API Programming Guide

25 AUG 2022
vRealize Operations 8.6

vmware®

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

https://docs.vmware.com/

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

Contents

About This Programming Guide

1

The vRealize Operations API Programming Guide provides information about the vRealize Operations REST APIs, including how to use the REST API resources, authenticate, and construct REST API calls.

Intended Audience

This information is intended for administrators and programmers who want to configure and manage vRealize Operations programmatically using the vRealize Operations REST API. The guide focuses on common use cases.

Understanding the vRealize Operations API

2

Developers can use the API to build interactive clients of vRealize Operations. The API follows the REST style and is available to all licensed users.

vRealize Operations clients communicate with the server over HTTP, exchanging representations of vRealize Operations objects. These representations take the form of JSON or XML elements. You use HTTP GET requests to retrieve the current representation of an object, HTTP POST and PUT requests to create or modify an object, and HTTP DELETE requests to delete an object.

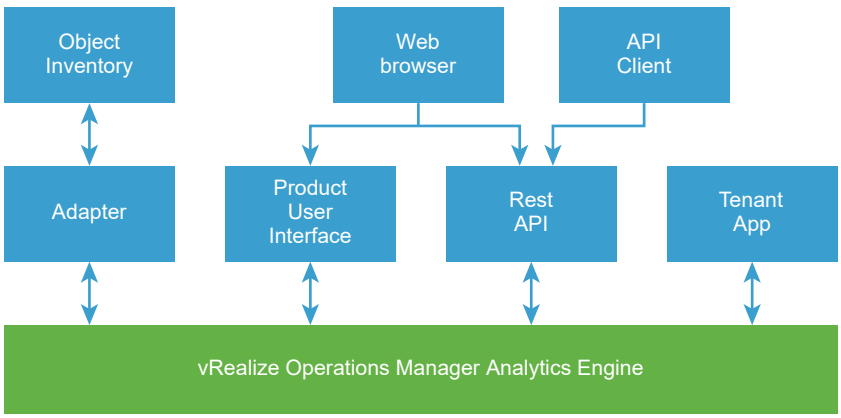
This chapter includes the following topics:

- How the vRealize Operations API Works
- Client Workflow Overview
- Using the API with vRealize Operations

How the vRealize Operations API Works

You use a Web browser to communicate with the vRealize Operations analytics engine, either through the product user interface or through API calls.

Figure 2-1. vRealize Operations Simplified Architecture



The adapter instance collects data from objects in your monitored environment. The vRealize Operations analytics engine processes the data and displays the complete model in the graphical interface.

Why Use the API

The API is most useful when there is a need to automate a well-defined workflow, such as repeating the same tasks to configure access control for new vRealize Operations users. The API is also useful when performing queries on the vRealize Operations data repository, such as retrieving data for particular assets in your virtual environment. In addition, you can use the API to extract all data from the vRealize Operations data repository and load it into a separate analytics system.

vRealize Operations Terminology

The XML syntax you use to describe the objects for an adapter corresponds to the API code syntax but differs from what you find in the user interface. The following terms appear in the user interface. Included with the description of each term is the corresponding XML syntax used in an API call.

Adapter types	Defines the adapter used to discover particular object types. For example, the vCenter adapter discovers objects connected to vSphere datacenters. The AWS adapter discovers AWS services and objects. XML syntax: `adapterkinds`.
Object types	The class of entities that represent objects or information sources. Objects report data to the vRealize Operations analytics engine. Virtual machines, datastores, and host systems are examples of object types defined in a vCenter adapter model. XML syntax: `resourcekinds`.

Client Workflow Overview

vRealize Operations API clients implement a REST workflow, making HTTP requests to the server and retrieving the information they need from the server's responses.

About REST

REST, an acronym for Representational State Transfer, describes an architectural style characteristic of programs that use the Hypertext Transfer Protocol (HTTP) to exchange serialized representations of objects between a client and a server. In the vRealize Operations API, these representations are JSON or XML documents.

In a REST workflow, representations of objects are passed back and forth between a client and a server with the explicit assumption that neither party need know anything about an object other than what is presented in a single request or response. The URLs at which these documents are available often persist beyond the lifetime of the request or response that includes them.

REST API Workflows

Application programs written to use a REST API use HTTP requests that are often executed by a script or other higher-level language to make remote procedure calls that create, retrieve, update, or delete objects that the API defines. In the vRealize Operations REST API, these objects are defined by a collection of XML schemas. The operations themselves are HTTP requests, and so are generic to all HTTP clients.

To write a REST API client application, you must understand only the HTTP protocol, and the semantics of JSON or XML, the transfer format that the vRealize Operations API uses. To use the API effectively in such a client, you must become familiar with the following concepts.

