OpenManage Integration for VMware vCenter Version 5.4

API Guide

October 2021
Rev. A00

Notes, cautions, and warnings

NOTE: A NOTE indicates important information that helps you make better use of your product.

CAUTION: A CAUTION indicates either potential damage to hardware or loss of data and tells you how to avoid the problem.

WARNING: A WARNING indicates a potential for property damage, personal injury, or death.

Contents

1

Overview

OpenManage Integration for VMware vCenter (OMIVV) provides RESTful APIs to enable automation. The support for RESTful API is available from version 5.2.0. The APIs are compliant with OpenAPI Specification (OAS) 3.0.0.

This document is intended for a vCenter Administrator. It is assumed that the reader is familiar with REST APIs and programmatic interaction with REST APIs. Any programming language can be used to create applications that interface with these APIs. You can run the APIs using an API Client such as Curl or Postman.

OMIVV sample API scripts are available at https://github.com/dell/omivv.

Topics:

What is new in this release

Table 1. List of new and enhanced APIs

Summary of new host management APIs	APIs
GET to retrieve host hardware FRUs	`Services/InventoryService/Hosts/{id}/Hardware/Frus`—Gets the host hardware FRUs
GET to retrieve host hardware NICs	`/Services/InventoryService/Hosts/{id}/Hardware/Nics`—Gets the host hardware NICs.
GET to retrieve host hardware PCI slots information	`/Services/InventoryService/Hosts/{id}/Hardware/PciSlots`—Get the host hardware PCI slots information.
GET to retrieve hardware remote access card	`/Services/InventoryService/Hosts/{id}/Hardware/RemoteAccessCard`—Get the host hardware remote access card.
GET to retrieve host storage details	`/Services/InventoryService/Hosts/{id}/Storage`—Gets the host storage details.
GET to retrieve host power monitoring details	`/Services/InventoryService/Hosts/{id}/PowerMonitoring`—Gets the host power monitoring details
GET to retrieve host system event log	`Services/InventoryService/Hosts/{id}/SystemEventLog`—Gets the host system event log.

Access OMIVV API documentation at Dell Technologies Developer Portal

The Dell Technologies Developer Portal is a single location for sharing and disseminating developer-focused documentation in an easily consumable and consistent format. Using the Developer Portal, developers can discover, explore, and test Dell APIs needed to integrate with other solutions and create custom automation. In addition to consolidating consistent content within a single portal, Dell enterprise products enable the 'build once or re-use many' mindset for faster implementation and easier automation of Dell's portfolio.

The API documentation for OpenManage Integration for VMware vCenter (OMIVV) is now available at Dell Technologies Developer Portal along with all available Dell APIs.

2

Get started

Topics:

Prerequisites

Ensure that:

- User profile used to access API has administrator privileges
- vCenter profile has required Dell privileges. For more information, see the **Required privileges for non-administrator users** topic in User's Guide.
- Host Credential Profile, Cluster Profile, and System Profile are created in OMIVV User Interface (UI) and use the profiles as required. For more information, see the **Create Host Credential Profile**, **Create Cluster Profile**, and **Create System Profile** topics in User's Guide.
- The hosts are management-complaint. For more information, see the **Management Compliance** topic in User's Guide.

Base URI

The base URI for any OMIVV RESTful API request must be in the following format:

`https://{OMIVV Address}/Spectre/api/rest/v1/Services/`

OMIVV Address: IPv4 address or Fully Qualified Domain Name (FQDN).

NOTE: All the URIs and query parameters are case-sensitive.

Request headers

The request header represents headers in the client HTTPS request that are used to communicate client preferences to the service end-point. The bearer token is the request header.

Rate limit

At a given time, only three unique client IPs can be active and can create total ten sessions.

OMIVV supports 100 requests per minute. If there are multiple sessions, make sure that each session has maximum 10 API requests per minute in parallel. However, if there is a single session, parallel API requests can go up to 100 per minute.

If there are more than ten sessions, **Request limit per minute exceeded** message is sent to the API Client.

Each OMIVV session is correspond to one vCenter. Ensure that you sequence the communication if you have more than ten vCenters registered in OMIVV because you can start only 10 sessions at a time.

The total number of failure login attempts that is allowed is six (count includes failed login attempts in administration console or REST API or use of invalid token for REST API access). After six failed login attempts, the account gets locked.

The account lockout duration is one minute.

You cannot start a new session when the account is locked. But, current active session remains active.

During the lockout period, any REST API call will not work except logoff API. An appropriate error message is displayed.

3

Security

The authentication and authorization flows are prerequisites to any API invocation. The request is forwarded to the CXF server defined for the REST API after the invocation.

If there is authorization and authentication failure, an appropriate error code with message is sent to the API client.

On success, the request is forwarded to the appropriate REST service endpoint (API) defined on the CXF servlet.

Topics:

OMIVV authentication

OMIVV web server handles incoming web requests and routes them to the REST endpoints.

- Authentication server performs the following:
  - Accepts login requests and provide the bearer token. This token is generated using the JWT scheme that includes a header, body, and footer.
  - Accepts log out requests which closes the session
- API server: Service all defined REST endpoints except endpoints that are related to authentication.

API clients establish a session with the API server using the endpoint: `/Services/AuthenticationService/login`.

OMIVV user credentials are required to authenticate a client of the RESTful API. Only user profile with admin privilege is allowed to log in as an API user. Other user profiles with privileges like Read only cannot access the API.

Sample code to create an OMIVV session:

```
```json
def login_with_OMIVV (omivvIP,username,domain, password):
    baseurl ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/AuthenticationService/
login"
    postBodyData={"apiUserCredential" : {"username":username,"domain" :
domain,"password" : password}}
    jsonReponse = requests.post(baseurl, json=postBodyData, verify=False)
    if(jsonReponse.status_code == 200):
        bearerToken = jsonReponse['accessToken']
        return bearerToken
    else:
        print("Login failed")
```
```

omivvIP can be a valid OMIVV IP or FQDN.

Bearer Token

Each session that is created using an authentication service contains a bearer token that is generated using the JWT scheme.

Expiration period for bearer token is 60 minutes. If you are using the token after 60 minutes, **Token is already expired** message is sent to the API client.

The account lockout duration is one minute. If an account lockout happens after creating the Bearer token, bearer token cannot be used during lockout duration (one minute). After an account lockout duration, same Bearer token can be used until it expires.

vCenter authorization

Authorization flow validates the following:

- Session token
- vCenter user credentials for the required permissions to run the API

Validate session token

OMIVV validates the token that is received from the API client against the following:

- Tampering
- Session validity

Validate vCenter user credentials

APIs are authorized against the vCenter user credentials, when required. Set an operational context which indicates the registered vCenter and the associated user credentials.

An operation context is required to ensure the permissions available to the user. You can set an operation context by invoking the corresponding API.

This context is unique to an OMIVV user session, and only a single context can be active for a session at a time. The different OMIVV user sessions can invoke APIs using a different vCenter context.

The required vCenter user privileges for all other APIs are verified when you trigger the API.

The following is the sample code to set the operational context:

```
```json
def setOperationalContext(omivvIP, bearerToken, vcenterId,
vcenterUsername,vCenterDomain, vCenterPassword):
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
OperationalContext"
    postBodyData={"consoleId" : vcenterId, "consoleUserCredential":
{"username":vcenterUsername,"domain" : vCenterDomain,"password" : vCenterPassword}}
    head = {'Authorization': 'Bearer ' + bearerToken}
    jsonReponse = requests.post(url, json=postBodyData, verify=False,headers=head)
    if(jsonReponse.status_code == 204):
        return pass;
    else:
        print("setOperationalContext failed.")
```
```

The vCenter user privilege is not required to access the following APIs:

- `/Services/AuthenticationService/login`
- `/Services/AuthenticationService/logoff`
- `/Services/ConsoleService/OperationalContext`
- `/Services/ConsoleService/Consoles`
- `/Services/ConsoleService/Consoles/{id}`

4

Use cases

Topics:

Navigate vCenter heirachy

Prerequisites

Ensure that you have verified all the pre-requisites mentioned in the Prerequisites on page 7.

About this task

The vSphere Client lists all the registered vCenters, hosts, clusters, and datacenters in a hierarchical manner.

You can use OMIVV APIs to retrieve the required details of host, cluster, vCenter, and datacenter listed in the vSphere Client.

This topic describes how you can:

- Retrieve host ID using Service Tag
- Retrieve cluster ID using cluster name
- Retrieve vCenter ID using vCenter IP
- Retrive cluster ID for a given host ID
- Get list of registered vCenters
- Get host details
- Retrieve host ID using Host IP
- Retrieve host details using host ID

Steps

Start an OMIVV session using OMIVV user credentials. OMIVV credentials are required to authenticate a client of the RESTful API.

For more information, see OMIVV authentication on page 9.

