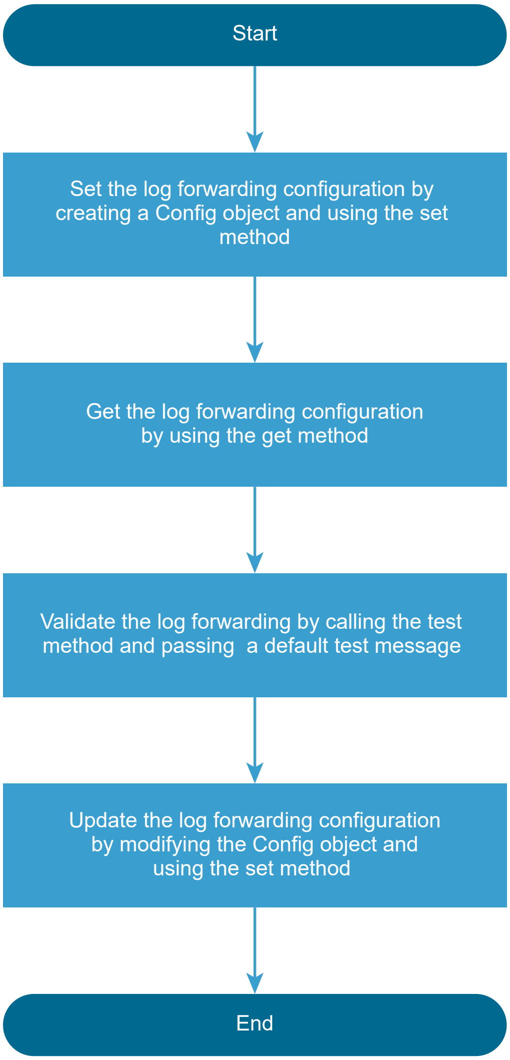
Operation	Description
Get forwarding configuration	You can retrieve information about the log forwarding configuration. See HTTP Requests for Configuring Syslog Forwarding.
Test forwarding configuration	You can validate the current log forwarding configuration. Optionally, you can send a test diagnostic log message from the vCenter Server instance to all configured logging servers to allow manual end-to-end validation. See HTTP Requests for Configuring Syslog Forwarding.
Set forwarding configuration	You can change the log forwarding configuration. See HTTP Requests for Configuring Syslog Forwarding.

The forwarding configuration includes the IP or FQDN of the remote server, the remote port for receiving syslog information, and the communication protocol. The remote server must be a server with running `rsyslog`, for example, another vCenter Server instance. The API supports the TCP, UDP, TLS, and RELP protocols. For information about the supported TLS versions, see KB article 2147469. By creating a `Forwarding.Config` object, you specify the connection with a remote server. For information about the `Forwarding` class and its methods, see the *API Reference* documentation and *Example Configuration Workflow*.

You can use several remote servers by creating a list with `Forwarding.Config` objects and passing it to the `set` method. The maximum number of remote servers is three. You can validate the forwarding configuration by using the `test` method. The returned `Forwarding.ConnectionStatus` object shows the status of the connection between the vCenter Server instance and a remote server. The `State` enumeration shows whether the vCenter Server instance can reach the remote server. `State` can be `UP`, `DOWN`, or `UNKNOWN`. If the state is `DOWN` or `UNKNOWN`, the vCenter Server instance cannot access the remote server and you must check the remote server and its settings such as network ports, firewall, supported protocols, and syslog configuration.

Note If you use UDP, the connection status is always `UNKNOWN`.

Figure 10-4. Example Configuration Workflow


Start
Set the log forwarding configuration by creating a Config object and using the set method
Get the log forwarding configuration by using the get method
Validate the log forwarding by calling the test method and passing a default test message
Update the log forwarding configuration by modifying the Config object and using the set method
End

HTTP Requests for Configuring Syslog Forwarding

By using the API or HTTP requests, you can set and get the forwarding configuration, check the connection with the remote server or servers, and exchange test messages with them.

The following HTTP requests show the syntax that you can use to perform the available user operations.

Note When you send the requests, you must authenticate with vCenter Server **root** credentials.

```
GET https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/logging/forwarding
```

```
POST https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/logging/forwarding?action=test
```

```
PUT https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/logging/forwarding
```

For information about the body of each HTTP request, see the *REST API Reference* documentation.

Table 10-7. HTTP Status Codes

HTTP Status Code	Description	Operations That Return the Status Code
200	The operation is successful.	All available operations. You can check the returned data in the results data structure.
400	You use an invalid argument. For example, a protocol that it is not supported, invalid port number, or the number of configurations is greater than 3.	Set forwarding configuration.
401	You use invalid user name or password, or authentication is failed.	All available operations.
500	There is a `vapi std` error. For information about the types of `vapi std` errors, see the `vapi.std.errors` API in the *API reference* documentation.	Set forwarding configuration.

Managing Support Bundles

You can retrieve information about support bundles and create support bundles.

The following table lists the operations that you can perform to manage support bundles.

Table 10-8. User Operations

Operation	Description
Get support bundle components	You can retrieve a list of components and manifests included in the support bundle. Manifests specify the files that must be collected for a component as part of the support bundle.
List support bundles	You can retrieve a list of generated support bundles. The list contains details about each support bundle.
Create a support bundle	You can generate a support bundle. Optionally, you can specify the support bundle components and the partition where you want to save the support bundle.

You can run support bundle management operations by using the vSphere Automation SDK or sending an HTTP request. For information about the HTTP requests that you can use to perform the user operations, see HTTP Requests for Support Bundle Management Operations.

HTTP Requests for Support Bundle Management Operations

You can use HTTP requests to perform support bundle management operations.

The following HTTP requests show the syntax that you can use to perform the available user operations.

- Get support bundle components

  ```
  GET https://<vcenter_ip_address_or_fqdn>:443/api/appliance/support-bundle/components
  ```

- List support bundles

  ```
  GET https://<vcenter_ip_address_or_fqdn>:443/api/appliance/support-bundle
  ```

- Create a support bundle

  ```
  POST https://<vcenter_ip_address_or_fqdn>:443/api/appliance/support-bundle
  ```

For information about the content and syntax of the HTTP request body for each operation, see the *API reference* documentation.

Example Input and Output for Support Bundle Management Operations

The following examples provide sample input and output values for the support bundle management operations.

Example: Get Support Bundle Components

This operation does not require input.

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/appliance/support-bundle/components
```

Type	Example

Output	`{`
	`"Group1" : List of manifests under this group,`
	`"Group2": ...`
	`}`

Example: List Support Bundles

This operation does not require input, but supports optional query parameters.

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/appliance/support-bundle
```

Type	Example

Input	`"IterationSpec":{`
	`"size" : 10`
	`}`
	`FilterSpec:{`
	`"creationStatus" : "SUCCEEDED",`
	`"available" : True/False`
	`}`

Output	`{`
	`"Description": "Reason for generation",`
	`"Generation time": 2 Mar 2021,`
	`"URL" : "Download URL",`
	`"Downloadable" : True/False`
	`"Expiration time": "URL expiration time"`
	`}`

Example: Create a Support Bundle

This operation requires description input and supports optional body parameters. As output, you receive a task ID, which you can use to track the progress of the task. When the operation is completed, you receive the download URL in the task response.

```
POST https://<vcenter_ip_address_or_fqdn>:443/api/appliance/support-bundle
```

Type	Example

Input	`{`
	`"Description" : "Reason",`
	`"Inclusions" : List of components to be included in the bundle`
	`<Optional> path: partition to be used.`
	`}`

Output	Task ID

vCenter Server Upgrade

You can use the API to perform operations related to stage 2 of the upgrade process. You can also perform historical data transfer operations.

Upgrade Stage 2

You can upgrade your vCenter Server instance by using the API, CLI, or GUI.

For information about how to upgrade the vCenter Server instance by using CLI and GUI, see the *vCenter Server Upgrade* documentation.

Upgrading a vCenter Server Instance

You can use the API during stage 2 of the vCenter Server instance upgrade.

By using the API, you can upgrade your vCenter Server instance. For information about the upgrade process, its stages, supported configurations, upgrade paths, prerequisites for upgrading, and the sequence for upgrading a vSphere environment, see the *vCenter Server Upgrade* documentation.