- The set of objects that the API supports, and what they represent.
- How the API represents these objects.
- How a client refers to an object on which it wants to operate.

The API reference includes a complete list of API requests. See About the Schema Reference.

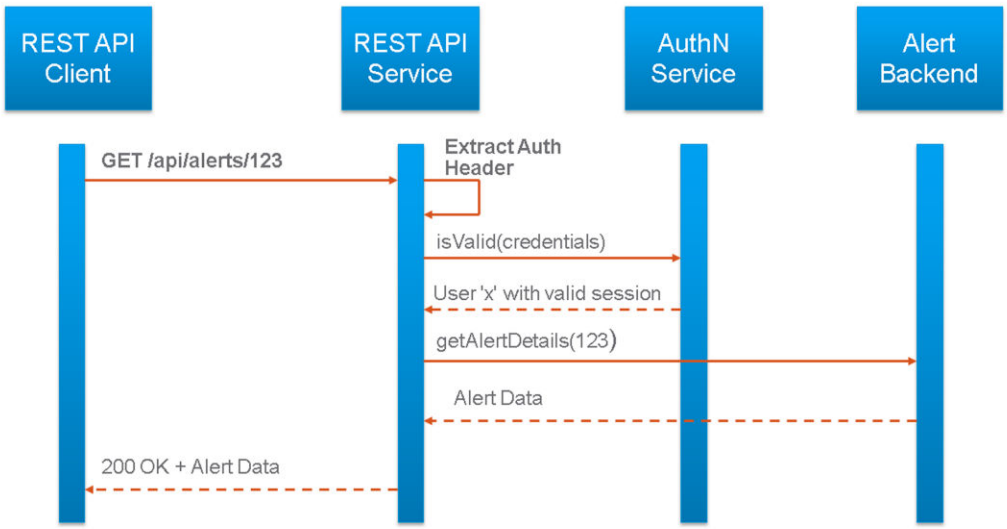
vRealize Operations API REST Requests

To retrieve object representations, clients make HTTP requests to object references.

Security

The HTTP link between an API client and server is established using SSL. API clients configure token-based authentication to communicate with the server.

Figure 2-2. Scenario: Provide user credentials to obtain details about alert with ID 123



With token-based authentication, you POST a login request to the vRealize Operations API server, supplying valid user credentials to obtain an authentication token. The following example presents a token-based authentication scenario.

1 You obtain valid user credentials for your vRealize Operations instance.

2 POST a request to the REST endpoint for authentication.

```
https://RESTendpoint.example.com/suite-api/api/auth/token/acquire
```

The request body includes the user name, password, and authentication source.

3 In the response body, the endpoint returns the token, expiry date, and time.

4 For further communication, you include the token object in the Authorization header with the format :

```
Authorization: vRealizeOpsToken <vROps_token>
```

Alternatively, if you acquired the token from an SSO source, the Authorization header is of the format:

```
Authorization: SSO2Token <SSO_SAML_TOKEN>
```

5 You can invalidate the token before the expiration date and time by sending a POST request to the logout endpoint.

```
POST https://RESTendpoint.example.com/suite-api/api/auth/token/release
```

Request Headers

The following HTTP headers are typically included in API requests:

Accept-Language	To specify the language desired in responses, use the `Accept-Language` request header. Message strings in `ErrorType` responses are localized. To request a response with message strings localized to French, use the following header: `Accept-Language: fr-FR`
Authorization	All requests to create an API session must include an `Authorization` header of the form prescribed by the identity provider that your organization uses
Content-Type	Requests that include a body must include an appropriate HTTP `Content-Type` header. ■ For a request body in XML, the header must include `Content-Type: application/xml` ■ For a request body in JSON, the header must include `Content-Type: application/json`
Accept	To specify the desired response format , include the `Accept` request header. ■ For a response in XML, the header must include `Accept: application/xml` ■ For a response in JSON, the header must include `Accept: application/json`

Request Bodies in XML

For a request body written in XML, vRealize Operations uses a validating XML parser that requires elements in a request body to agree with the schema in order and number. Request bodies are rejected as invalid unless they meet the following criteria:

- XML namespace attributes must be supplied for all namespaces represented by elements in the request.
- If multiple namespaces are represented in the request, XML namespace attributes must include an identifying prefix, and that prefix must be used with all elements from that namespace.
- All required elements must appear in request bodies. All elements that appear in request bodies must appear in the order that the schema establishes, and with content that conforms to the type constraint that the schema specifies.

vRealize Operations API REST Responses

All responses include an HTTP status code and, unless the status code is 204 (No Content), an Accept header. Response content depends on the request. Some responses include a document body, some include only a URL, and some are empty.

HTTP Response Codes

An API client can expect a subset of HTTP status codes in a response.