- Retrieve host ID using Service Tag

Sample code to retrieve the host ID using Service Tag:

```
```json
def get_host_id_by_service_tag(bearer_token,ip_address, vcenter_ip, serviec_tag,
vc_username, vc_domain, vc_password):
    """Authenticate with OMIVV and enumerate reports."""

    try:
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address, bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
        else:
            vcenter_id = None;
            registered_vc_array = json.loads(registered_vcenter_list)
```
```

```
            for vc in registered_vc_array:
                if vcenter_ip.upper() == vc['ip'].upper():
                    vcenter_id = vc['id']

            vcenter_tree_response = get_vcenter_tree_from_omivv(ip_address,vcenter_id,
bearer_token)
            flag = True;
            cluster_list = []
            if vcenter_tree_response is not None:
                vcenter_datacenters = json.loads(vcenter_tree_response)
                vcenter_datacenters_List = vcenter_datacenters['datacenters']
                for vcenter_datacenter in vcenter_datacenters_List:

                    hosts_list = vcenter_datacenter["hosts"]
                    if hosts_list is not None:
                            for host in hosts_list:
                                if host['serviceTag'] == serviec_tag:
                                    host_response = json.dumps(host, indent=4,
sort_keys=True)
                                    print(host_response['id'])
                                    flag = False
                    cluster_array = vcenter_datacenter['clusters']
                    for cluster in cluster_array:
                        href = cluster['href']
                        cluster_list.append(href)

        if flag:
            """set operational context"""
            set_operational_context(ip_address,vcenter_id, bearer_token, vc_username,
vc_domain, vc_password)

            """ Found the host inside the clusters"""
            cluster_flag = False
            for cluster_href in cluster_list:
                """Get cluster details"""
                cluster_details = get_cluster_details(cluster_href, bearer_token)
                cluster_json_response = cluster_details.json()
                hosts_list = cluster_json_response['hosts']
                for host in hosts_list:
                    if host['serviceTag'] == serviec_tag:
                        host_response = json.dumps(host, indent=4, sort_keys=True)
                        print(host_response)
                        cluster_flag = True
                        break
                if cluster_flag:
                        break;

            if cluster_flag == False:
                print("Host with serice tag " + serviec_tag + " not found.")
    except:
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])


def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response;
    else:
        print("Get registered vCenter list failed")

def get_vcenter_tree_from_omivv(OMIVV_ip, vcenter_id, session_token):
    url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles/" + vcenter_id
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
```

```
        print("Get registered vCenter list failed")

def get_cluster_details(url, session_token):
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        return response
    else:
        print(response.status_code)
        print("Get cluster details  failed")

def set_operational_context(omivv_ip, vcenter_id, session_token, vcenter_username,
vcenter_domain, vcenter_password):
    base_url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
OperationalContext"
    postBodyData = {"consoleId": vcenter_id,
                    "consoleUserCredential": {"username": vcenter_username, "domain":
vcenter_domain,
                                              "password": vcenter_password}}
    head = {'Authorization': 'Bearer ' + session_token}
    jsonReponse = requests.post(base_url, json=postBodyData, verify=False, headers=head)
    if (jsonReponse.status_code != 204):
         print("setOperationalContext failed.")
```

- Retrieve cluster ID using cluster name

Sample code to retrieve the cluster ID:

```
```json
def get_cluster_id_by_cluster_name(bearer_token, ip_address, vcenter_ip, cluster_name):
    """Authenticate with OMIVV and enumerate reports."""
    try:
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address, bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
        else:
            vcenter_id = None;
            registered_vc_array = json.loads(registered_vcenter_list)
            for vc in registered_vc_array:
                if vcenter_ip.upper() == vc['ip'].upper():
                    vcenter_id = vc['id']

            vcenter_tree_response = get_vcenter_tree_from_omivv(ip_address,vcenter_id,
bearer_token)
            found = False
            if vcenter_tree_response is not None:
                vcenter_datacenters = json.loads(vcenter_tree_response)
                vcenter_datacenters_List = vcenter_datacenters['datacenters']
                for vcenter_datacenter in vcenter_datacenters_List:
                    cluster_list = vcenter_datacenter["clusters"]
                    if cluster_list is not None:
                            for cluster in cluster_list:
                                if cluster['name'] == cluster_name:
                                    print("Cluster name: " + cluster_name + ",  Cluster
id: " + cluster['id'])
                                    found = True
                                    break
                    if found:
                        break
            if found == False:
                print("Cluster " + cluster_name + " not found.")
        """ Log out from OMIVV """
        logout(ip_address, bearer_token)
    except:
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])
def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
```
```

```
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response;
    else:
        print("Get registered vCenter list failed")
def get_vcenter_tree_from_omivv(OMIVV_ip, vcenter_id, session_token):
    url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles/" + vcenter_id
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get registered vCenter list failed")

```

- Retrieve vCenter ID using vCenter IP

Sample code retrieve the vCenter ID using the vCenter IP:

```
```json
def get_vcenter_id(bearer_token, ip_address, vcenter_ip):
    """Authenticate with OMIVV and enumerate reports."""

    try:
        bearer_token = get_bearer_token(ip_address, user_name, password)
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address, bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
        else:
            registered_vc_array = json.loads(registered_vcenter_list)
            for vc in registered_vc_array:
                if vcenter_ip.upper() == vc['ip'].upper():
                    print(vc['id'])
                    break

    except:
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])


def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response;
    else:
        print("Get registered vCenter list failed")

```
```

- Retrieve cluster ID for a given host ID

Sample code to retrive the cluster ID using host ID:

```
```json
def get_cluster_id_by_host_id(bearer_token, ip_address, vcenter_ip, host_id,
vc_username, vc_domain, vc_password):
    """Authenticate with OMIVV and enumerate reports."""

    try:
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address, bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
```
```

```
        else:
            vcenter_id = None;
            registered_vc_array = json.loads(registered_vcenter_list)
            for vc in registered_vc_array:
                if vcenter_ip.upper() == vc['ip'].upper():
                    vcenter_id = vc['id']

            vcenter_tree_response = get_vcenter_tree_from_omivv(ip_address,vcenter_id,
bearer_token)
            flag = True;
            cluster_list = []
            dacenter_name = "";
            if vcenter_tree_response is not None:
                vcenter_datacenters = json.loads(vcenter_tree_response)
                vcenter_datacenters_List = vcenter_datacenters['datacenters']
                for vcenter_datacenter in vcenter_datacenters_List:

                    hosts_list = vcenter_datacenter["hosts"]
                    if hosts_list is not None:
                            for host in hosts_list:
                                if host['id'] == host_id:
                                    datacenter_name = vcenter_datacenter['name']
                                    flag = False
                    cluster_array = vcenter_datacenter['clusters']
                    for cluster in cluster_array:
                        href = cluster['href']
                        cluster_list.append(href)

        if flag:
            """set operational context"""
            set_operational_context(ip_address,vcenter_id, bearer_token, vc_username,
vc_domain, vc_password)

            """ Found the host inside the clusters"""
            cluster_flag = False
            cluster_id = "";
            for cluster_href in cluster_list:
                """Get cluster details""
                cluster_details = get_cluster_details(cluster_href, bearer_token)
                cluster_json_response = cluster_details.json()
                hosts_list = cluster_json_response['hosts']
                for host in hosts_list:
                    if host['id'] == host_id:
                        host_response = json.dumps(host, indent=4, sort_keys=True)
                        cluster_flag = True
                        print("Host id: " + host_id + ", cluster id: " +
cluster_json_response['id'])
                        break
                if cluster_flag:
                        break;

            if cluster_flag == False:
                print("Host with host id " + host_id + " not found.")
        else:
            print("Host is not inside a cluster. It is inside the datacenter:  " +
datacenter_name)
    except:
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])

def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response;
    else:
        print("Get registered vCenter list failed")

def get_vcenter_tree_from_omivv(OMIVV_ip, vcenter_id, session_token):
    url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles/" + vcenter_id
```

```
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get registered vCenter list failed")

def get_cluster_details(url, session_token):
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        return response
    else:
        print(response.status_code)
        print("Get cluster details  failed")

def set_operational_context(omivv_ip, vcenter_id, session_token, vcenter_username,
vcenter_domain, vcenter_password):
    base_url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
OperationalContext"
    postBodyData = {"consoleId": vcenter_id,
                    "consoleUserCredential": {"username": vcenter_username, "domain":
vcenter_domain,
                                              "password": vcenter_password}}
    head = {'Authorization': 'Bearer ' + session_token}
    jsonReponse = requests.post(base_url, json=postBodyData, verify=False, headers=head)
    if (jsonReponse.status_code != 204):
         print("setOperationalContext failed.")

```
```

- Get list of registered vCenters

Sample code to get the list of registered vCenters:

```
```json
def get_regitered_vcenter(bearer_token,ip_address, vcenter_ip, serviec_tag, vc_username,
vc_domain, vc_password):
    """Authenticate with OMIVV and enumerate reports."""

    try:
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address, bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
        else:
            print(registered_vcenter_list)
    except:
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])


def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response;
    else:
        print("Get registered vCenter list failed")
```
```

- Get host details

Sample code to get the host details:

```
```json
def get_host_detail_by_service_tag(bearer_token,ip_address, vcenter_ip, serviec_tag,
vc_username, vc_domain, vc_password):
```

```
    """Authenticate with OMIVV and enumerate reports."""

    try:
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address, bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
        else:
            vcenter_id = None;
            registered_vc_array = json.loads(registered_vcenter_list)
            for vc in registered_vc_array:
                if vcenter_ip.upper() == vc['ip'].upper():
                    vcenter_id = vc['id']

            vcenter_tree_response = get_vcenter_tree_from_omivv(ip_address,vcenter_id,
bearer_token)
            flag = True;
            cluster_list = []
            if vcenter_tree_response is not None:
                vcenter_datacenters = json.loads(vcenter_tree_response)
                vcenter_datacenters_List = vcenter_datacenters['datacenters']
                for vcenter_datacenter in vcenter_datacenters_List:

                    hosts_list = vcenter_datacenter["hosts"]
                    if hosts_list is not None:
                            for host in hosts_list:
                                if host['serviceTag'] == serviec_tag:
                                    host_response = json.dumps(host, indent=4,
sort_keys=True)
                                    print(host_response)
                                    flag = False
                    cluster_array = vcenter_datacenter['clusters']
                    for cluster in cluster_array:
                        href = cluster['href']
                        cluster_list.append(href)

        if flag:
            """set operational context"""
            set_operational_context(ip_address,vcenter_id, bearer_token, vc_username,
vc_domain, vc_password)

            """ Found the host inside the clusters"""
            cluster_flag = False
            for cluster_href in cluster_list:
                """Get cluster details"""
                cluster_details = get_cluster_details(cluster_href, bearer_token)
                cluster_json_response = cluster_details.json()
                hosts_list = cluster_json_response['hosts']
                for host in hosts_list:
                    if host['serviceTag'] == serviec_tag:
                        host_response = json.dumps(host, indent=4, sort_keys=True)
                        print(host_response)
                        cluster_flag = True
                        break
                if cluster_flag:
                        break;

            if cluster_flag == False:
                print("Host with serice tag " + serviec_tag + " not found.")
    except:
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])


def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response;
    else:
```

```
        print("Get registered vCenter list failed")

def get_vcenter_tree_from_omivv(OMIVV_ip, vcenter_id, session_token):
    url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles/" + vcenter_id
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get registered vCenter list failed")

def get_cluster_details(url, session_token):
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        return response
    else:
        print(response.status_code)
        print("Get cluster details  failed")

def set_operational_context(omivv_ip, vcenter_id, session_token, vcenter_username,
vcenter_domain, vcenter_password):
    base_url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
OperationalContext"
    postBodyData = {"consoleId": vcenter_id,
                    "consoleUserCredential": {"username": vcenter_username, "domain":
vcenter_domain,
                                              "password": vcenter_password}}
    head = {'Authorization': 'Bearer ' + session_token}
    jsonReponse = requests.post(base_url, json=postBodyData, verify=False, headers=head)
    if (jsonReponse.status_code != 204):
         print("setOperationalContext failed.")

```
```

- Retrieve host ID using Host IP

Sample code to retrieve the host ID:

```
```json
def get_host_id_by_service_tag(bearer_token,ip_address, vcenter_ip, hostIp, vc_username,
vc_domain, vc_password):
    """Authenticate with OMIVV and enumerate reports."""

    try:
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address, bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
        else:
            vcenter_id = None;
            registered_vc_array = json.loads(registered_vcenter_list)
            for vc in registered_vc_array:
                if vcenter_ip.upper() == vc['ip'].upper():
                    vcenter_id = vc['id']

            vcenter_tree_response = get_vcenter_tree_from_omivv(ip_address,vcenter_id,
bearer_token)
            flag = True;
            cluster_list = []
            if vcenter_tree_response is not None:
                vcenter_datacenters = json.loads(vcenter_tree_response)
                vcenter_datacenters_List = vcenter_datacenters['datacenters']
                for vcenter_datacenter in vcenter_datacenters_List:

                    hosts_list = vcenter_datacenter["hosts"]
                    if hosts_list is not None:
                           for host in hosts_list:
                                if host['hostip'] == hostIp:
                                    host_response = json.dumps(host, indent=4,
```

```
sort_keys=True)
                                    print(host_response['id'])
                                    flag = False
                    cluster_array = vcenter_datacenter['clusters']
                    for cluster in cluster_array:
                        href = cluster['href']
                        cluster_list.append(href)

        if flag:
            """set operational context"""
            set_operational_context(ip_address,vcenter_id, bearer_token, vc_username,
vc_domain, vc_password)

            """ Found the host inside the clusters"""
            cluster_flag = False
            for cluster_href in cluster_list:
                """Get cluster details""
                cluster_details = get_cluster_details(cluster_href, bearer_token)
                cluster_json_response = cluster_details.json()
                hosts_list = cluster_json_response['hosts']
                for host in hosts_list:
                    if host['hostip'] == hostIp:
                        host_response = json.dumps(host, indent=4, sort_keys=True)
                        print(host_response)
                        cluster_flag = True
                        break
                if cluster_flag:
                        break;

            if cluster_flag == False:
                print("Host with serice tag " + hostIp + " not found.")
    except:
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])


def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response;
    else:
        print("Get registered vCenter list failed")

def get_vcenter_tree_from_omivv(OMIVV_ip, vcenter_id, session_token):
    url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles/" + vcenter_id
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get registered vCenter list failed")

def get_cluster_details(url, session_token):
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        return response
    else:
        print(response.status_code)
        print("Get cluster details  failed")

def set_operational_context(omivv_ip, vcenter_id, session_token, vcenter_username,
vcenter_domain, vcenter_password):
    base_url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
OperationalContext"
    postBodyData = {"consoleId": vcenter_id,
                    "consoleUserCredential": {"username": vcenter_username, "domain":
vcenter_domain,
```

```
                                                "password": vcenter_password}}
    head = {'Authorization': 'Bearer ' + session_token}
    jsonReponse = requests.post(base_url, json=postBodyData, verify=False, headers=head)
    if (jsonReponse.status_code != 204):
         print("setOperationalContext failed.")
```

- Retrieve host details using host ID

Retrieve the host IP, hostname, management IP, ServiceTag, model, and system ID for a host using host ID.

- host IP: Hypervisor IP address
- hostname: DNS host name of the hypervisor. If hostName is not configured value will be empty.
- management IP : iDRAC IP address of the server
- ServiceTag : Service tag of the server
- model : Dell Technologies server model
- system ID : Unique ID used to determine the server model. Server model will be same for all servers of a particular model

Sample code to retrieve the host details using host ID:

```
```json
def get_vcenter_tree(ip_address, user_name, password, bearer_token, vcenter_ip,
vc_username, vc_domain, vc_password, host_id, details):
    """Authenticate with OMIVV and enumerate reports."""
    try:
        local_bearer_token=None
        if(bearer_token == None):
            local_bearer_token = get_bearer_token(ip_address, user_name, password)
        else:
            local_bearer_token = bearer_token
        """ Get all registered vcenter """
        registered_vcenter_list =  get_registered_vcenter_list(ip_address,
local_bearer_token)
        if(registered_vcenter_list is None):
            print("No vcenter is regsitered with this OMIVV")
            return
        else:
            vcenter_id = None;
            registered_vc_array = json.loads(registered_vcenter_list)
            for vc in registered_vc_array:
                if vcenter_ip.upper() == vc['ip'].upper():
                    vcenter_id = vc['id']

            #Set operational Context.
            set_operational_context(ip_address, vcenter_id, local_bearer_token,
vc_username, vc_domain, vc_password)
            # Get host subsystem heath
            if(host_id is None):
                print("Pass the host_id")
            else:
                get_host_details(ip_address, local_bearer_token, host_id, details)

        """ Log out from OMIVV """
        if(bearer_token == None):
            logout(ip_address, local_bearer_token)
    except:
        traceback.print_exc()
        print("get_vcenter_tree: Unexpected error:", sys.exc_info()[0])


def get_bearer_token(ip_address, user_name, password):
    try:
        baseurl = "https://" + ip_address + "/Spectre/api/rest/v1/Services/
AuthenticationService/login"
        postBodyData = {"apiUserCredential": {"username": user_name, "domain": "",
"password": password}}
        response = requests.post(baseurl, json=postBodyData, verify=False)
        if (response.status_code  == 200):
            json_response = response.json()
            bearerToken = json_response['accessToken']
            return bearerToken
        else:
```
```

```
            print("Login to OMIVV failed")
    except:
        print("get_bearer_token: Unexpected error:", sys.exc_info()[0])

def logout(omivvIP,session_token):
    try:
        head = {'Authorization': 'Bearer ' + session_token}
        url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/
AuthenticationService/logoff"
        response = requests.post(url, verify=False, headers=head)
        if (response.status_code != 200):
            print("Log out failed.")
    except:
        print("logout: Unexpected error:", sys.exc_info()[0])

def get_registered_vcenter_list(omivvIP,bearerToken):
    """Get all registered vcenter with this OMIVV"""
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get registered vCenter list failed")

def set_operational_context(omivv_ip, vcenter_id, session_token, vcenter_username,
vcenter_domain, vcenter_password):
    base_url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
OperationalContext"
    postBodyData = {"consoleId": vcenter_id,
                    "consoleUserCredential": {"username": vcenter_username, "domain":
vcenter_domain,
                                              "password": vcenter_password}}
    head = {'Authorization': 'Bearer ' + session_token}
    jsonReponse = requests.post(base_url, json=postBodyData, verify=False, headers=head)
    if (jsonReponse.status_code != 204):
        print("setOperationalContext failed.")

def get_host_details(omivv_ip, bearer_token, host_id, details_str):
    """ Get the Host details value"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/InventoryService/
Hosts/" + host_id + "/" + details_str
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        print(json_response)
    else:
        print("Get host details failed.")


if __name__ == '__main__':
    urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

    PARSER = argparse.ArgumentParser(description=__doc__,
                                     formatter_class=RawTextHelpFormatter)
    PARSER.add_argument("--ip", "-i", required=True, help="OMIVV Appliance IP")
    PARSER.add_argument("--user", "-u", required=False,
                        help="Username for OMIVV Appliance", default="admin")
    PARSER.add_argument("--password", "-p", required=False, default=None,
                        help="Password for OMIVV Appliance")
    PARSER.add_argument("--bearertoken", "-bt", required=False, default=None,
                        help="bearertoken for OMIVV Appliance")
    PARSER.add_argument("--vcenterip", "-vcip", required=True,
                        help="IP for vcenter")
    PARSER.add_argument("--vcusername", "-vcuser", required=True,
                        help="username of vcenter")
    PARSER.add_argument("--vcdomain", "-vcdomain", required=True,
                        help="domain of the vcenter")
    PARSER.add_argument("--vcpassword", "-vcpass", required=True,
                        help="password of vcenter")
```

```
    PARSER.add_argument("--hostId", "-hId", required=True,
                        help="Id of the host")
    PARSER.add_argument("--getdetails", "-getd", required=True,
                        help="get details of the host. Accepted value: 1.SubSystemHealth
2.HostOverview, 3.FirmwareInventory, 4.Warranty")


    ARGS = PARSER.parse_args()
    if(ARGS.password == None and ARGS.bearertoken == None):
        print("Pass either the password or the bearer token")

    get_vcenter_tree(ARGS.ip, ARGS.user, ARGS.password, ARGS.bearertoken,
ARGS.vcenterip,  ARGS.vcusername, ARGS.vcdomain, ARGS.vcpassword, ARGS.hostId,
ARGS.getdetails)
```

Retrieve firmware repository data

Prerequisites

Ensure that you have verified all the pre-requisites mentioned in the Prerequisites on page 7.