After you deploy the vCenter Server instance on stage 1 by using the GUI or CLI, the instance enters in an `INITIALIZED` state. If the vCenter Server instance is not initialized, you cannot run stage 2 of the upgrade process. You can get the state of the vCenter Server instance by using the `vcenter deployment` service. There are six states during the upgrade process.

Figure 10-5. Upgrade Stage 2 State Diagram

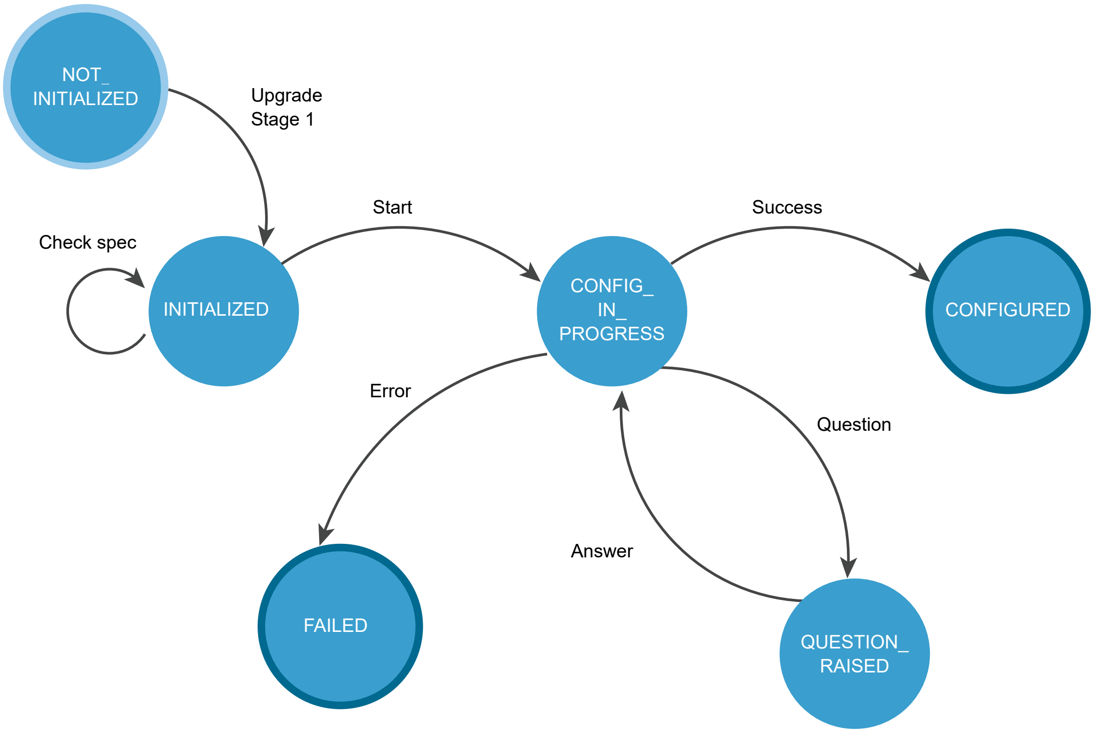


Table 10-9. vCenter Server Instance States During Upgrade Stage 2

State	Description
`NOT_INITIALIZED`	The upgrade stage 1 phase is in progress, not started, or failed.
`INITIALIZED`	The vCenter Server instance is deployed and ready for upgrading.
`CONFIG_IN_PROGRESS`	The upgrade process is in progress.
`QUESTION_RAISED`	You must answer the question to continue the upgrade process. The vCenter Server instance stays in the `QUESTION_RAISED` state until it receives the correct answer.

Table 10-9. vCenter Server Instance States During Upgrade Stage 2 (continued)

State	Description
`FAILED`	Errors appeared during the upgrade process. You can check the errors, warnings, and info data structures.
`CONFIGURED`	The vCenter Server instance is upgraded or configured successfully.

`FAILED` and `CONFIGURED` are final states.

You can roll back a vCenter Server instance upgrade by using the GUI. For information about how to roll back a vCenter Server instance, see the *vCenter Server Upgrade* documentation.

After the upgrade, you can check the vCenter Server instance type, domain registration, services, their state and health status by using the API. For information about how to verify whether the upgrade of your vCenter Server instance is successful, see the *vCenter Server Upgrade* documentation.

Table 10-10. User Operations shows operations that you can perform to upgrade your vCenter Server instance.

Table 10-10. User Operations

Operation	Description
Operations for upgrading	
Get the state of the vCenter Server instance	You can get the state of the vCenter Serverinstance before, during and after the upgrade process. For information about the HTTP requests for diagnostic and deferring the transfer of historical data, see HTTP Requests for Upgrade Stage 2.
Check	You can validate the upgrade spec before you run the upgrade process. If the vCenter Server instance is in the `INITIALIZED` state, you can run the validation. The operation runs upgrade pre-checks. You can check the errors, warnings, and status data structures before you run the upgrade process. For information about the HTTP requests for upgrading, see HTTP Requests for Upgrade Stage 2.
Start	If the vCenter Server instance is in an `INITIALIZED` state, you can run the upgrade process. If errors appear during the upgrade, you can download the vCenter Server support bundle. For information about the HTTP requests for upgrading, see HTTP Requests for Upgrade Stage 2.
Get	If the vCenter Server instance is in a `CONFIGURED` state, you can get the `spec` that is used for upgrading. For information about the HTTP requests for upgrading, see HTTP Requests for Upgrade Stage 2.
Operations for getting and answering a question	

Table 10-10. User Operations (continued)

Operation	Description
Get	You can get the raised question. If you set the `Upgrade.auto_answer` to `true`, the upgrade process will be in a `silent` mode and the vCenter Serverinstance does not generate questions. It uses default answers and you should not provide an answer. For information about the HTTP requests for getting and answering a question, see HTTP Requests for Upgrade Stage 2.
Answer	You can provide an answer to the raised question. The available answers for the upgrading are `OK`, `CANCEL`, `YES`, `NO`, `ABORT`, `RETRY`, and `IGNORE`. The answer depends on the type of the question. If you set the `Upgrade.auto_answer` to `true`, the upgrade process will be in a `silent` mode and the vCenter Serverinstance does not generate questions. It uses default answers and you should not provide an answer. For information about the HTTP requests for getting and answering a question, see HTTP Requests for Upgrade Stage 2.

For information about the available operations in the API, see the `vcenter deployment`, `vcenter deployment upgrade`, `vcenter services`, and `vcenter system-config deployment type` services in the *API reference* documentation.

You can upgrade your vCenter Server instance by using HTTP requests. For information about the HTTP requests, see HTTP Requests for Upgrade Stage 2.

Workflows for Upgrade Stage 2

You can use the `vcenter deployment` API to run the upgrade stage 2 process of your vCenter Server instance.

Figure 10-6. Upgrade Workflow and Figure 10-7. Upgrade Stage 2 Workflow show example upgrade workflows.