Table 2-1. HTTP Status Codes that the API Returns

Status Code	Status 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 can be found at the URL specified in the Location header.
`202 Accepted`	The request is valid and a task was created to handle it. This response is usually accompanied by a `TaskStatus` element .
`204 No Content`	The request is valid and was completed. The response does not include a body.
`400 Bad Request`	The request body is malformed, incomplete, or otherwise invalid.
`401 Unauthorized`	Login failed or authentication token has expired.
`403 Forbidden`	The user is not authenticated or does not have adequate privileges to access one or more objects specified in the request.
`404 Not Found`	The object specified in the request could not be found.
`405 Method Not Allowed`	The HTTP method specified in the request is not supported for this object.
`406 Not Acceptable`	The resource identified by the request is not capable of generating a response of the type specified in the request's `Accept` header.
`415 Unsupported Media Type`	The resource identified by the request does not support a request of the specified `Content-Type` and HTTP method.
`422 Not Found`	Usually indicates a malformed request URL or request body.
`429 Too Many Requests`	A client has sent too many requests or multiple clients are sending too many simultaneous requests and the server is unable to process them due to rate limits. To work around the problem, try sending the request again later.
`500 Internal Server Error`	The request was received but could not be completed because of an internal error on the server.

Table 2-1. HTTP Status Codes that the API Returns (continued)

Status Code	Status Description
`503 Service Unavailable`	The server is currently unable to handle the request due to a temporary condition such as resource exhaustion or server maintenance.
`504 Gateway Timeout`	The server, while acting as a gateway or proxy, did not receive a timely response from the upstream server specified by the request URL.

Using the API with vRealize Operations

You can use a browser or an HTTP client program to send requests and receive responses.

REST Client Programs

Any client application that can send HTTPS requests is an appropriate tool for developing REST applications with the vRealize Operations API. REST client plug-ins are available for most browsers and many IDEs. The following open-source programs are commonly used:

- cURL. http://curl.haxx.se
- Postman application. http://www.getpostman.com

In addition, VMware provides language-specific client bindings for the vRealize Operations API. See About the Schema Reference.

About the Schema Reference

The vRealize Operations REST API documentation includes reference material for all elements, types, queries, and operations in the vRealize Operations API. It also includes the schema definition files.

Swagger based API documentation is available with the product, with the capability of making REST API calls right from the landing page.

To access the API documentation, use the URL of your vRealize Operations instance. For example, if the URL of your instance is `https://vrealize.example.com`, the API reference is available from: `https://vrealize.example.com/suite-api/doc/swagger-ui.html`.

Language-specific client bindings are available from:

```
https://vrealize.example.com/suite-api/
```

About the vRealize Operations API Examples

All examples include HTTP requests and responses. These examples show the workflow and content associated with operations such as creating and querying for information about objects in your monitored environment.

Example request bodies are in JSON. Request headers required by the vRealize Operations API are included in example requests that are not fragments of a larger example.

Most example responses show only those elements and attributes that are relevant to the operation being discussed. Ellipses (…) indicate omitted content within response bodies.

Getting Started with the API

3

API clients and vRealize Operations servers communicate over HTTPS, exchanging XML representations of API objects.

This simple example of a REST API workflow shows how to obtain a list of all metrics for a virtual machine object type that is included in the model definition of the VMware vCenter® adapter. Using the API, you can obtain the complete list of available metrics for any object type.

This chapter includes the following topics:

Acquire an Authentication Token

vRealize Operations Manager requires API requests to be authenticated. The first step in this workflow is to obtain an authentication token.

To obtain an authentication token, the login request supplies the user credentials in a form that Basic HTTP authentication requires. In this example, the user is logging in to a vRealize Operations Manager instance with URL `https://vrealize.example.com/`.

Note This example uses token-based authentication. For more information regarding authentication mechanisms, see Security.

Using `authSource`, you can import and authenticate users and user group information that reside on another machine. For example, you can authenticate users from LDAP, Active Directory, VMware Identity Manager, Single Sign-On and so on. When you import user account information that resides on another machine, you must define the criteria used to import the user accounts from the source machine.

After creating an auth source you can use it for acquiring a token by specifying the name. The default auth source type is LOCAL.

Prerequisites

- Secure a channel between the web browser and the vRealize Operations Manager server. Open a browser and enter the URL of a vRealize Operations Manager instance such as:

  ```
  https://vrealize.example.com
  ```

  The system warns that your connection is not private. Click through to confirm the security exception and establish an SSL handshake.
- Verify that you can access the APIs. Enter the URL of your vRealize Operations Manager instance with `suite-api/docs/rest/index.html` added to the end, such as:

  ```
  https://vrealize.example.com/suite-api/docs/rest/index.html
  ```
- Verify that you have the login credentials for a user of your vRealize Operations Manager instance.