About this task

Software components needs to be identified from a desired firmware repository before updating the firmware associated to those components on a cluster or an individual host.

This topic describes various steps to retrieve the software components details of a host.

Steps

1. Start an OMIVV session using OMIVV user credentials. OMIVV credentials are required to authenticate a client of the RESTful API.

   For more information, see OMIVV authentication on page 9.

   Sample code to start an OMIVV session:

   ```json
   def login_with_OMIVV (omivvIP,username,domain, password):
       baseurl ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/
   AuthenticationService/login"
       postBodyData={"apiUserCredential" : {"username":username,"domain" :
   domain,"password" : password}}
       jsonReponse = requests.post(baseurl, json=postBodyData, verify=False)
       if(jsonReponse.status_code == 200):
           bearerToken = jsonReponse['accessToken']
           return bearerToken
       else:
           print("Login failed")
   ```

2. Set the vCenter context. After you log in to OMIVV, select or identify a vCenter on which you want to perform operation. This is done by setting an operational context.

   For more information, see OMIVV authentication on page 9.

   Before setting the vCenter Context, get the list of registered vCenters.

   Sample code to get the list of registered vCenters:

   ```json
   def getRegisteredVcenterList(omivvIP,bearerToken):
       url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
   Consoles"
       head = {'Authorization': 'Bearer ' + bearerToken}
       jsonReponse = requests.get(url, verify=False, headers=head)
       if(jsonReponse.status_code == 200):
           return Response
   ```

```
    else:
        print("Get registered vCenter list failed")
```

After you get the list of registered vCenters, set the vCenter context.

Sample code to set the vCenter context:

```
```json
def setOperationalContext(omivvIP, bearerToken, vcenterId,
vcenterUsername,vCenterDomain, vCenterPassword):
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
OperationalContext"
    postBodyData={"consoleId" : vcenterId, "consoleUserCredential":
{"username":vcenterUsername,"domain" : vCenterDomain,"password" : vCenterPassword}}
    head = {'Authorization': 'Bearer ' + bearerToken}
    jsonReponse = requests.post(url, json=postBodyData, verify=False,headers=head)
    if(jsonReponse.status_code == 204):
        return pass;
    else:
        print("setOperationalContext failed.")
  pass
```
```

3. Identify system ID of the host for retrieving repository inventory. To get the repository inventory for the host, identify the host ID based on the Service Tag of the host, and then retrieve the system ID using host ID.

   Invoke the `/services/consoleServices/Console/{1d}/Hosts?serviceTags={serviceTag1}&servicetas={serviceTags2}` API to get the host ID.

   Sample code to get the Host ID:

```
```
def get_host_id_from_serviceTag(omivvIP, bearerToken, vcenter_ip,host_servicetag):
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    response = requests.get(url, verify=False, headers=head)
    print(response)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        registered_vc_array = json.loads(json_response)
        for vc in registered_vc_array:
            if vcenter_ip.upper() == vc['ip'].upper():
                vcenter_id = vc['id']
                url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/
ConsoleService/Consoles/" + vcenter_id + "/Hosts?serviceTags=" + host_servicetag
                head = {'Authorization': 'Bearer ' + bearerToken}
                print(url)
                response = requests.get(url, verify=False, headers=head)
                print(response)
                if (response.status_code == 200):
                    json_response = json.dumps(response.json(), indent=4,
sort_keys=True)
                    print(json_response)
                    return json_response
                else:
                    print("Get host details failed")
    else:
        print("Get registered vCenter list failed")
```
```

   Invoke the `/services/consoleServices/Consoles/{id}/Hosts/{hostid}` to get the system ID.

   Sample code to get the system ID:

```
```
def get_vcenter_tree_from_omivv(omivvIP, bearerToken, vcenter_ip,host_id):
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken}
```
```

```
    response = requests.get(url, verify=False, headers=head)
    print(response)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        registered_vc_array = json.loads(json_response)
        for vc in registered_vc_array:
            if vcenter_ip.upper() == vc['ip'].upper():
                vcenter_id = vc['id']
                if(host_id == None):
                    url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/
ConsoleService/Consoles/" + vcenter_id + "/Hosts"
                else:
                    url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/
ConsoleService/Consoles/" + vcenter_id + "/Hosts/" + host_id

                head = {'Authorization': 'Bearer ' + bearerToken}
                response = requests.get(url, verify=False, headers=head)
                if (response.status_code == 200):
                    json_response = json.dumps(response.json(), indent=4,
sort_keys=True)
                    print(json_response)
                    return json_response
                else:
                    print("Get host details failed")
    else:
        print("Get registered vCenter list failed")
```

4. Get the repository profile ID.

   Repository profile contains the firmware and driver repository information. You can use firmwrae repository to update the firmware.

   To get the list of repository profile, ensure that you have the following vCenter privilege: Dell.Inventory.Configure Inventory.

   Sample code to get the repository profile ID:

```
```json
def get_repo_ID(omivvIP,bearerToken):
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/PluginProfileService/
RepositoryProfiles"
    head = {'Authorization': 'Bearer ' + bearerToken}
    jsonReponse = requests.get(url, verify=False, headers=head)
    if(jsonReponse.status_code == 200):
        return jsonReponse.json()
    else:
        print("Get getallRepositoryProfile failed")
  pass
```
```

5. Get the firmware repository inventory details. From step 3 and 4, we have received the system ID and repository profile ID of the inventory details.

   Firmware repository inventory contains the bundle ID and firmware component details. Ensure that you have the following vCenter privilege to get the firmware repository inventory details: Dell.Configuration.Firmware Update

   Sample code to get the firmware repository inventory:

```
```json
def postFirmwareInventoryReportforHost(omivvIP, bearerToken, systemID, bundleID,
repoID, hostID='',clusterID=''):
    url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/UpdateService/
FWReport"
    head = {'Authorization': 'Bearer ' + bearerToken}
    if(hostID != ''):
        postBodyData = {"reportType": "HOST", "hostID": hostID, "repoProfileID":
repoID, "bundleAssociation" : [{"systemId": systemID, "bundleID": bundleID}]}
    else:
        postBodyData = {"reportType": "HOST", "clusterID": clusterID,
"repoProfileID": repoID, "bundleAssociation": [{"systemId": systemID, "bundleID":
bundleID}]}
    jsonReponse = requests.post(url, json=postBodyData, verify=False,headers=head)
    if(jsonReponse.status_code == 200):
```
```

```
        return jsonReponse.json()
    else:
        print("PostFirmwareInventoryReport API is failed.")
```

Update firmware using catalogs

Prerequisites

Before you create firmware update job, ensure that you have:

- Service Tag of the host
- Repository profile name
- Verified all the required pre-requisites mentioned in the Prerequisites on page 7.
- Verified the following for vSAN host and cluster:
  - DRS is enabled
  - Host is not already in maintenance mode
  - vSAN data objects are healthy
- Verified that DRS is enabled for vSphere host and cluster

About this task

This topic explains how you can use OMIVV APIs to update the host firmware components.

Steps

1. Log in to OMIVV. Start an OMIVV session using OMIVV user credentials. OMIVV credentials are required to authenticate a client of the RESTful API.

   For more information, see OMIVV authentication on page 9.

2. Get firmware repository profile ID. You can get the firmware repository profile ID from the repository profile name.

   Sample code to get the repository profile ID:

```
```json
def getRepoId(omivv_ip, bearer_token):
    response = getAllRepositoryProfile(omivv_ip, bearer_token)
        for repo in response:
            if(repo_name == repo['profileName']):
                return repo['id']

def getAllRepositoryProfile(omivvIP,bearerToken):
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/PluginProfileService/
RepositoryProfiles"
    print("url: " + url)
    head = {'Authorization': 'Bearer ' + bearerToken}
    jsonReponse = requests.get(url, verify=False, headers=head)
    if(jsonReponse.status_code == 200):
        return jsonReponse.json()
    else:
        print("Get getallRepositoryProfile failed")

```
```

3. Get Host ID and System ID. Get the host ID and system ID by using the host Service Tag.

   Sample code to get the host ID and system ID:

```
```json
def get_vcenter_tree_from_omivv(OMIVV_ip, vcenter_ip, session_token
,host_service_Tag):
    url ="https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles"
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
```
```

```
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        registered_vc_array = json.loads(json_response)
        for vc in registered_vc_array:
            if vcenter_ip.upper() == vc['ip'].upper():
                vcenter_id = vc['id']
                url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/
ConsoleService/Consoles/" + vcenter_id + "/Hosts"
                head = {'Authorization': 'Bearer ' + session_token}
                response = requests.get(url, verify=False, headers=head)
                if (response.status_code == 200):
                    allHost = response.json()
                    for host in allHost:
                        if (host["serviceTag"] == host_service_Tag):
                            url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/
Services/ConsoleService/Consoles/" + vcenter_id + "/Hosts/" + host['id']
                            head = {'Authorization': 'Bearer ' + session_token}
                            response = requests.get(url, verify=False, headers=head)
                            if (response.status_code == 200):
                                json_response = json.dumps(response.json(), indent=4,
sort_keys=True)
                                print(json_response)
                                return json_response
```

4. Get bundle ID. Get the bundle ID using the repository profile ID received from step 2 and System ID received from step 3.