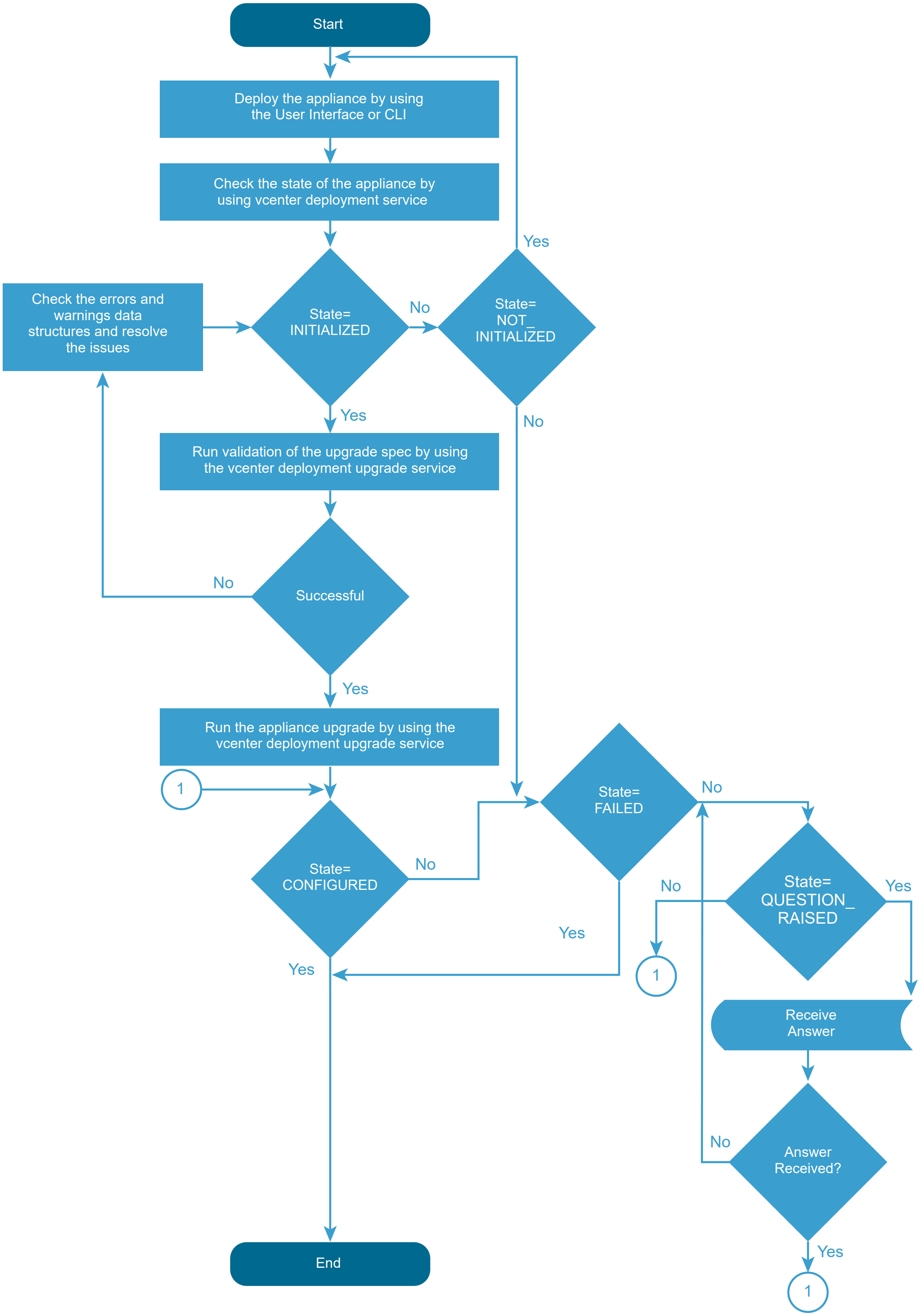
During stage 1, the vCenter Server instance is in a `NOT_INITIALIZED` state. After a successful deployment, the vCenter Server instance enters in an `INITIALIZED` state. If there are errors during stage 1, the vCenter Server instance stays in a `NOT_INITIALIZED` state and you must redeploy it.

You can check the state of the vCenter Server instance before, during, and after the upgrade process. You can run the upgrade stage 2 process if the vCenter Server instance is initialized. You can check the upgrade configuration before you run the upgrade by running pre-checks. If errors or warnings appear during the validation of the upgrade specification, you must remove the causes and correct the specification.

During the upgrade process, the vCenter Server instance can enter in a `FAILED` or `QUESTION_RASED` state. If a question appears during the upgrade, the vCenter Server instance enters in a `QUESTION_RAISED` state and stays in it until you provide an answer. You can run the upgrade in silent mode, in which the vCenter Server instance does not generate questions, and uses default answers.

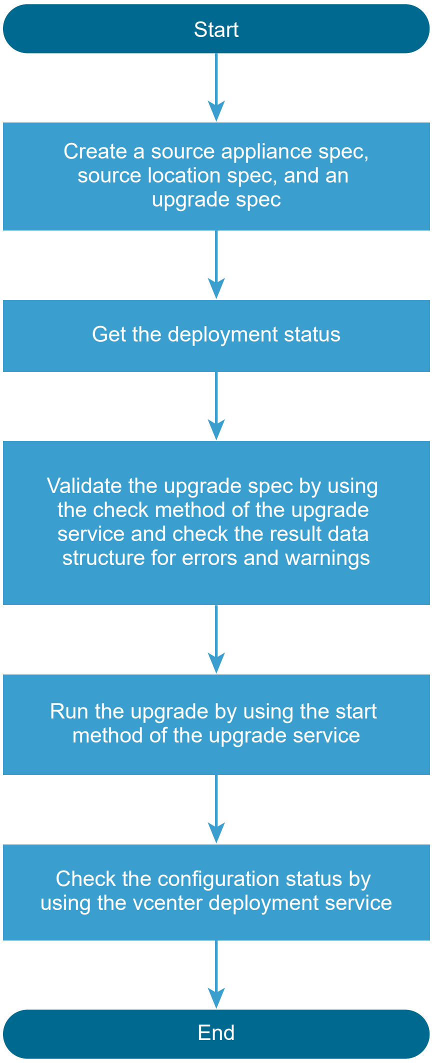
If errors appear during the upgrade, the vCenter Server instance enters in a `FAILED` state and you must remove the causes, redeploy the instance and restart the upgrade. If stage 2 of the upgrade process is successful, the vCenter Server instance enters in a `CONFIGURED` state. If the vCenter Server instance is configured, you can check its services and pause the historical data transfer.

Figure 10-6. Upgrade Workflow


Start
Deploy the appliance by using the User Interface or CLI
Check the state of the appliance by using vcenter deployment service
Yes
Check the errors and warnings data structures and resolve the issues
State= INITIALIZED
No
State= NOT_ INITIALIZED
Yes
No
Run validation of the upgrade spec by using the vcenter deployment upgrade service
No
Successful
Yes
Run the appliance upgrade by using the vcenter deployment upgrade service
1
State= FAILED
No
State= CONFIGURED
No
No
State= QUESTION_ RAISED
Yes
Yes
Yes
1
Receive Answer
No
Answer Received?
End
Yes
1

For information about the states of the vCenter Server instance and available operations, see Upgrading a vCenter Server Instance.

Figure 10-7. Upgrade Stage 2 Workflow


Start
Create a source appliance spec, source location spec, and an upgrade spec
Get the deployment status
Validate the upgrade spec by using the check method of the upgrade service and check the result data structure for errors and warnings
Run the upgrade by using the start method of the upgrade service
Check the configuration status by using the vcenter deployment service
End

You can run the upgrade pre-checks and the upgrade stage 2 process by creating and passing an `UpgradeSpec`. In `UpgradeSpec`, you define the upgrade configuration and specify the source vCenter Server instance and the source ESXi host in `SourceApplianceSpec` and `LocationSpec`. See Figure 10-7. Upgrade Stage 2 Workflow . You can run the upgrading in silent mode by setting the `UpgradeSpec.auto_answer` to `true`. The default value of `UpgradeSpec.auto_answer` is `false` and the upgrading is in interactive mode, in which you must provide answers to the raised questions.

For information about the classes, variables, and default values, see the *API reference* documentation.

HTTP Requests for Upgrade Stage 2

You can use HTTP requests or the API to upgrade your vCenter Server instance.

HTTP Requests

HTTP requests for upgrading

- Get the state of the vCenter Server instance

  ```
  GET https://<IP_address_upgraded_or_target_instance>:5480/rest/vcenter/deployment
  ```

- Validate the upgrade spec

  ```
  POST https://<IP_address_target_instance>:5480/rest/vcenter/deployment/upgrade?action=check
  ```

- Run the upgrading

  ```
  POST https://<IP_address_target_instance>:5480/rest/vcenter/deployment/upgrade?action=start
  ```

- Get the upgrade spec used for upgrading

  ```
  GET https:// https://<IP_address_upgraded_instance>:5480/rest/vcenter/deployment/upgrade
  ```

HTTP requests for getting and answering a question

- Get the raised question

  ```
  GET https://<IP_address_upgraded_or_target_instance>:5480/rest/vcenter/deployment/question
  ```

- Answer to the question

  ```
  POST https://<IP_address_upgraded_or_target_instance>:5480/rest/vcenter/deployment/
  question?action=answer
  ```

For information about the content and syntax of the HTTP request body, see the *API reference* documentation.

Note When you send the requests, you must authenticate with vCenter Server**root** credentials.