Procedure

1 POST a request to the login URL to acquire a token.

  ```
  POST https://vrealize.example.com/suite-api/api/auth/token/acquire
  ```

  See Login Request and Response.

2 Examine the response.

  A successful request returns an `ops` authorization token, which you must included in subsequent API requests.

Example: Login Request and Response

This example shows a request and response for a user with the login username: **`vRealize-user`** and password: **`vRealize-dummy-password`**.

Request header:

```
POST https://vrealize.example.com/suite-api/api/auth/token/acquire
Content-Type: application/json
Accept: application/json
```

Request body in JSON format:

```
{
  "username" : "vRealize-user",
  "password" : "vRealize-dummy-password"
}
```

Response in JSON:

```
200 OK
```

```
{
  "token": "8f868cca-27cc-43d6-a838-c5467e73ec45::77cea9b2-1e87-490e-b626-e878beeaa23b",
  "validity": 1470421325035,
  "expiresAt": "Friday, August 5, 2016 6:22:05 PM UTC",
  "roles": []
}
```

The response code indicates whether the request succeeded, or how it failed.

- If the request is successful, the server return HTTP response code 200 (OK) and re-usable `ops` authorization token that expires after six hours. This token must be included in each subsequent API request.
- If the authorization header is missing for the request, the server returns HTTP response code 403.
- If the credentials supplied in the Authorization header are invalid, the server returns HTTP response code 401.

Find the Adapter Type and Object Type

Your vRealize Operations instance includes multiple adapter types. To find the adapter type for the vCenter adapter, you make a GET request to retrieve a list of all adapter types. The API response includes all the object types that the adapter monitors.

Prerequisites

Verify that you are logged in to the vRealize Operations instance.

Procedure

1 Make a GET request for all adapter types.

```
GET https://vrealize.example.com/suite-api/api/adapterkinds
```

2 Examine the response to find the vCenter adapter and list of monitored object types.

See the response portion of Determine the Adapter Type and Object Types for the vCenter Adapter.

Example: Determine the Adapter Type and Object Types for the vCenter Adapter

This example finds the adapter type for the vCenter adapter and all the object types included in the adapter model definition.

Request header:

```
GET https://vRealize.example.com/suite-api/api/adapterkinds
Content-Type: application/json
Authorization: vRealizeOpsToken <vROps_token>
Accept: application/json
```

Where *vROps_token* is the token that you obtained from the response in Acquire an Authentication Token.

Snippet of the response in JSON for the vCenter Adapter:

```
200 OK
```

```
{
  "key": "VMWARE",
  "name": "vCenter Adapter",
  "description": "Provides the connection information and credentials required...",
  "adapterKindType": "GENERAL",
  "describeVersion": 573,
  "identifiers": [],
  "resourceKinds": [
    "ClusterComputeResource",
    "ComputeResource",
    "CustomDatacenter",
    "Datacenter",
    "Datastore",
    "StoragePod",
    "DatastoreFolder",
    "VM Entity Status",
    "Folder",
    "HostFolder",
    "HostSystem",
    "NetworkFolder",
    "ResourcePool",
    "VMwareAdapter Instance",
    "VirtualMachine",
    "VMFolder",
    "DistributedVirtualPortgroup",
    "VmwareDistributedVirtualSwitch",
    "vSphere World"
  ],
  ...
}
```

For the vCenter adapter, the `adapter-kind` key is `VMWARE`. The `resourceKinds` are the object types that the vCenter adapter monitors. For virtual machine object type, the `resourceKinds` is `VirtualMachine`.

Generate a List of All Metrics for the Object

To generate a complete list of metrics for any virtual machine defined in the vCenter adapter model, you make a GET request to the URL with the adapter type and the object type.

Prerequisites

Verify that the following requirements are met:

- You are logged in to the vRealize Operations instance.
- You know the `adapterKind` value for the vCenter adapter and the `resourceKinds` value for the virtual machine. See Determine the Adapter Type and Object Types for the vCenter Adapter

Procedure

1 Make a GET request to obtain the metadata for metrics.

```
GET https://vrealize.example.com/suite-api/api/adapterkinds/VMWARE/resourcekinds/
VirtualMachine/statkeys
```

2 Compare the metrics listed in the response to metrics displayed in the user interface. See Virtual Machine Metrics from the API and in the User Interface

Example: Virtual Machine Metrics from the API and in the User Interface

This example shows how the virtual machine metrics listed in the XML response compare to the metrics displayed in the vRealize Operations user interface.