   Sample code to get the bundle ID:

```
```json
def getBundleId(omivvIP, bearerToken, repoProfileID, systemId):
    url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/
RepositoryManagementService/RepositoryData?repoProfileID=" + repoProfileID +
"&systemId=" + systemId
    head = {'Authorization': 'Bearer ' + bearerToken}
    jsonReponse = requests.get(url, verify=False, headers=head)
    if (jsonReponse.status_code == 200):
        json_object = json.dumps(jsonReponse.json(), indent=4, sort_keys=True)
        print(json_object)
    else:
        print("getRepositoryDataReport API is failed.")

```
```

5. Create firmware inventory report. You can create the firmware inventory report using the repository profile ID, System ID, Host ID, and Bundle ID.

   Sample code to create the firmware inventory report:

```
```json
def postFirmwareInventoryReportforHost(omivvIP, bearerToken, systemID, bundleID,
repoID, hostID):
    url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/UpdateService/
FWReport"
    head = {'Authorization': 'Bearer ' + bearerToken}
    postBodyData = {"reportType": "HOST", "hostID": hostID, "repoProfileID": repoID,
"bundleAssociation" : [{"systemId": systemID, "bundleID": bundleID}]}
    jsonReponse = requests.post(url, json=postBodyData, verify=False,headers=head)
    if(jsonReponse.status_code == 200):
        json_object = json.dumps(jsonReponse.json(), indent=4, sort_keys=True)
        print(json_object)
    else:
        print("PostFirmwareInventoryReport API is failed.")

```
```

6. Create firmware update job

   In this topic, the firmware components that are having Criticality as **Urgent** and **Upgrade Action** as **Update** used in firmware inventory report.

   Once you get all the firmware target details such as repository profile ID, Host ID, Bundle ID, and Package ID, create the firmware update job

When you are creating firmware update job, enter the following job specific custom configuration details:

- preCheck: Prerequisites check before firmware update.
- exit MaintenanceMode: Option to exit from maintenance mode.
- resetIDracAndDeleteJobs: Set this value as TRUE. Clears all the iDRAC jobs present in the Job Queue followed by iDRAC reset before updating firmware on the host.

  If you do not mention any value while running API, OMIVV considers the settings configured on the **Settings** > **Firmware Update Settings** on the User Interface.

- enterMaintenanceModetimeout: Time out value for host to enter maintenance mode. Enter the Maintenance Mode timeout value between 60–1440 minutes. If the wait time goes beyond the specified time, the update jobs fail and enter maintenance task will be canceled or timed out. However, the components may get updated automatically when the host is restarted.
- reboot options: Host reboot options. In this article, we use the host reboot option as Safe reboot.

The possible input value is true or false. Default value is true if the required value is not entered. For cluster-level firmware update, the only allowed value is true.

Sample code to create the firmware update job:

```
```json
def creatHostFirmwareUpdateJob(omivvIP, bearerToken, repoId, bundleID, packageIds,
hostID, jobName):
    url = "https://" + omivvIP + "/Spectre/api/rest/v1/Services/UpdateService/Jobs"
    head = {'Authorization': 'Bearer ' + bearerToken}

    postBodyData = {"jobname": jobName, "jobdesc": "FWTest1Desc", "updateType":
"FIRMWARE", "updateTargetType" : "HOST", "schedule":{"runLater":"true",
"runNow":"false","dateTime":"2021-10-24T02:30:01Z"},
"firmwareRepoProfileID":repoId,"rebootOptions":"FORCEREBOOT","preCheck":"false","firmw
areUpdateTargets":
[{"hostId":hostID,"bundleId":bundleID,"packageIDs":packageIds}],"jobSpecificCustomConf
iguration":
{"exitMaintenanceMode":"true","migratePoweredOffAndSuspendedVMs":"true","resetIDracAnd
DeleteJobs":"false","enterMaintenanceModetimeout":60, "enterMaintenanceModeOption" :
"Ensure accessibility"}}

    jsonReponse = requests.post(url, json=postBodyData, verify=False,headers=head)
    if(jsonReponse.status_code == 202):
        json_object = json.dumps(jsonReponse.json(), indent=4, sort_keys=True)
        print(json_object)
    else:
        print("creatHostFirmwareUpdateJob API is failed.")

```
```

7. Track firmware update job. After you create the firmware update job, you can track the job status by calling the following API received from the response of step 6.

   `/Services/UpdateService/Jobs/{firmware update job ID}`

```
```json
def getHostFirmwareUpdateJobDetails(omivvIP, bearerToken, JobId):
    url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/UpdateService/Jobs/" +
JobId
    head = {'Authorization': 'Bearer ' + bearerToken}
    jsonReponse = requests.get(url, verify=False, headers=head)
    if (jsonReponse.status_code == 200):
        json_object = json.dumps(jsonReponse.json(), indent=4, sort_keys=True)
        print(json_object)
    else:
        print("Get getHostFirmwareUpdateJobDetails failed")
```
```

Retrieve OMIVV license information

Prerequisites

Ensure that:

- OMIVV has at least one license installed.
- You have verified all the pre-requisites mentioned in the Prerequisites on page 7.

About this task

OMIVV requires a valid license to successfully manage the hosts. Multiple licenses can be installed at a time to the appliance. This topic describes how to get list of all licenses installed in OMIVV and retrieve information of individual license.

Steps

Start an OMIVV session using OMIVV user credentials. OMIVV credentials are required to authenticate a client of the RESTful API.

For more information, see OMIVV authentication on page 9.

- Get all licenses

Invoke `/Services/LicenseService/Licenses` API get list of all licenses.

The API response contains following attributes for all the licenses installed:

- Id - License ID generated by OMIVV when the license is installed
- href - URI to individual license in the format /Services/LicenseService/Licenses/{Id}
- objectType - Type of the JSON object. Value is "LicenseInfo"
- entitlementID: Unique license ID
- licenseType - License term. Possible values are Evaluation or Standard or Perpetual
- maxHosts - Maximum hosts that can be managed with this license
- startDate - License start date
- expirationDate - License expiration date
- duration - License validity duration in days
- licenseStatus - Current license status. Possible values are Active or Inactive or Expired

Sample code to get the list of all licenses:

```
```
def get_all_licenses(omivv_ip, bearer_token):
    """Get all license jobs"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/LicenseService/Licenses"
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get all license jobs failed.")
```
```

- Get individual license information

Invoke /Services/LicenseService/Licenses/{Id} API get list of all licenses.

The attributes are same as /Services/LicenseService/Licenses response.

Sample code to get the individual license information:

```
```
def get_license_deatil_by_id(omivv_ip, bearer_token, licesne_id):
    """ Get the license details"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/LicenseService/
Licenses/" + licesne_id;
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        print(json_response)
```
```

```
        return json_response
    else:
        print("Get all license jobs failed.")
```

The sum of maxHosts of all active licenses during the time of API invocation gives the maximum number of hosts that can be managed using the OMIVV instance in context.

Compute server drift from baseline

Prerequisites

Ensure that you have verified all the pre-requisites mentioned in the Prerequisites on page 7.

About this task

Configuration drift deals with calculating drift from the baseline (hardware configuration, firmware, or driver versions). This article describes about identifying the drift between hardware configuration, firmware, and driver versions, run the drift detection job, and then viewing the drift report using OMIVV APIs.

Steps

1. Start an OMIVV session using OMIVV user credentials. OMIVV credentials are required to authenticate a client of the RESTful API.

   For more information, see OMIVV authentication on page 9.
2. Once you log in to OMIVV, select or identify a vCenter on which you want to perform operation. This is done by setting operational context.

   For more information, see OMIVV authentication on page 9.

   Before setting the vCenter Context, get the list of registered vCenter.

   Sample code to get the list of registered vCenters:

   ```json
   def getRegisteredVcenterList(omivvIP,bearerToken):
       url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
   Consoles"
       head = {'Authorization': 'Bearer ' + bearerToken}
       jsonReponse = requests.get(url, verify=False, headers=head)
       if(jsonReponse.status_code == 200):
           return Response
       else:
           print("Get registered vCenter list failed")

   ```

   After you get the list of registered vCenters, set the vCenter context.

   Sample code to set the vCenter context:

   ```json
   def setOperationalContext(omivvIP, bearerToken, vcenterId,
   vcenterUsername,vCenterDomain, vCenterPassword):
       url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
   OperationalContext"
       postBodyData={"consoleId" : vcenterId, "consoleUserCredential":
   {"username":vcenterUsername,"domain" : vCenterDomain,"password" : vCenterPassword}}
       head = {'Authorization': 'Bearer ' + bearerToken}
       jsonReponse = requests.post(url, json=postBodyData, verify=False,headers=head)
       if(jsonReponse.status_code == 204):
           return pass;
       else:
           print("setOperationalContext failed.")
     pass
   ```

3. Create repository profile.

   Repository Profile enables to create or maintain multiple driver or firmware repository profiles.

   These driver or firmware repository profiles can be used to create cluster profile for baselining the clusters.