HTTP Status Codes and Errors

Table 10-11. HTTP Status Codes and Errors

HTTP Status Code	Description	Operations That Return the Status Code
200	The operation is successful.	All operations. You can check the returned data in the results data structure.
400	You cannot perform the operation because the vCenter Server instance is in the current state. For information about the states of the vCenter Serverinstance, see the Upgrading a vCenter Server Instance topic.	■ Get the state of the vCenter Server instance ■ Validate the upgrade spec ■ Run the upgrade ■ Get the upgrade spec ■ Get the raised question ■ Answer the question

Table 10-11. HTTP Status Codes and Errors (continued)

HTTP Status Code	Description	Operations That Return the Status Code
401	You use invalid user name or password, or authentication is failed.	■ Get the state of the vCenter Server instance ■ Validate the upgrade spec ■ Run the upgrade ■ Get the upgrade spec ■ Get the raised question ■ Answer the question
404	The state of the vCenter Server instance cannot be determined	Get the state of the vCenter Server instance.
500	There is a `vapi std` error. For information about the types of `vapi std` errors, see the `vapi.std.errors` API in the *API reference* documentation.	Get the raised question.

If errors appear during the upgrade process, you can check the `results` data structure, the API log file, and download the vCenter Server support bundle from `https://`*`<vcenter_ip_address_or_fqdn>`*`:443/appliance/support-bundle`.

Historical Data Transfer

If you migrate vCenter Server for Windows, you can transfer the historical data of your source vCenter Server instance together with the core configuration data.

Deferred Import

The deferred import is a process of historical data transfer after the successful migration of a vCenter Server instance with an external database. The historical data includes statistics, events, and tasks.

By using the deferred import feature, you can postpone the historical data transfer after the migration process completes, so that you manage the downtime of your environment. You can select whether all historical data, or only events and tasks, will be migrated with the core data during the migration. The historical data transfer and deferred import of historical data are deactivated by default. You can enable and configure the historical data transfer by using the API, vCenter Server Management Interface, vCenter Server installer, or CLI installer. A vCenter Server super administrator can run and control the migration and deferred import processes.

If you use the deferred import feature, the historical data is migrated with the core data and the historical data import process starts automatically after a successful migration and when the vCenter Server instance is running. You can pause the historical data import and resume it later.

For information about how to configure and run the migration and deferred import processes by using the vCenter Server Management Interface, see the *vCenter Server Upgrade* documentation.

By using the API, you can configure, control, and monitor the data transfer process. If you use the API to enable the deferred import feature, you must create a history migration spec and set `defer_import` to `true`. For information about how to configure the deferred import by using the API, see the *API reference*.

The data import process has five states that you can check. If the historical data migration and the deferred import are configured, the historical data import starts automatically after a successful migration.

Figure 10-8. Deferred Import State Diagram

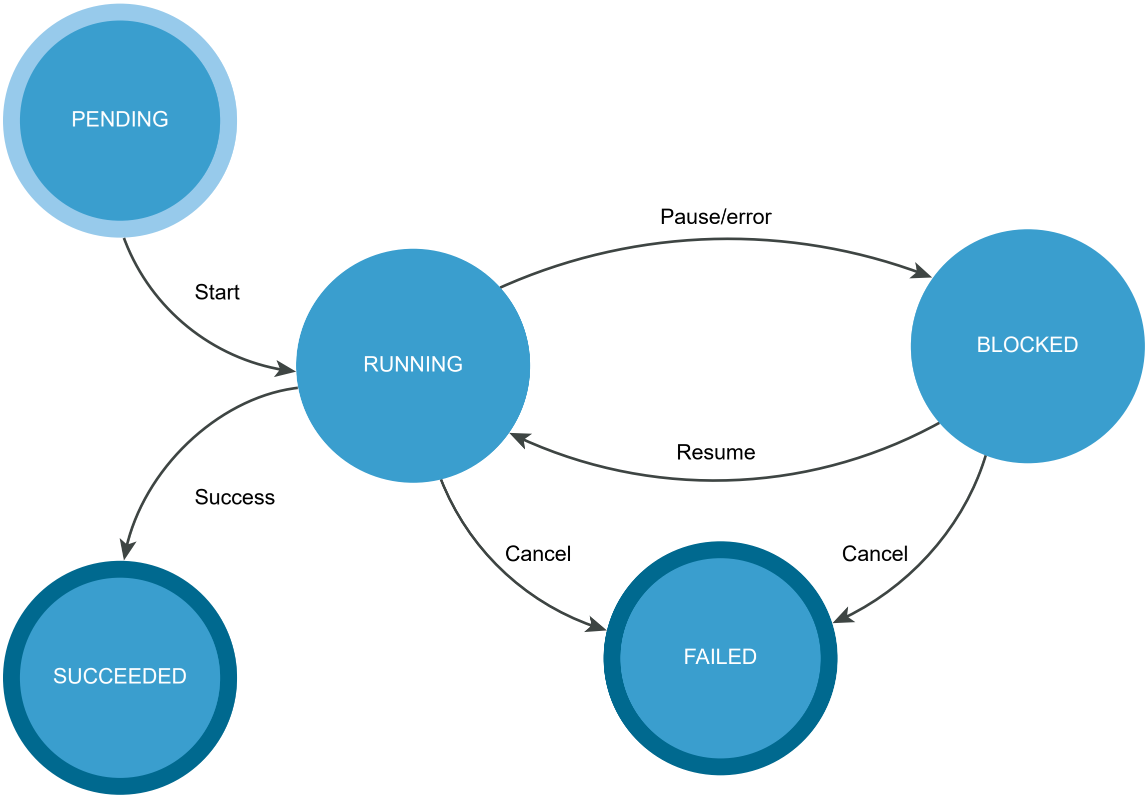


Table 10-12. User Operations

Operation	Description
Pause	You can pause the running data transfer process. If the data transfer is paused, you can resume or cancel it.
Cancel	You can cancel the data transfer process if it is in a `RUNNING` or `BLOCKED` state. **Note** If you cancel the data transfer, the process enters in a final `FAILED` state and you cannot resume the transfer.

Table 10-12. User Operations (continued)

Operation	Description
Resume	You can resume the stopped data transfer.
Get status	You can retrieve the status of the data transfer process. There are five states. **`PENDING`** The transfer is not started. **`RUNNING`** The transfer is started or resumed. **`BLOCKED`** The transfer is paused or there was a recoverable error, such as not enough disk space, during the import. **`SUCCEEDED`** The transfer is successful. **`FAILED`** The transfer is canceled.

You can run the deferred import operations by using the API or sending an HTTP request.

Note When you send the requests, you must use an authentication.

If you send requests to port `5480`, you must authenticate with vCenter Server **root** credentials. If you send requests to the vCenter Server reverse proxy port, you must authenticate with vCenter Single Sign-On credentials.

The following HTTP requests show the syntax that you can use to perform the available user operations.

```
https://<IP_of_migrated_instance>:5480/rest/vcenter/deployment/history?action=pause
```

```
https://<IP_of_migrated_instance>:5480/rest/vcenter/deployment/history?action=cancel
```

```
https://<IP_of_migrated_instance>:5480/rest/vcenter/deployment/history?action=resume
```

```
https://<IP_of_migrated_instance>:5480/rest/vcenter/deployment/history
```

```
https://<IP_of_migrated_instance>/api/vcenter/deployment/history?action=pause
```

```
https://<IP_of_migrated_instance>/api/vcenter/deployment/history?action=cancel
```

```
https://<IP_of_migrated_instance>/api/vcenter/deployment/history?action=resume
```

```
https://<IP_of_migrated_instance>/api/vcenter/deployment/history
```

For information about the historical data transfer errors, see Historical Data Import Errors.

If you pause the data transfer by using the API or an HTTP request, you can resume or cancel the process by using the API or the vCenter Server Management Interface.

Important If you cancel the transfer process, and want to transfer the historical data later, you must restart the migration process.

Historical Data Import Errors

If an error appears during the data import, the import stops and the process enters in a `BLOCKED` state. You can resume the data import after you eliminate the cause.

You can check the errors, warnings, and info messages by reading the `info`, `status`, and `notifications` data structures.

If the information in the data structures is not enough, you can download the vCenter Server support bundle from `<vcenter_ip_address_or_fqdn>:443/appliance/support-bundle` and check the log files.