Request:

```
GET https://vrealize.example.com/suite-api/api/adapterkinds/VMWARE/resourcekinds/
VirtualMachine/statkeys
Content-Type: application/json
Authorization: vRealizeOpsToken <vROps_token>
Accept: application/json
```

Where:

- `VMWARE` is the `adapterKindKey`.
- `VirtualMachine` is the `resourceKindKey`.
- *vROps_ token* is the token that you obtained from the response in Acquire an Authentication Token

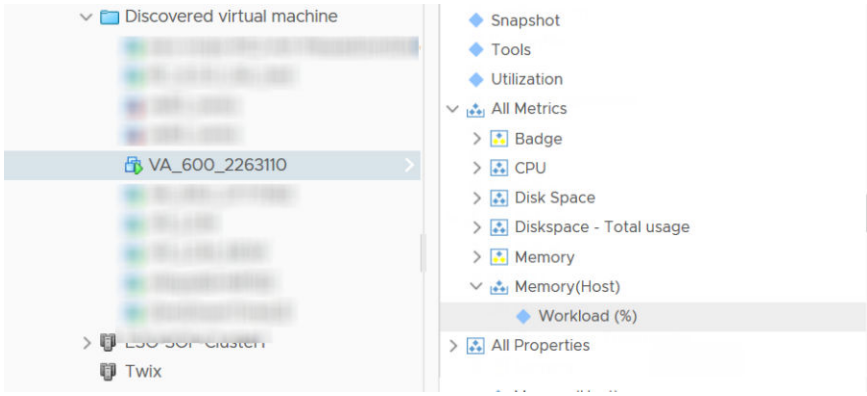
Snippet of the response in JSON:

```
200 OK
```

```
{
    "resourceTypeAttributes": [
        ...
        {
            "key": "mem|host_workload",
            "name": "Memory|Host Workload",
            "description": "Host Workload (%)",
            "defaultMonitored": false,
            "rollupType": "AVG",
            "instanceType": "INSTANCED",
            "unit": "%",
            "dataType2": "FLOAT",
            "monitoring": false,
            "property": false
        },
        ...
    ]
}
```

Every `resourceTypeAttribute` in the response is a metric with metadata for a virtual machine object. The `name` corresponds to text displayed in the vRealize Operations user interface. In this example, the snippet lists metrics for Memory and Host Workload.

To compare metrics in the response with metrics in the user interface, log in to the vRealize Operations instance running on `vrealize.example.com` and navigate to the metrics for a virtual machine. The following example shows where you find metrics for Memory(Host) and Workload.



The example shows how to retrieve metrics for the virtual machine object type. To retrieve metrics for other object types, replace `VirtualMachine` in the GET request with other `resourceKinds`.

Configuring an Adapter Instance 4

After installing a solution that includes a management pack with an adapter, you must configure an adapter instance to collect data from the objects in the adapter model definition. You can use the vRealize Operations API to configure an adapter instance.

This use case example shows how to configure an adapter instance for a vSphere Solution and includes:

- summary of operations with request, request body, and response for each
- specific procedure for each operation

This chapter includes the following topics:

Summary of Configuring an Adapter Instance Requests

You make sequential API requests to configure an adapter instance. Responses from earlier requests yield information required for a subsequent requests.

Table 4-1. Summary of Requests

Operation	Request	Request Body	Response
Get all solutions registered with the product and identify the adapter types	GET <*API-URL*>/suite-api/api/solutions	None	`adapterkindkeys`
Get all the object types for a particular adapter type.	GET <*API-URL*>/suite-api/api/adapterkinds/{key}/resourcekinds	None	`resourceIdentifierTypes`
Create an adapter instance object	POST <*API-URL*>/suite-api/api/adapters	Values for `resourceIdentifiers` and `credential`	`adapterid`

Table 4-1. Summary of Requests (continued)

Operation	Request	Request Body	Response
Patch an adapter instance to acknowledge the presented certificate	PATCH <*API-URL*>/suite-api/api/adapters	API response of POST <*API-URL*>/suite-api/api/adapters	API response of POST <*API-URL*>/suite-api/api/adapters without `adapter-cerfiticates`
Start adapter monitoring	PUT <*API-URL*>/suite-api/api/adapters/{adapterid}/monitoringstate/start	None	200 OK

Identify the Solution and Its Adapters

Your vRealize Operations instance may have several solutions installed. To find the vSphere solution and its adapter types, you make a GET request to retrieve a list of all solutions. The response includes all the adapters included with the solution.

For this example use case, the *API-URL* for the vRealize Operations instance is `vrealize.example.com`.

Prerequisites

Verify that you can log in to the API URL for a vRealize Operations instance. See Acquire an Authentication Token.