   Sample code to create repository profile:

```
```
def create_repository_profile(omivv_ip, bearer_token, create_args_file):
    base_url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/
PluginProfileService/RepositoryProfiles"
    """Open the JSON input file"""
    input_file = open(create_args_file,"r")
    input_data = json.load(input_file)
    head = {'Authorization': 'Bearer ' + bearer_token}
    jsonReponse = requests.post(base_url, json=input_data, verify=False, headers=head)
    if (jsonReponse.status_code == 200):
        json_response = json.dumps(jsonReponse.json(), indent=4, sort_keys=True)
        print(json_response)
        print("Successfully creating the repository profile.")
        return json_response
    else:
        print("Error in creating the repository profile.")
        print(json_response)

Sample of input_file:
{
    "name":"LocalOnelineCatalog",
    "description": "LocalOnlineCatalog",
    "globalDefault": "false",
    "repoType": "FIRMWARE",
    "protocolType": "CIFS",
    "uri": "\\\\xx.xx.xx.xx\\ShareFolder\\onlineCatalog\\Catalog.xml",
    "credential": {
        "username": "xxxx",
        "domain": "",
        "password": "xxxx"
    }
}
```
```

4. Create system profile. System profile is used to configure hardware attributes for the desired configuration. Create the system profile in OMIVV UI.
5. Create cluster profile using repository profile and system profile using OMIVV UI.

   A cluster profile enables to capture the configuration baseline (hardware configuration, firmware, or driver versions).

6. View drift detection job. A drift detection job is run to find the comparison between the validated baseline and the server configuration which includes hardware configuration, firmware, and driver versions. Drift job gets triggered automatically after you create the cluster profile.

   Invoke the following API to retrieve all drift job: `/Services/DriftDetectionService/Jobs`

```
```
def get_all_drift_jobs(omivv_ip, bearer_token):
    """Get all drift jobs"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/
DriftDetectionService/Jobs"
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get all drift jobs failed.")
```
```

   Invoke the following API to get cluster details about a specific job: `/Services/DriftDetectionService/Jobs/{id}`

Sample code to get the cluster details:

```
```
def get_drift_job_by_id(omivv_ip, bearer_token, d_job_id):
    """ Get the drift job details by job id"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/
DriftDetectionService/Jobs/" + d_job_id;
    print("url: " + url)
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        print(json_response)
        return json_response
    else:
        print("Get drift job detail by id.")
```
```

Invoke the following API to get host level details about specific drift job: `/Services/DriftDetectionService/Jobs/{id}/Details`

Sample code to get host level details about specific drift job:

```
```
def get_drift_job_deatil_by_id(omivv_ip, bearer_token, d_job_id):
    """ Get the drift job details details"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/
DriftDetectionService/Jobs/" + d_job_id + "/Details";
    print("url: " + url)
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        print(json_response)
        return json_response
    else:
        print("Get all drift jobs details failed.")
```
```

7. View drift report. Drift detection job computes drift details of the hardware, firmware, and driver components from the baseline.

   Invoke the following API to view the complete drift for all clusters: /Services/DriftDetectionService/DriftReport

   Sample code to view the complete drift for all clusters:

```
```
def get_all_drift_jobs_report(omivv_ip, bearer_token):
    """Get all drift report jobs"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/
DriftDetectionService/DriftReport"
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        #print(json_response)
        return json_response
    else:
        print("Get all drift jobs report failed.")
```
```

   Invoke the following API to view drift for a specific cluster: /Services/DriftDetectionService/DriftReport/{id}

   Sample code to view drift for a specific cluster:

```
```
def get_drift_job_by_id(omivv_ip, bearer_token, d_job_id):
    """ Get the drift job report by job id"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/
DriftDetectionService/Jobs/" + d_job_id;
    print("url: " + url)
```
```

```
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        print(json_response)
        return json_response
    else:
        print("Get the drift job report by job id.")
```

Invoke the following API to get only driver drift for a specific cluster: /Services/DriftDetectionService/DriftReport/{id}/DriverDriftDetails

Sample code to get only driver drift for a specific cluster:

```
d_job_id_href = It is the href for drift job
i.e "https://omivvIP/Spectre/api/rest/v1/Services/DriftDetectionService/DriftReport/
domain-c47"

def get_driver_drift_report_detail_by_id(omivv_ip, bearer_token, d_job_id_href):
    """ Get the drift job details"""
    print("get Driver report")
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(d_job_id_href + "/DriverDriftDetails", verify=False,
headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get all drift jobs failed.")
```

Invoke the following API to get only hardware drift for a specific cluster: /Services/DriftDetectionService/DriftReport/{id}/ConfigurationDriftDetails

Sample code to get only hardware drift for a specific cluster:

```

d_job_id_href = It is the href for drift job
i.e "https://omivvIP/Spectre/api/rest/v1/Services/DriftDetectionService/DriftReport/
domain-c47"


def get_drift_report_configuration_detail_by_id(omivv_ip, bearer_token,
d_job_id_href):
    "" Get the drift job details"""
    print("get Configuration report")
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(d_job_id_href + "/ConfigurationDriftDetails",
verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get configuration failed..")
```

Invoke the following API to get only firmware drift for a specific cluster: /Services/DriftDetectionService/DriftReport/{id}/FwDriftDetails

Sample code to get only firmware drift for a specific cluster:

```

d_job_id_href = It is the href for drift job
i.e "https://omivvIP/Spectre/api/rest/v1/Services/DriftDetectionService/DriftReport/
domain-c47"
```

```
def get_drift_report_fwDriftDetails_detail_by_id(omivv_ip, bearer_token,
d_job_id_href):
    """ Get the drift job details"""
    print("get Configuration report")
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(d_job_id_href + "/FwDriftDetails", verify=False,
headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get fw report failed.")
```

Next steps

Update firmware for drifted components

The intention is to maintain the desired state for the cluster. For the drifted components in the drift report, trigger firmware update using the use case described earlier. For sample code, see https://github.com/dell/omivv.

Monitor component health in OMIVV

Prerequisites

Ensure that:

- Hosts added in the vCenter
- Host Credential Profile is created using OMIVV UI and inventory is run successfully

About this task

OMIVV manages ESXi host in vCenter server and calculates status of different sub systems of the server. This topic helps you to check the health of various server components.

Steps

1. Start an OMIVV session using OMIVV user credentials. OMIVV credentials are required to authenticate a client of the RESTFul API.

   For more information, see OMIVV authentication on page 9.

2. View component health.

   Invoke the following API to retrieve different server component health: `Services/InventoryService/Hosts/{host-id}/SubSystemHealth`

   Invoke the following API to retrieve host ID: `Services/ConsoleService/Consoles/{console-id}`

   Invoke the following API to retrieve Console id: `/Services/ConsoleService/Consoles`

```
```
detail_str -> Only below 4 strings is supported
            1.SubSystemHealth
            2.HostOverview
            3.FirmwareInventory
            4.Warranty

e.g def get_host_details(omivv_ip, bearer_token, host_id, 'SubSystemHealth')

def get_host_details(omivv_ip, bearer_token, host_id, details_str):
    """ Get the Host details value"""
    url = "https://" + omivv_ip + "/Spectre/api/rest/v1/Services/InventoryService/
Hosts/" + host_id + "/" + details_str
    head = {'Authorization': 'Bearer ' + bearer_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        print(json_response)
```
```

```
    else:
        print("Get host details failed.")
```

Next steps

View different health status of the components that is Healthy, Warning, Critical and take corrective actions if necessary.

Retrieve storage subsystem data in OMIVV

Prerequisites

Ensure that:

- Hosts added in the vCenter
- Host credential profile is created using OMIVV UI and inventory is run successfully

About this task

OMIVV manages ESXi host in vCenter server and provides a single pane of view for storage subsystem. This article helps to get the hierarchy of different storage component present in the ESXi host.

Steps

1. Log in to OMIVV. Start an OMIVV session using OMIVV user credentials. OMIVV credentials are required to authenticate a client of the RESTful API.

   For more information, see the Authentication and Authorization article.

2. Set the vCenter context. After you log in to OMIVV, select or identify a vCenter on which you want to perform operation. This is done by setting an operational context.

   For more information, see OMIVV authentication on page 9.

   Before setting the vCenter Context, get the list of registered vCenters.

   Sample code to get the list of registered vCenters:

   ```json
   def getRegisteredVcenterList(omivvIP,bearerToken):
       url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
   Consoles"
       head = {'Authorization': 'Bearer ' + bearerToken}
       jsonReponse = requests.get(url, verify=False, headers=head)
       if(jsonReponse.status_code == 200):
           return Response
       else:
           print("Get registered vCenter list failed")
   ```

   After you get the list of registered vCenters, set the vCenter context.