Table 10-13. Log Files

Log File	Path
API log file	`/var/log/vmware/applmgmt/applmgmt.log`
Backend log file	`/var/log/vmware/upgrade/upgrade-post-import.log`

Table 10-13. Log Files (continued)

Log File	Path
Upgrade Runner log file	`/var/log/vmware/upgrade/deferredimport-upgrade-runner.log`
Deferred import log file	`/var/log/vmware/upgrade/DeferredImport_com.vmware.vcdb_<date_time>.log`

vCenter Server Configuration

You can use the API to perform operations related to health monitoring and capacity monitoring of vCenter Server. You can also manage the global FIPS compliance and perform infrastructure profile management operations.

Health Monitoring of vCenter Server

The vCenter Server API offers health status indicators for several key components of the system. These indicators can be polled periodically to monitor the components for problems.

The health status indicators report graded values from green to red. The general meanings of the grades are as follows.

green

The component is healthy.

yellow

The component is healthy, but may have some problems.

orange

The component is degraded, and may have serious problems.

red

The component is unavailable, or will stop functioning soon.

gray

No health data is available.

Check Overall System Health of vCenter Server

vCenter Server provides a composite health indicator that enables you to test a single value that represents the health of all system components. This procedure shows how to test the composite health indicator.

The value of the overall system health indicator reflects the strongest trouble indication among the vCenter Server components. If any component has a `red` indicator, the overall system health indicator is `red`, else if any component has an `orange` indicator, the overall system health indicator is `orange`, and so on.

A `gray` value for any component indicates that data for the subsystem is unknown. If one or more components have a `gray` value, but all other subsystems are `green`, then the overall system health indicator is `gray` rather than `green`. However, if any component has a definite trouble indication, the overall system health indicator reflects the strongest trouble indication among the components.

Prerequisites

Verify that you have an active authenticated session with vCenter Server. This procedure assumes that the session ID is present in the security context of a stub configuration.

Procedure

1. Create an interface stub or REST path that uses the stub configuration.
2. Invoke the `health.system` method.
3. Format and display the resulting value.

Capacity Monitoring of vCenter Server

vCenter Server keeps a history of statistics that you can use to monitor resources used by the vCenter Server instance.

You can use the statistics to spot peak usage demands or to monitor trends for advance warning of potential resource exhaustion.

Frequency and Retention of Statistics Collection in vCenter Server

vCenter Server collects statistics from the guest operating system at regular intervals and stores them in a database. Users can query the statistics in the database by selecting a time period and a roll-up function that vCenter Server applies to the statistics before returning them to the client.

After the monitoring service starts up, it begins requesting statistics from the guest operating system periodically, at a frequency that depends on the type of statistic. The service requests storage statistics once every 10 minutes, while it requests memory, CPU, and networking statistics once per minute. The collection times are fixed relative to the startup time of the monitoring service, rather than to clock time.

The monitoring service retains statistics approximately 13 months, by default. Older statistics are deleted by the service, creating a 13-month moving window within which you can query statistics. You can choose to delete statistics as needed to conserve storage resources.

Nature of Statistics in vCenter Server

vCenter Server supplies statistics of several types.

The guest operating system computes statistics either as rates, such as CPU cycles per second, or as snapshots of size, such as KB used for storage. Statistics stored as size snapshots are collected at the end of their sample periods. Statistics stored as rates are computed as averages of values sampled frequently during each sample period.

When you query the statistics database, the units are not returned with the data, but you can determine the units for any metric by requesting metadata for the metric with the `get()` method.

Requesting Statistics from vCenter Server

To request statistics, you must construct an appropriate request structure to filter statistics from the database.

To request data or metadata for a metric, you must supply the ID of the metric. You can get a list of metric IDs by using the `list()` method, which returns information on all available metrics.

When you query statistics, you provide a list of IDs to specify the metrics in which you are interested. You also supply a start time, an end time, a roll-up interval, and a roll-up function. These values interact as follows to determine the data returned to you.

- The response contains a list of data points for each metric ID you specified in the request.
- The start time and end time control the limits for the data you want in the response. The response contains data points only for statistics that have timestamps between those limits, inclusive of the endpoints. However, the start time is adjusted to a round number, in some cases. For more information, see Statistics Interval Adjustment in vCenter Server.
- The roll-up interval enables you to control the granularity of the data points in the response. Rather than a response with a data point for every statistic between the start time and end time, you get a response with a number of data points equal to the number of intervals between the start and end times. Generally, you should specify a time period that is an even multiple of the interval, so that each data point in the response represents the same number of statistics.
- The roll-up function specifies how the response summarizes the statistics that fall within each interval. The resulting data point can be the maximum statistic value within collection interval, or the mean of the statistics values within the interval, and so on.

Statistics Collection Times

The actual time that a statistic was collected is not readily predictable.

The API does not enable you to determine the exact time that a statistic was collected. Furthermore, some statistics, such as those for storage metrics, might take seconds or minutes to collect, so that they are not available immediately at the time a request is made to the guest operating system.

However, because statistics are collected at regular intervals, and roll-up intervals for a request generally all have the same size, each data point in the response represents the same number of statistics as the others. See Statistics Interval Adjustment in vCenter Server for more information.

Statistics Interval Adjustment in vCenter Server

When you make a request for statistics, the monitoring service might adjust the specified roll-up interval times to improve the appearance of statistics graphs in a graphical interface.

The monitoring service adjusts the start time of a data collection request when it is not an exact multiple of the interval length. In these instances, the start time is rounded downward to the previous UTC time that is a multiple of the interval. All subsequent intervals of the data collection are also adjusted to align with the new start time.

For example, if the start time is 10:31 and the interval length is 1 hour, the monitoring service adjusts the start time to 10:00 and the roll-up intervals have the following continuous pattern.

- 10:00 to 10:59:59.999
- 11:00 to 11:59:59.999
- 12:00 to 12:59:59.999

The monitoring service does not adjust the end time of a data collection. Consequently, the response to a statistics query might contain one more data value than expected, or an incomplete final interval might be lengthened.

Empty Data Values

In some instances, you might encounter a response that reports an empty data value, or even a series of empty data values. This might manifest as a list of data values containing some numeric values alternating with empty values.

- Empty data values can happen when the report time period is too short to be certain of containing any statistics. For instance, a time period of 30 seconds is half the length of the sample period for network metrics, so you have only a 50% chance of finding a network statistic during any 30-second reporting period.
- Empty data values can also happen when the interval is shorter than the sample period for a metric you have requested. In this case, some data points are present in the list, while others are empty because no statistic was collected during those intervals. For instance, an interval of 5 minutes is only half the length of the sample period for storage metrics, so every second data value is empty.
- Empty data values can also happen when the monitoring service has not finished collecting and writing the last sample to the database, even if the nominal sample timestamp falls within the report time period. For example, calculation of storage used can delay writing a storage statistic to the database. A request for the statistic during that delay time produces an empty data point in the response.

When a response contains an empty data value, this indicates that no statistics were collected during a collection interval. An appropriate action for the client in such a case depends on how the client is using the data. For example, if you are graphing a resource usage trend, you might choose to interpolate for the missing value to produce a smooth line.

Check Database Usage in vCenter Server

vCenter Server contains a database of all objects managed by the vCenter Server instance. In addition to inventory objects, the database includes vCenter Server statistics, events, alarms, and tasks. You can calculate the database storage consumption by adding the sizes of all data categories.

You need to monitor storage consumption in vCenter Server.

Prerequisites

This task assumes you have previously authenticated and created a client session.

Procedure

1 Prepare a request for database usage statistics.

  Include metric IDs both for `vcdb_core_inventory` and `vcdb_seat`. The name `vcdb_seat` refers to Statistics, Events, and Tasks in the vCenter Server database.

2 Issue the request to the API endpoint.

3 Process the resulting data points as needed.

4 Format and print the results.

Results

The result of this procedure shows the storage used in the vCenter Server database, which includes storage overhead used for indexes and other optimizations beyond the actual size of the data.

List Storage Consumption By Data Type in vCenter Server

vCenter Server provides statistics on several types of storage.

For example, you can query statistics about inventory storage, transaction log, and vCenter Server tasks. Many of these statistics are available both for storage consumed and storage available.