Procedure

1 Make a GET request to list all the solutions.

```
GET https://vrealize.example.com/suite-api/api/solutions
```

2 Examine the response to find the vSphere solution and its adapter types.

See the response portion of Adapter Types for the vSphere Solution.

Example: Adapter Types for the vSphere Solution

This example lists all the installed solutions and the adapter types for each.

Request header:

```
GET https://vrealize.example.com/suite-api/api/solutions
```

The response in JSON:

```
200 OK
```


```
{
  "solution":
    {
      "id": "MPforLogInsight",
      "name": "VMware vRealize Operations Management Pack for Log Insight",
      "version": "6.0.3171089",
      "description": "VMware vRealize Operations Management Pack for Log Insight... ",
      "vendor": "VMware Inc.",
      "adapterKindKeys": [
        "LogInsightAdapter"
      ]
    },
    {
      "id": "ep-ops-os-and-availability",
      "name": "Operating Systems / Remote Service Monitoring",
      "version": "1.0.4071095",
      "description": "The End Point Operations Management Solution for Operating... ",
      "vendor": "VMware Inc.",
      "adapterKindKeys": [
        "ep-ops-os-and-availability-kind"
      ]
    },
    {
      "id": "VMware vSphere",
      "name": "VMware vSphere",
      "version": "6.0.7496664",
      "description": "Manages vSphere objects such as Clusters, Hosts...",
      "vendor": "VMware Inc.",
      "adapterKindKeys": [
        "VMWARE",
        "PythonRemediationVcenterAdapter"
      ]
    }
  ]
}
```


The response shows three solutions installed:

- Management Pack for Log Insight solution
- End Point Operations solution
- vSphere solution

The vSphere solution has two adapter types:

- VMWARE
- PythonRemediationVcenterAdapter

For the vCenter adapter, the adapter type is VMWARE.

Identify the Object Types Required for the Adapter

After you determine that you want to create an instance of the vCenter adapter, you must identify the required object types for that adapter. You make a GET request to retrieve a list of all object types for the vCenter adapter.

Prerequisites

Verify that you know the adapter type for the vCenter adapter.

Procedure

1 Make a GET request to list all the object types for the vCenter adapter.

```
GET https://vrealize.example.com/suite-api/api/adapterkinds/VMWARE/resourcekinds
```

2 Examine the response to identify the required object types..

See the response portion of Object Types Required for the vCenter Adapter.

Example: Object Types Required for the vCenter Adapter

This example finds all the object types for the vCenter adapter.

Request header:

```
GET https://vrealize.example.com/suite-api/api/adapterkinds/VMWARE/resourcekinds
```

Snippet of the response in JSON:

```
200 OK
```


```
{
  "key": "VMwareAdapter Instance",
  "name": "vCenter Server",
  "adapterKind": "VMWARE",
  "resourceKindType": "ADAPTER_INSTANCE",
  "resourceKindSubType": "NONE",
  "resourceIdentifierTypes": [
    {
      "name": "AUTODISCOVERY",
      "dataType": "STRING",
      "isPartOfUniqueness": false
    },
    {
      "name": "DISABLE_COMPUTATION_BASED_ON_CONSUMERS",
      "dataType": "STRING",
      "isPartOfUniqueness": false
    },
    {
      "name": "DV_PORT_GROUP_DISABLED",
      "dataType": "STRING",
      "isPartOfUniqueness": false
    },
```

```
    {
      "name": "DVS_DISABLED",
      "dataType": "STRING",
      "isPartOfUniqueness": false
    },
    {
      "name": "PROCESSCHANGEEVENTS",
      "dataType": "STRING",
      "isPartOfUniqueness": false
    },
    {
      "name": "VCURL",
      "dataType": "STRING",
      "isPartOfUniqueness": true
    },
    ...
    {
      "name": "VM_LIMIT",
      "dataType": "INTEGER",
      "isPartOfUniqueness": false
    }
    ],
  ...
}
```

This snippet shows the Resource Kind with the attribute `"resourceKindType": "ADAPTER_INSTANCE"`. Any object type that has the resource identifier `"isPartOfUniqueness":true` requires a value for that object type with the API request to create the adapter instance.

An adapter instance of the vCenter adapter requires a value for `VCURL` or the URL of the vCenter.

Create the Adapter Instance

After you identify the object types required for the adapter, you provide parameter values for the object types to create an adapter instance. Your POST request includes a request body with the required parameters.

To create an adapter instance, the VCURL setting is mandatory.

Prerequisites

Verify that you have an IP address and credentials for a vCenter.

Procedure

1 Make a POST request to create the adapter instance.

```
POST https://vrealize.example.com/suite-api/api/adapters
```

2 Examine the response to find the name for the vSphere Solution and its adapter types.

See the response portion of Adapter Instance.