   Sample code to set the vCenter context:

   ```json
   def setOperationalContext(omivvIP, bearerToken, vcenterId,
   vcenterUsername,vCenterDomain, vCenterPassword):
       url ="https://" + omivvIP + "/Spectre/api/rest/v1/Services/ConsoleService/
   OperationalContext"
       postBodyData={"consoleId" : vcenterId, "consoleUserCredential":
   {"username":vcenterUsername,"domain" : vCenterDomain,"password" : vCenterPassword}}
       head = {'Authorization': 'Bearer ' + bearerToken}
       jsonReponse = requests.post(url, json=postBodyData, verify=False,headers=head)
       if(jsonReponse.status_code == 204):
           return pass;
       else:
           print("setOperationalContext failed.")
   ```

```
    pass
```

3. Get host ID.

   Host ID for particular host can be searched with the help of Service Tag of server.

```
def get_vcenter_tree_from_omivv(OMIVV_ip, vcenter_ip, bearerToken,host_service_Tag):
    url ="https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/ConsoleService/
Consoles"
    head = {'Authorization': 'Bearer ' + bearerToken }
    response = requests.get(url, verify=False, headers=head)
    if(response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        registered_vc_array = json.loads(json_response)
        for vc in registered_vc_array:
            if vcenter_ip.upper() == vc['ip'].upper():
                vcenter_id = vc['id']
                url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/
ConsoleService/Consoles/" + vcenter_id + "/Hosts"
                head = {'Authorization': 'Bearer ' + session_token}

response = requests.get(url, verify=False, headers=head)
                if (response.status_code == 200):
                    allHost = response.json()
                    for host in allHost:
                        if (host["serviceTag"] == host_service_Tag):
                            url = "https://" + OMIVV_ip + "/Spectre/api/rest/v1/
Services/ConsoleService/Consoles/" + vcenter_id + "/Hosts/" + host['id']
                            head = {'Authorization': 'Bearer ' + session_token}
                            response = requests.get(url, verify=False, headers=head)
                            if (response.status_code == 200):
                                json_response = json.dumps(response.json(), indent=4,
sort_keys=True)
                                print(json_response)
                                return json_response
```

4. Call storage service.

```
def get_host_storage_detail_from_omivv(OMIVV_ip,bearer_token ,host_id):
    url ="https://" + OMIVV_ip + "/Spectre/api/rest/v1/Services/InventoryService/
Hosts/"+host_id+"/Storage"
    head = {'Authorization': 'Bearer ' + session_token}
    response = requests.get(url, verify=False, headers=head)
    if (response.status_code == 200):
        json_response = json.dumps(response.json(), indent=4, sort_keys=True)
        return json_response
    else:
        print("Get storage detail failed for host")
```

Next steps:

a. User can view different storage components and establish the mapping.

b. Storage response is organized in following hierarchy:

- Storage response is a collection of Controller Object
- Each Controller consists of following sub sub-components:
  - Connector
  - Virtual Disk (RAID, and physical disk attributes can be found here)
  - Controller Information
- The mapping between Physical Disk with corresponding Virtual Disk is listed.
- Each connector contains following sub-components:
  - Enclosure Object
  - Physical Disk
  - Connector Information
- Each enclosure consists of following sub-components:
  - Physical Disk
  - Enclosure Information

5

Session management

Topics:

- Start an OMIVV session
- End an OMIVV session

Start an OMIVV session

Description: Starts an OMIVV session for the given OMIVV IP. The OMIVV session is valid for 60 minutes. Only the admin user is allowed to successfully log in as an API user.

The bearer token received from this API must be used for other APIs as a header parameter.

Command or URL: `/Services/AuthenticationService/login`

Method: POST

Request body:

```
{
        "apiUserCredential" : {
            "username":"{OMIV username (admin)}",
            "domain" : "",
            "password" : "{OMIVV password}"
        }
    }
```

For more information about request body parameters, see Request body on page 98.

Parameters: None

vCenter Privileges required: None

HTTP response code:

Table 2. HTTP response codes

Code	Description or response object
200	OK
400	Invalid parameters
429	Too many requests
500	Internal Server error / timeout
503	Client limit exhausted.

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "accessToken": "{token ID}",
    "tokenType": "{token type}",
    "expiresAt": "{token expire date and time}"
}
```

For more information, see Response body on page 102.

End an OMIVV session

Description: Logs out from the OMIVV session. This API internally invalidate the bearer token.

Command or URL: `/Services/AuthenticationService/logoff`

Method: POST

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: None

HTTP response code:

Table 3. HTTP response codes

Code	Description or response object
200	Logged off successfully
401	Authorization failure
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response: `Logged off successfully.`

6

License management

Topics:

Get list of vCenter and host license

Description: Gets list of all licenses uploaded in OMIVV.

Command or URL: `/Services/LicenseService/Licenses`

Method: GET

Parameters: None

Authorization: Bearer authentication

vCenter privileges required: None

HTTP response code:

Table 4. HTTP response code

Code	Description or response object
200	License Info
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "id": "{License ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/LicenseService/Licenses/
{LicenseID}",
        "objectType": "LicenseInfo",
        "entitlementID": "{Entitlement ID}",
        "licenseType": "{License Type}",
        "maxHosts": {number of maximum hosts allowed},
        "startDate": "{License start date}",
        "expirationDate": "{License expiry date}",
        "duration": {Duration of license},
        "licenseStatus": "{License status}"
    },
    ]
```

NOTE: After a standard license is uploaded, the evaluation license status is displayed as "inactive".

For more information, see Response body on page 102.

Get vCenter and hosts license details

Description: Gets license details using license ID.

Command or URL: `/Services/LicenseService/Licenses/{id}`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: None

Parameters:

Table 5. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	License ID. Use the license ID received from the Get list of vCenter and host license on page 38 API.	N/A	Path	String

HTTP response code:

Table 6. HTTP response code

Code	Description or response object
200	License Info
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
        "id": "{License ID}",
        "href": "https://{OMIVV IP}/Spectre/api/rest/v1/Services/LicenseService/Licenses/
{LicenseID}",
        "objectType": "LicenseInfo",
        "entitlementID": "{Entitlement ID}",
        "licenseType": "{License Type}",
        "maxHosts": {number of maximum hosts allowed},
        "startDate": "{License start date}",
        "expirationDate": "{License expiry date}",
        "duration": {Duration of the license},
        "licenseStatus": "{License status}"
 }
```

For more information, see Response body on page 102.

Upload license to OMIVV

Description: Uploads new license in .XML file format to OMIVV. You can upload one license file at a time.

After you upload the standard license, the existing evaluation license becomes inactive.

Command or URL: `/Services/LicenseService/Licenses`

Method: POST

Request Body:

```
{
  "sharetype": "{share type}",
  "path": "{license file path}",
  "credential": {
    "username": "{username}",
    "domain": "{domain}",
    "password": "{password}"
  }
}
```

For more information about request body parameters, see Request body on page 98.

Parameters:

For NFS, credential is not required.

For CIFS credentials are required. domain parameter is not mandatory.

vCenter privileges required: None

HTTP response code:

Table 7. HTTP response code

Code	Description or response object
200	Created
204	No content
401	Authorization failure
424	Failed Dependency
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
        "id": "License ID",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/LicenseService/Licenses/
{LicenseID}",
        "objectType": "LicenseInfo",
        "entitlementID": "{Entitlement ID}",
        "licenseType": "{License Type}",
        "maxHosts": {number of maximum hosts allowed},
        "startDate": "{License start date}",
        "expirationDate": "{License expiry date}",
        "duration": {Duration of license},
        "licenseStatus": "{License status}"
}
```

7

Console management

Topics:

Get list of registered vCenters

Description: Gets the list of all the registered vCenters for the given OMIVV.

Command or URL: `/Services/ConsoleService/Consoles`

Method: GET

Parameters: None

Authorization: Bearer authentication

vCenter privileges required: None

HTTP response code:

Table 8. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "id": "{vCenter ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/ConsoleService/Consoles/
{vCenter ID}",
        "objectType": "ConsoleMetadata",
        "hostname": "{vCenter hostname or FQDN}",
        "ip": "{vCenter IP}"
    }
]
```

For more information, see Response body on page 102.

Get list of all hosts in vCenter

Description: Gets the list of all the hosts registered in a vCenter for the given OMIVV IP. All Dell EMC and OEM managed hosts are listed.

Command or URL: `/Services/ConsoleService/Consoles/{id}/Hosts?serviceTags={service Tags 1}&serviceTags={service Tags 2}`

Method: GET

Optional query Parameter: serviceTag

Authorization: Bearer authentication

vCenter privileges required: None

HTTP response code:

Table 9. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
  {
    "id": "{Host ID}",
    "href": "string",
    "objectType": "string",
    "serviceTag": "{Service Tag of host}"
  }
]
```

Example response using query parameter to get details of only particular host using serviceTags:

```
[
    {
        "id": "{Host ID}",
        "href": "https://{OMIVV IP}/Spectre/api/rest/v1/Services/ConsoleService/
Consoles/33/Hosts/{Host ID}",
        "objectType": "Host",
        "serviceTag": "{Service Tag of host}"
    }
]
```

For more information, see Response body on page 102.

Get host details

Description: Gets details of a particular host in a vCenter. Use host ID as a input parameter.

Command or URL: `/Services/ConsoleService/Consoles/{id}/Hosts/{hostid}`

Method: GET

Parameters: None

Authorization: Bearer authentication

vCenter privileges required: None

HTTP response code:

Table 10. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
  "id": "{Host ID}",
  "href": "string",
  "objectType": "string",
  "hostip": "{Host IP}",
  "hostName": "{Hostname}",
  "managementIP": "{iDRAC IP}",
  "serviceTag": "{Service Tag}",
  "model": "{Server model name}",
  "systemId": "{System ID}"
}
```

For more information, see Response body on page 102.

Get vCenter tree view of data center

Description: Gets the vCenter tree view of all the data centers, clusters, and hosts of the given vCenter ID.

This API helps you to get the host and cluster details for operations like firmware update.

All Dell EMC and OEM managed or unmanaged hosts are listed.

Command or URL: `/Services/ConsoleService/Consoles/{id}`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 11. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Resource ID. Use the vCenter ID received from the **Get list of registered vCenters** API. For more information, see Get list of registered vCenters on page 41.	N/A	Path	String

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

vCenter privileges required: None

HTTP response code:

Table 12. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
{
     "id": "{vCenter ID}",
     "href": "https://{OMIVV IP}/Spectre/api/rest/v1/Services/ConsoleService/Consoles/
{vCenterID}",
     "objectType": "Console",
     "ip": "{vCenter IP}",
     "hostname": "{vCenter hostname or FQDN}",
     "registeredWithVlcm": "{vLCM registration status (true or false)},
    "datacenters": [
          {
             "id": "{Datacenter ID}",
             "href": "",
             "objectType": "Datacenter",
             "name": "{Datacenter name}",
             "clusters": [
                {
                   "id": "{cluster ID}",
                   "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
ConsoleService/Clusters/{clusterID}",
                   "objectType": "ClusterMetadata",
                   "name": "{cluster name}"
                },
                {
                   "id": "{cluster id}",
                    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
ConsoleService/Clusters/{clusterID}",
                     "objectType": "ClusterMetadata",
                      "name": "{Cluster name}"
                }
                ],
    "hosts": ": [
          {
             "id": "{Host ID}",
              "href": "",
              "objectType": "Host",
              "hostip": "{Host IP}",
              "hostName": "{Hostname or FQDN}",
              "managementIP": "{iDRAC IP}",
              "serviceTag": "{Host Service Tag}",
              "model": "{Server Model Name}",
              "systemId": "{System ID}"
          }
 ]
 }
,
```

For more information, see Response body on page 102.