This task provides data for system administrators who need to monitor storage consumption in the guest operating system of vCenter Server.

Prerequisites

Verify that you have authenticated and created a client session.

Procedure

1 Prepare a request for database usage statistics.

  Include metric IDs for each data type you want to monitor.

2 Issue the request to the API endpoint.

3 Process the resulting data points as needed.

4 Format and print the results.

Managing the Global FIPS Compliance

You can retrieve information about the current FIPS (Federal Information Processing Standards) settings of vCenter Server. You can also enable or deactivate the global FIPS compliance.

FIPS 140-2 is a U.S. and Canadian government standard that specifies security requirements for cryptographic modules. vSphere uses FIPS-validated cryptographic modules to match those specified by the FIPS 140-2 standard. The goal of vSphere FIPS support is to ease the compliance and security activities in various regulated environments.

The following table lists the operations that you can perform to manage the FIPS settings of your vCenter Server system.

Table 10-14. User Operations

Operation	Description
Get FIPS status	You can check whether the global FIPS compliance is currently enabled on the vCenter Server system.
Manage FIPS status	You can enable or deactivate the global FIPS compliance on the vCenter Server system.

Note When you enable FIPS compliance, some components might present functional constraints. For more information, see *vSphere Security*.

You can run FIPS management operations by using the vSphere Automation SDK or sending an HTTP request. For information about the HTTP requests that you can use to perform the user operations, see HTTP Requests for Global FIPS Compliance Operations.

HTTP Requests for Global FIPS Compliance Operations

You can use HTTP requests to perform global FIPS compliance management operations.

The following HTTP requests show the syntax that you can use to perform the available user operations.

- Get FIPS status

  ```
  GET https://<vcenter_ip_address_or_fqdn>:443/api/appliance/system/global-fips
  ```

- Manage FIPS status

  ```
  PUT https://<vcenter_ip_address_or_fqdn>:443/api/appliance/system/global-fips
  ```

  You can enable the global FIPS compliance by using the following request body.

  ```
  {
      "enabled": true
  }
  ```

You can deactivate the global FIPS compliance by using the following request body.

```
{
    "enabled": false
}
```

After you change the global FIPS compliance, the vCenter Server system reboots to apply the change.

Performing Infrastructure Profile Management Operations

You can export an existing vCenter Server configuration and import it to other vCenter Server instances.

You can export multiple configuration profiles at once. The exported data can contain general configuration settings and user content. You can replicate the same configuration across all vCenter Server instances in your environment by importing the same data package. You can also use the exported data as a backup if you need to revert to the last known good configuration. To avoid configuration issues, you can validate the exported data before importing it to a vCenter Server instance.

The following table lists the operations that you can perform to manage the configuration profiles in your infrastructure.

Table 10-15. User Operations

Operation	Description
List configuration profiles	You can retrieve a list of all configuration profiles that are registered with vCenter Server.
Export configuration profiles	You can export specific vCenter Server configuration profiles.
Validate configuration profiles	You can validate the exported vCenter Server configuration profiles. The validation process examines the configuration file for possible errors and conflicts and returns output. This operation can help avoid configuration issues or loading the wrong configuration file.
Import configuration profiles	You can import specific vCenter Server configuration profiles into another vCenter Server instance.

You can run infrastructure profile management operations by using the vSphere Automation SDK or sending an HTTP request. For information about the HTTP requests that you can use to perform the user operations, see HTTP Requests for Infrastructure Profile Management Operations.

HTTP Requests for Infrastructure Profile Management Operations

You can use HTTP requests to perform infrastructure profile management operations such as exporting, validating, and importing vCenter Server configuration profiles.

The following HTTP requests show the syntax that you can use to perform the available user operations.

- List configuration profiles

  ```
  GET https://<vcenter_ip_address_or_fqdn>:443/api/appliance/infraprofile/configs
  ```

- Export configuration profiles

  ```
  POST https://<vcenter_ip_address_or_fqdn>:443/api/appliance/infraprofile/configs?
  action=export
  ```

- Validate configuration profiles

  ```
  POST https://<vcenter_ip_address_or_fqdn>:443/api/appliance/infraprofile/configs?vmw-
  task=true&action=validate
  ```

- Import configuration profiles

  ```
  POST https://<vcenter_ip_address_or_fqdn>:443/api/appliance/infraprofile/configs?vmw-
  task=true&action=import
  ```

For information about the content and syntax of the HTTP request body for each operation, see the *API reference* documentation.

Patching and Updating vCenter Server Deployments

You can use the API to perform operations related to the planning and installing of vCenter Server software updates.

Planning vCenter Server Updates

The vCenter Server API provides operations that can help you plan the life cycle of vCenter Server instances in your environment.

You can use the operations to discover VMware products that can be associated with vCenter Server, list associated products, and manage product associations. You can also get details about available vCenter Server updates, perform pre-checks, and produce reports. The reports can contain interoperability or pre-check information. Interoperability reports contain information about the interoperability between the associated products and a specific vCenter Server version. Pre-check reports contain information about the compatibility of the current vCenter Server version with a pending update version. You can plan to perform vCenter Server updates based on the information gathered in the produced reports.

Performing Discovery and Planning Operations

You can retrieve information about VMware products associated with vCenter Server, list available vCenter Server updates, and produce reports. The findings can help you plan vCenter Server updates in your environment.

The life cycle management API provides operations that are grouped in the discovery, update, and reports categories. The discovery functionality of the API consists of the operations in the discovery category. The planning functionality of the API consists of the operations in the update category. Both functionalities can produce reports that you can download by using the reports category.

The discovery category provides operations for listing VMware products that can be associated with vCenter Server, managing products associations, and creating interoperability reports. The update category provides operations for listing all available updates and upgrades for vCenter Server and generating a pre-check compatibility report. The reports category provides an operation for downloading reports generated by interoperability and pre-check operations. By using the retrieved information, you can select one of the available patches and can plan an actual patch or upgrade for a specific vCenter Server version.

The following table lists the operations that are available in the discovery category.

Table 10-16. Discovery User Operations

Operation	Description
Get product catalog	You can retrieve a list of all VMware products that can be associated with vCenter Server.
List associated products	You can retrieve a list of all VMware product deployments in the environment that are associated with vCenter Server. **Note** The list contains both product deployments discovered automatically and deployments registered manually through the API.
Get associated product information	You can retrieve detailed information about a product associated with vCenter Server.
Create product association	You can manually associate a VMware product with vCenter Server.
Update product association	You can modify a manually added VMware product that is associated with vCenter Server. **Note** You cannot modify VMware products that are discovered automatically.
Delete product association	You can delete or dissociate a manually added VMware product that is associated with vCenter Server. **Note** You cannot delete or dissociate VMware products that are discovered automatically.
Create interoperability report	You can create an interoperability report between a vCenter Server release version and all products registered with the vCenter Server instance.

The following table lists the operations that are available in the update category.

Table 10-17. Update User Operations

Operation	Description
List updates	You can retrieve a list of all available vCenter Server updates. The list can contain minor, in-place, updates and major, migration-based, upgrades.
Get update info	You can retrieve detailed vCenter Server information about a specific update or upgrade.
Create pre-check report	You can create a vCenter Server pre-update compatibility check report for a pending update version. **Note** You can export and download the report in CSV format.

The following table lists the operations that are available in the reports category.

Table 10-18. Reports User Operations

Operation	Description
Get report	You can download the report generated by the interoperability and pre-check operations. For information about downloading the report, see the *API reference* documentation.

You can run life cycle management operations by using the vSphere Automation SDK or sending an HTTP request. For information about the HTTP requests that you can use to perform the user operations, see HTTP Requests for Discovery and Planning Operations. In addition to sending HTTP requests, you can also run cURL commands to perform operations. See cURL Examples of Performing Discovery and Planning Operations.

HTTP Requests for Discovery and Planning Operations

You can use HTTP requests to perform discovery and planning operations and download reports.

The following HTTP requests show the syntax that you can use to perform the available user operations.

Note The default port for sending HTTP requests is 443. If the vCenter Server instance is configured to use a custom port, you must replace 443 with the custom port number when sending HTTP requests.