Example: Adapter Instance

This example creates the adapter instance for a vCenter with the following parameters:

- Display Name: VC Adapter Instance
- Description: A vCenter Adapter Instance for VC 12.345.678.9
- vCenter Server IP address: https://12.345.678.9
- Credential name: VC-Credential-1
- User Name: administrator@vsphere.local
- Password: VC-dummy-passwd

`AUTODISCOVERY` and `PROCESSCHANGEEVENTS` are optional, but are included to show additional examples of resource identifiers in the request body and in the response.

Request header:

```
POST https://vrealize.example.com/suite-api/api/adapters
```

Request body in JSON format:

```
{
  "name" : "VC Adapter Instance",
  "description" : "A vCenter Adapter Instance for VC 12.345.678.9",
  "collectorId" : "1",
  "adapterKindKey" : "VMWARE",
  "resourceIdentifiers" : [
    {
      "name" : "AUTODISCOVERY",
      "value" : "true"
    },
    {
      "name" : "PROCESSCHANGEEVENTS",
      "value" : "true"
    },
    {
      "name" : "VCURL",
      "value" : "https://12.345.678.9"
    }
  ],
  "credential" : {
    "id" : null,
    "name" : "VC-Credential-1",
    "adapterKindKey" : "VMWARE",
    "credentialKindKey" : "PRINCIPALCREDENTIAL",
    "fields" : [
      {
        "name" : "USER",
        "value" : "administrator@vsphere.local"
      },
      {
        "name" : "PASSWORD",
```

```
        "value" : "VC-dummy-passwd"
      }
    ],
  },
}
```

Snippet of the response in JSON:

```
201 Created
```

```
{
  "resourceKey": {
    "name": "VC Adapter Instance",
    "adapterKindKey": "VMWARE",
    "resourceKindKey": "VMwareAdapter Instance",
    "resourceIdentifiers": [
      {
        "identifierType": {
          "name": "AUTODISCOVERY",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": "true"
      },
      {
        "identifierType": {
          "name": "DISABLE_COMPUTATION_BASED_ON_CONSUMERS",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
          "name": "DV_PORT_GROUP_DISABLED",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
        "name": "DVS_DISABLED",
        "dataType": "STRING",
        "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
          "name": "PROCESSCHANGEEVENTS",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
```

```
        "value": "true"
      },
      {
        "identifierType": {
          "name": "VCURL",
          "dataType": "STRING",
          "isPartOfUniqueness": true
        },
        "value": "https://12.345.678.9"
      },
      {
        "identifierType": {
          "name": "VM_FOLDER_DISABLED",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
          "name": "VM_LIMIT",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      }
    ]
  },
  "description": "A vCenter Adapter Instance for VC 12.345.678.9",
  "collectorId": 1,
  "collectorGroupId": "909c2fbf-2c2c-4957-9a75-21bf2a887d31",
  "credentialInstanceId": "65081a8d-d462-43b2-b4e0-596eaf3d497e",
  "monitoringInterval": 5,
  "adapter-certificates": [
    {
      "thumbprint": "2520fb4351bc91ee7b82ef7cc54a8d88fa893da9",
      "certificateDetails": "[
        Version: V3 Subject: C=US, CN=12.345.678.9
        Signature Algorithm: SHA256withRSA, OID = 1.2.840.113549.1.1.11
        Key: Sun RSA public key, 2048 bits modulus: ...
        Validity: [From: Wed Jul 15 19:26:51 UTC 2015, To: Tue Jul 08 11:26:30 UTC 2025]
        Issuer: O=W12R2UINanduVC, C=US, DC=local, DC=vsphere, CN=CA ...
          ...
      ]"
    }
  ],
  ...
  "id": "a97bd204-e3e5-404b-a219-e2b20cf158d2"
}
```


The API creates a new adapter with an internally generated UUID that uniquely identifies the object. The API response includes the certificates that vCenter 12.345.678.9 presents. The value of the adapter instance ID is used to start monitoring and collecting data.

Provide Proof of Certificate Validity

Before vRealize Operations can connect to the vCenter and start collecting data, it needs to verify that data sources discovered by the adapter instance are presenting valid certificates. Your PATCH request provides the proof that the certificates are valid by including a request body that is the response from the POST request used to create the adapter.

Prerequisites

Verify that you have the response from the POST request used to create the adapter. See the response in Adapter Instance.

Procedure

- Make a PATCH request to notify the system that the user has accepted the certificate presented by the vCenter.

```
PATCH https://vrealize.example.com/suite-api/api/adapters
```

Example: Certificate Validation

In this example, the request body for the PATCH request is the same as the response from the POST request used to create the adapter instance.