Discovery Operations

- Get product catalog

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/product-catalog
```

- List associated products

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/associated-products
```

- Get associated product information

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/associated-products/
<product>
```

- Create product association

```
POST https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/associated-products
```

- Update product association

```
PATCH https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/associated-
products/<product>
```

- Delete product association

```
DELETE https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/associated-
products/<product>
```

- Create interoperability report

```
POST https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/interop-report
```

Update Operations

- List updates

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/update/pending
```

- Get update info

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/update/pending/<version>
```

- Create pre-check report

```
POST https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/update/pending/<version>/
precheck-report
```

Reports Operations

- Get report

```
GET https://<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/reports/<report>
```

For information about the content and syntax of the HTTP request body for each operation, see the *API reference* documentation.

cURL Examples of Performing Discovery and Planning Operations

The following cURL command examples show the syntax for performing life cycle operations such as discovering and managing product associations, listing updates, performing pre-checks, and retrieving reports.

Example: List Product Catalog

This example lists all compatible versions of VMware products that can be associated with vCenter Server.

```
curl -X GET -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/product-catalog'
```

Example: List Products Registered with vCenter Server

This example lists all VMware products associated with vCenter Server including manually added products.

```
curl -X GET -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/associated-products'
```

Example: Add a New Product

This example associates a VMware product with vCenter Server.

```
curl -H "Content-Type: application/json" -X POST -d '{"name" :"vRealize
Automation", "version": "6.8.9"}' -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/associated-products' -k -i
```

Example: Get Product Details

This example retrieves details about a product associated with vCenter Server.

```
curl -X GET -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/products/com.vmware.vRA-6.8.9'
```

Example: Modify a Product

This example updates a VMware product associated with vCenter Server. You can modify only manually added products.

```
curl -H "Content-Type: application/json" -X PATCH
-u administrator@vsphere.local:<password> 'https://<vcenter_ip_address_or_fqdn>:443/api/
vcenter/lcm/discovery/associated-products/com.vmware.vRealOrche_7.3.1' -k -d '{"spec":
{"deployments": ["<ip_address>"]}}'
```

Example: Delete a Product

This example deletes or dissociates a VMware product associated with vCenter Server. You can delete or dissociate only manually added products.

```
curl -X DELETE -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/products/com.vmware.vRA-6.8.9'
```

Example: Create an Interoperability Report

This example generates an interoperability report for all VMware products associated with vCenter Server against a specific vCenter Server version.

```
curl -X POST -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/discovery/interop-report?vmw-task=true' -k
-d '{"spec":{"target_version":"6.7.0.2000"}}' -H "Content-Type: application/json"
```

Example: List Available Updates

This example lists all available and applicable updates for vCenter Server versions discovered in your environment. The list can contain minor, in-place, updates and major, migration-based, upgrades. The operation calculates which updates or upgrades are applicable based on the current vCenter Server version.

```
curl -X GET -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/update/pending'
```

Example: Get Update Details

This example retrieves details about a specific vCenter Server update. The update is identified by the ID provided in the URL. The update can be either a minor, in-place, update or a major, migration-based, upgrade.

```
curl -X GET -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/update/pending/7.0.0.20000'
```

Example: Run Update Pre-Checks

This example performs source update pre-checks and identifies whether the provided update ID corresponds to a minor update or a major upgrade. In case of a minor update, the operation invokes the appropriate update API. In case of a major upgrade, the operation downloads and installs the `requirements.rpm` file for the upgrade. The pre-check can be a long-running task, which you can monitor by using the task ID that the operation returns.

```
curl -X POST -u
administrator@vsphere.local:<password> 'https://<vcenter_ip_address_or_fqdn>:443/api/
vcenter/lcm/update/pending/7.0.0.50000/precheck-report?vmw-task=true' -k
```

Example: Monitor the Pre-Check Task

This example tracks the status, progress, and retrieves the final result of the pre-check operation.

```
curl -X GET -u
administrator@vsphere.local:<password> 'https://<vcenter_ip_address_or_fqdn>:5480/rest/cis/
tasks/3f77223f-1edb-4992-9c48-2d75d7b3b91d:com.vmware.vcenter.lcm.update.precheck_report' -k
```

Example: Get Report

This example retrieves information about the location of the actual report for downloading.

```
curl -X GET -u administrator@vsphere.local:<password> 'https://
<vcenter_ip_address_or_fqdn>:443/api/vcenter/lcm/reports/abcd123_report.csv'
```

List Available Products and Manage Associated Products

You can automate the management of VMware products associated with vCenter Server by using the API.

This procedure includes the operations that you can use to manage the product catalog and associated products.

Prerequisites

- Verify that you have an active authenticated session with vCenter Server.

Procedure

1. Create a stub configuration.
2. Retrieve the product catalog.
3. Retrieve a list of VMware products associated with vCenter Server.
4. Associate a VMware product with vCenter Server.
5. Update a VMware product associated with vCenter Server.
6. Delete a product form the list of VMware products associated with vCenter Server.

List Available Updates

You can retrieve a list of available vCenter Server updates, details about the updates, and pre-check information by using the API.

Prerequisites

- Verify that you have an active authenticated session with vCenter Server.

Procedure

1. Create a stub configuration.
2. Retrieve a list of available vCenter Server updates.

   If there are available updates, you can retrieve details about the updates.
3. Retrieve pre-check information.

Retrieve a Report

You can retrieve a report generated by the interoperability and pre-check operations by using the API.

Prerequisites

- Verify that you have an active authenticated session with vCenter Server.

Procedure

1 Create a stub configuration.

2 Retrieve the report details.

Updating vCenter Server

vCenter Server provides interfaces to perform software updates.

Before applying updates, you must make sure that your environment is prepared for the vCenter Server software update process.

Applying vCenter Server Software Updates

You can automate the installation of vCenter Server software updates to ensure that your system is stable and protected. Software updates can include security fixes, performance optimizations, and new features.

Security patches usually address vulnerabilities in third-party components and do not affect the vCenter Server functionality. vCenter Server bug fixes can introduce changes to the functionality without affecting the data format or database schema of the system.

Each update contains metadata with information about the updated content, for example, whether high-priority OS updates are included. The update metadata includes a list of components to be updated, the release date of the update, a list of fixed issues, time and disk space requirements, and information whether a reboot is required. The metadata can also contain a new vCenter Server version number, including a build number. In addition to the metadata, an update can contain optional components such as update scripts, new versions of vCenter Server software components, and new versions of OS components.

vCenter Server can obtain software updates from either a URL or an ISO image. The URL can either point to the VMware Web repository or to a custom repository in which you upload the updates in ZIP format. To perform an update by using an ISO image, attach the image to the CD/DVD drive of the vCenter Server instance.

There are multiple phases of the update process. For details, see vCenter Server Software Update Workflow.

If you want to prevent issues related to the possibility of update installation failures, you should create a backup or take a snapshot of your vCenter Server instance before you start the update process. A backup can also be useful when an update is successfully installed. For example, you might decide to revert to the previous version if you encounter any undesired system behavior related to functional changes in the new software version.

Table 10-19. User Operations

Operation	Description
Get state information	You can retrieve information about the update state.
Check for update	You can check whether a new update is available.
Get update information	You can retrieve information about the available updates.
Get update requirements	You can retrieve information about the update requirements.
Stage	You can initiate the download of the update. **Note** The check phase must have completed successfully before you can stage the update.
Get staging status	You can retrieve information about the status of the stage operation. **Note** You must provide the task ID value that you received as a response when you initiated the stage operation.
Install	You can initiate the installation of the update. **Note** The update must be staged before you can install it.
Get installation status	You can retrieve information about the status of the install operation. **Note** You must provide the task ID value that you received as a response when you initiated the install operation.
Stage and install	You can initiate the download of the update and the installation starts when the download completes.

You can run software update operations by using the vSphere Automation SDK or sending an HTTP request.

Note When you send the requests, you must use an authentication.

If you send requests to port `5480`, you must authenticate with vCenter Server **root** credentials. If you send requests to the vCenter Server reverse proxy port, you must authenticate with vCenter Single Sign-On credentials.

The following HTTP requests show the syntax that you can use to perform the available user operations.

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending?source_type=DEFAULT
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending/<target_version>
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending/<target_version>/
requirements
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending/<target_version>?
action=stage
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/task/<update_stage_task_id>
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending/<target_version>?
action=install
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/task/<update_install_task_id>
```

```
https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending/<target_version>?
action=stage-and-install
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/update
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/update/pending?source_type=DEFAULT
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/update/pending/<target_version>
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/update/pending/<target_version>/
requirements
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/update/pending/<target_version>?
action=stage
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/task/<update_stage_task_id>
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/update/pending/<target_version>?
action=install
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/task/<update_install_task_id>
```

```
https://<vcenter_ip_address_or_fqdn>/api/appliance/update/pending/<target_version>?
action=stage-and-install
```

In addition to sending HTTP requests, you can also run cURL commands to perform update operations. See cURL Examples of Performing vCenter Server Software Update Operations.