Request header:

```
PATCH https://vrealize.example.com/suite-api/api/adapters
```

Request body in JSON format:

```
{
  "resourceKey": {
    "name": "VC Adapter Instance",
    "adapterKindKey": "VMWARE",
    "resourceKindKey": "VMwareAdapter Instance",
    "resourceIdentifiers": [
      {
        "identifierType": {
          "name": "AUTODISCOVERY",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": "true"
      },
      {
        "identifierType": {
          "name": "DISABLE_COMPUTATION_BASED_ON_CONSUMERS",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
```

```
        "identifierType": {
          "name": "DV_PORT_GROUP_DISABLED",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
        "name": "DVS_DISABLED",
        "dataType": "STRING",
        "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
          "name": "PROCESSCHANGEEVENTS",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": "true"
      },
      {
        "identifierType": {
          "name": "VCURL",
          "dataType": "STRING",
          "isPartOfUniqueness": true
        },
        "value": "https://12.345.678.9"
      },
      {
        "identifierType": {
          "name": "VM_FOLDER_DISABLED",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
          "name": "VM_LIMIT",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      }
    ]
  },
  "description": "A vCenter Adapter Instance for VC 12.345.678.9",
  "collectorId": 1,
  "collectorGroupId": "909c2fbf-2c2c-4957-9a75-21bf2a887d31",
  "credentialInstanceId": "65081a8d-d462-43b2-b4e0-596eaf3d497e",
  "monitoringInterval": 5,
  "adapter-certificates": [
```

```
    {
      "thumbprint": "2520fb4351bc91ee7b82ef7cc54a8d88fa893da9",
      "certificateDetails": "[
        Version: V3 Subject: C=US, CN=12.345.678.9
        Signature Algorithm: SHA256withRSA, OID = 1.2.840.113549.1.1.11
        Key: Sun RSA public key, 2048 bits modulus: ...
        Validity: [From: Wed Jul 15 19:26:51 UTC 2015, To: Tue Jul 08 11:26:30 UTC 2025]
        Issuer: O=W12R2UINanduVC, C=US, DC=local, DC=vsphere, CN=CA ...
          ...
      ]"
    }
  ],
  ...
  "id": "a97bd204-e3e5-404b-a219-e2b20cf158d2"
}
```

Response in JSON:

```
{
  "resourceKey": {
    "name": "VC Adapter Instance",
    "adapterKindKey": "VMWARE",
    "resourceKindKey": "VMwareAdapter Instance",
    "resourceIdentifiers": [
      {
        "identifierType": {
          "name": "AUTODISCOVERY",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": "true"
      },
      {
        "identifierType": {
          "name": "DISABLE_COMPUTATION_BASED_ON_CONSUMERS",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
          "name": "DV_PORT_GROUP_DISABLED",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
        "name": "DVS_DISABLED",
        "dataType": "STRING",
        "isPartOfUniqueness": false
        },
        "value": ""
```

```
      },
      {
        "identifierType": {
          "name": "PROCESSCHANGEEVENTS",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": "true"
      },
      {
        "identifierType": {
          "name": "VCURL",
          "dataType": "STRING",
          "isPartOfUniqueness": true
        },
        "value": "https://12.345.678.9"
      },
      {
        "identifierType": {
          "name": "VM_FOLDER_DISABLED",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      },
      {
        "identifierType": {
          "name": "VM_LIMIT",
          "dataType": "STRING",
          "isPartOfUniqueness": false
        },
        "value": ""
      }
    ]
  },
  "description": "A vCenter Adapter Instance for VC 12.345.678.9",
  "collectorId": 1,
  "collectorGroupId": "909c2fbf-2c2c-4957-9a75-21bf2a887d31",
  "credentialInstanceId": "65081a8d-d462-43b2-b4e0-596eaf3d497e",
  "monitoringInterval": 5,
  ...
  "id": "a97bd204-e3e5-404b-a219-e2b20cf158d2"
}
```


The response is same as the request body without the `adapter-certificates` section.

Start Monitoring the New Adapter Instance

After the creating the adapter instance and configuring vRealize Operations to recognize a valid certificate, start monitoring and collecting data. Your PUT request provides the UUID of the adapter instance used to discover new objects.

Prerequisites

Verify that you have the UUID of the newly created adapter instance. See the response in Adapter Instance.

Procedure

- Make a PUT request to start monitoring with the new adapter instance.

```
PUT https://vrealize.example.com/suite-api/api/adapters/<adapter_UUID>/monitoringstate/
start
```

Example: Discover Objects and Collect Data

This example starts the adapter monitoring process using the adapter instance ID from the PUT request that created the adapter instance.

Request header:

```
PUT https://vrealize.example.com/suite-api/api/adapters/a97bd204-e3e5-404b-a219-e2b20cf158d2/
monitoringstate/start
```