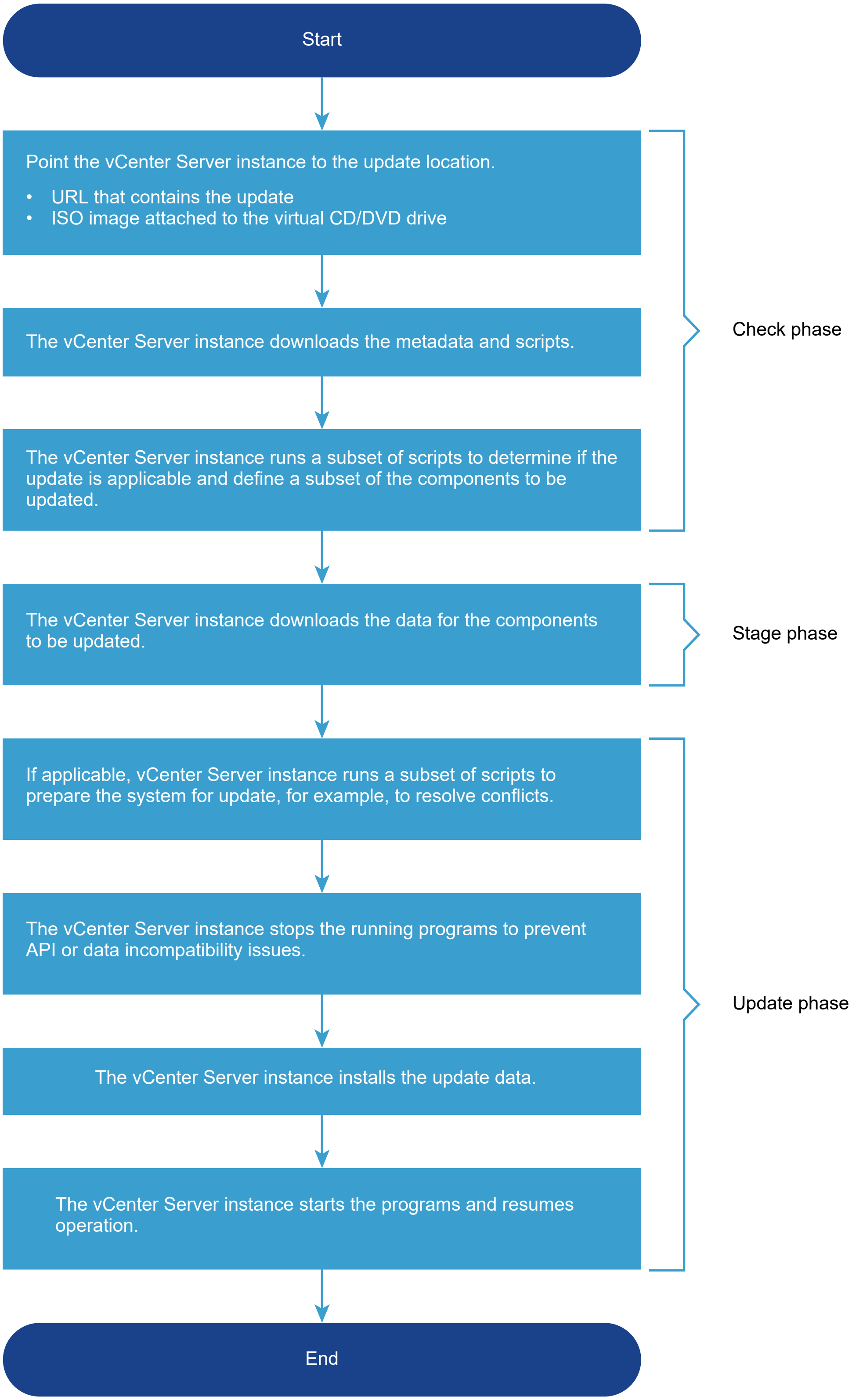
vCenter Server Software Update Workflow

The vCenter Server software update process consists of three major phases. In the first phase, the vCenter Server instance performs various checks, in the second phase it stages the update, and applies the update in the final phase.

To initiate the update process, you must choose whether the vCenter Server instance should obtain software updates from a URL or an ISO image. If you use an ISO image to update the vCenter Server instance, the image must remain attached to the CD/DVD drive of the instance during the stage and install operations.

The workflow in Figure 10-9. Update Process Workflow describes the standard steps of the update process.

Figure 10-9. Update Process Workflow


Start
Point the vCenter Server instance to the update location.
• URL that contains the update
• ISO image attached to the virtual CD/DVD drive
The vCenter Server instance downloads the metadata and scripts.
The vCenter Server instance runs a subset of scripts to determine if the update is applicable and define a subset of the components to be updated.
Check phase
The vCenter Server instance downloads the data for the components to be updated.
Stage phase
If applicable, vCenter Server instance runs a subset of scripts to prepare the system for update, for example, to resolve conflicts.
The vCenter Server instance stops the running programs to prevent API or data incompatibility issues.
The vCenter Server instance installs the update data.
The vCenter Server instance starts the programs and resumes operation.
Update phase
End

You can automate checks for new updates and staging of updates by using an update policy. For example, you can set an update policy to make the vCenter Server instance perform automatic checks for new updates at midnight every day. If there are new updates available, the vCenter Server instance can stage them automatically. Using an update policy reduces the waiting time by automating the first two phases and giving you the option to initiate only the update phase manually.

cURL Examples of Performing vCenter Server Software Update Operations

The following cURL command examples show the syntax for performing update operations such as checking for, staging, and installing updates, as well as retrieving information about update status, and setting update policies.

Example: Check for an Update

This example queries a custom URL for a new update.

```
curl -X GET -k -u root:<root_password> "https://<vcenter_ip_address_or_fqdn>:5480/rest/
appliance/update/pending?source_type=LOCAL_AND_ONLINE&url=https://<custom_url>"
```

Example: Stage an Update

This example initiates the staging of the update.

```
curl -X POST -k -u root:<root_password> -H  "Content-Type: application/json"
-d '{"version":"<target_version>"}' https://<vcenter_ip_address_or_fqdn>:5480/rest/appliance/
update/pending/<target_version>?action=stage
```

Example: Install an Update

This example initiates the installation of the update.

```
curl -X POST -k -u root:<root_password> -H "Content-Type: application/json" -d
'{"version":"<target_version>","user_data":
[{"key":"vmdir.password","value":"<sso_password>"}]}' https://
<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending/<target_version>?
action=install
```

Example: Stage and Install an Update

This example downloads the update and installs it when the download completes.

```
curl -X POST -k -u root:<root_password> -H
"Content-Type: application/json" -d '{"version":"<target_version>","user_data":
[{"key":"vmdir.password","value":"<sso_password>"}]}' https://
<vcenter_ip_address_or_fqdn>:5480/rest/appliance/update/pending/<target_version>?action=stage-
and-install
```

Example: Retrieve Update Status

This example retrieves information about the update state.

```
curl -X GET -k -u root:<root_password> https://<vcenter_ip_address_or_fqdn>:5480/rest/
appliance/update
```

Example: Set an Update Policy

This example sets an update policy to check a custom URL for new updates at specific times every Friday and Saturday, and if a new update is available, it is staged automatically.

```
curl -X PUT -k -u root:<root_password> -H "Content-Type: application/json" -d '{"policy":
{"auto_stage": true,"check_schedule": [{"day": "FRIDAY","hour": 23,"minute": 30},{"day":
"SATURDAY","hour": 12,"minute": 30}],"custom_URL":"https://123.com"}}'
```