Set vCenter context

Description: Sets the vCenter context on which user want to perform any operation.

Enter the vCenter ID received from `/Services/ConsoleService/Consoles` API.

The required vCenter user privilege for all other APIs are verified when you trigger the API.

Command or URL: `/Services/ConsoleService/OperationalContext`

Method: POST

Request Body:

```
{
            "consoleId": "{vCenterID}",
            "consoleUserCredential":
         {
                "username": "{vCenter username}",
                "domain": "{domain}",
                "password": "{vCenter password}"
            }
        }
```

For more information about request body parameters, see Request body on page 98.

Parameters: None

vCenter privileges required: None

HTTP response code:

Table 13. HTTP response code

Code	Description or response object
204	OK
400	Invalid parameters
401	Authorization failure
403	Failed to login to vCenter
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response: No content

Get cluster details

Description: Gets the cluster details of the given cluster ID. You can view all the hosts belong to the cluster. The vSAN and vSphere cluster details are displayed.

Command or URL: `/Services/ConsoleService/Clusters/{cluster-id}`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 14. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Resource ID. Use the cluster ID received from the **Get vCenter tree view of datacenter** API. For more information, see Get vCenter tree view of data center on page 43.	N/A	Path	String

vCenter privileges required: None

HTTP response code:

Table 15. HTTP response code

Code	Description or response object
200	OK
400	Operational context not set
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example response:

```
{
      "id": "{Cluster ID}",
      "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/ConsoleService/Clusters/
{cluster ID}",
      "objectType": "Cluster",
      "name": "{Cluster name}",
      "consoleID": {Console ID},
      "clusterType": "{Cluster Type (vSAN or vSphere)}",
      "hosts": [
         {
            "id": "{Host ID}",
            "href": "",
            "objectType": "Host",
            "hostip": "{host IP}",
            "hostName": "{Hostname}",
            "managementIP": "{iDRAC IP}",
            "serviceTag": "{Service Tag}",
            "model": "{Server model name}",
            "systemId": "{System ID}"
         }
    ]
}
```

For more information, see Response body on page 102.

Get cluster health

Description: Gets the health (DRS and vSAN cluster health) of the given cluster ID. The health data is used for assessing the firmware update.

The data that is displayed here is real-time health information.

Command or URL: `/Services/ConsoleService/Clusters/{cluster_id}/Details`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 16. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Resource ID. Use the cluster ID received from the **Get vCenter tree view of datacenter** API. For more information, see Get vCenter tree view of data center on page 43.	N/A	Path	String

vCenter privileges required: None

HTTP response code:

Table 17. HTTP response code

Code	Description or response object
200	OK
400	Operational context is not set.
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "clusterId": "{cluster ID}",
    "drsState": "{DRS status}",
    "vsanObjectHealth": "{vSAN object health status}"
}
```

For more information, see Response body on page 102.

8

Repository profile management

Topics:

Get list of repository profiles

Description: Gets the list of all repository profiles that are created in OMIVV UI.

Command or URL: `/Services/PluginProfileService/RepositoryProfiles`

Method: GET

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 18. HTTP response code

Code	Description or response object
200	OK
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
     {
        "id": "{Repository profile ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
RepositoryProfiles/{vCenterID}",
        "objectType": "BaseProfileMetadata",
        "profileName": "{Repository profile name}",
        "description": "{Profile description}"
},
]
```

For more information, see Response body on page 102.

Get repository profile details

Description: Gets the details of the given repository profile ID. You can view the firmware and driver repository details. Also, you can view factory-created and user-created repository details.

Command or URL: `/Services/PluginProfileService/RepositoryProfiles/{id}`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 19. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Resource ID. Use the repository profile ID received from the **Get list of repository profiles** API. For more information, see Get list of repository profiles on page 48	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 20. HTTP response code

Code	Description or response object
200	OK
400	Operational Context is not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{Repository profile ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
RepositoryProfiles/{Repository profile ID}",
    "objectType": "RepositoryProfile",
    "data":
       {
          "name": "{Profile Name}",
          "description": "{Repository Profile description}",
          "globalDefault": {Dell default repository profile},
          "repoType": "{Repository type}",
          "protocolType": "{Protocol type}",
          "uri": "{Catalog location}",
          "credential":
       {
          "username": "{Repository profile username}",
          "domain": "{domain name}",
          "password": "{Profile password}"
        },
           "synchronizeRepository": {Synchronize with current repository location}
 }
}
```

For more information, see Response body on page 102.

Create a new repository profile

Description: Creates a new repository profile

Command or URL: `/Services/PluginProfileService/RepositoryProfiles`

Method: POST

Request Body:

```
{
  "name": "{profile name}",
  "description": "{profile description}",
  "globalDefault":{Dell default repository profile(false), currently not supported},
  "repoType": "{repository type}",
  "protocolType": "{protocol type}",
  "uri": "{Catalog location}",
  "credential": {
    "username": "{share username}",
    "domain": "{domain}",
    "password": "{share password}"
  },
  "synchronizeRepository": {Synchronize with current repository location (false)}
}
```

For more information about request body parameters, see Request body on page 98.

For NFS, credential is not required.

For CIFS credentials are required. domain parameter is not mandatory.

Parameters: None

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 21. HTTP response code

Code	Description or response object
200	Repository Profile Data
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{repository profile ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
RepositoryProfiles/{repository profile ID}",
    "objectType": "RepositoryProfile",
    "data": {
        "name": "{Repository profile name}",
        "description": "{Repository profile description}",
        "globalDefault": {Dell default repository profile (true or false)},
        "repoType": "{Repository Type}",
        "protocolType": "{Protocol type}",
        "uri": "{Repository location}",
        "credential": {
            "username": "{share username}",
            "domain": "{domain}",
            "password": "{}"
        },
```

```
        "synchronizeRepository": {Synchronize with current repository location(true)}
    }
}
```

Modify or edit repository profile

Description: Modifies repository profile

Command or URL: `/Services/PluginProfileService/RepositoryProfiles`

Method: PUT

Request Body:

```
{
  "name": "{profile name}",
  "description": "{profile description}",
  "repoType": "{Repository type}",
  "protocolType": "{Protocol type}",
  "uri": "{catalog location}",
  "credential": {
    "username": "{share username}",
    "domain": "{domain}",
    "password": "{share password}"
  },
  "synchronizeRepository": {Synchronize with current repository location}
}
```

For more information about request body parameters, see Request body on page 98.

For NFS, credential is not required.

For CIFS credentials are required. domain parameter is not mandatory.

Parameters: None

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 22. HTTP response code

Code	Description or response object
200	Repository Profile Data
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{profile ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
RepositoryProfiles/{profile ID}",
    "objectType": "RepositoryProfile",
    "data": {
        "name": "{profile name}",
        "description": "{Profile description}",
        "globalDefault": {Dell default repository profile},
        "repoType": "{Repository type}",
        "protocolType": "{Protocol type}",
        "uri": "{Catalog location}",
```

```
        "credential": {
            "username": "{share username}",
            "domain": "{domain}",
            "password": "{}"
        },
        "synchronizeRepository": {Synchronize with current repository location}
    }
}
```

9

Cluster profile management

Topics:

Get list of cluster profiles

Description: Gets the list of all cluster profiles that are created in OMIVV UI.

Command or URL: `/Services/PluginProfileService/ClusterProfiles`

Method: GET

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 23. HTTP response code

Code	Description or response object
200	OK
400	Operation context is not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "id": "{Cluster Profile ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
ClusterProfiles/{Cluster profile ID}",
        "objectType": "BaseProfileMetadata",
        "profileName": "{Cluster profile name}",
        "description": "{Cluster profile description}"
    }
]
```

For more information, see Response body on page 102.

Get details of cluster profiles

Description: Gets the details of a given cluster profile. You can view the associated system profile, driver repository profile, firmware repository profile, and associated cluster details.

Command or URL: `/Services/PluginProfileService/ClusterProfiles/{id}`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 24. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Cluster profile ID. Use the cluster profile ID received from the **Get list of cluster profiles** API. For more information, see Get list of cluster profiles on page 53	N/A	Path	String

vCenter Privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 25. HTTP response code

Code	Description or response object
200	OK
400	Operational Context not set
401	Authorization failure
403	VCenter permission denied
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{Cluster profie ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
ClusterProfiles/{Cluster profile ID}",
    "objectType": "ClusterProfile",
    "data": {
        "profileName": "{Cluster profile name}",
        "description": "{Cluster profile description}",
        "clusters": [
            {
                "id": "{Cluster ID}",
                "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/ConsoleService/
Clusters/{Cluster ID}",
                "objectType": "ClusterMetadata",
                "name": "{Cluster name}"
            }
        ],
        "repo": [
            {
                "id": "{Repostory profile ID}",
                "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
PluginProfileService/RepositoryProfiles/{Repository profile ID}",
                "objectType": "BaseProfileMetadata",
                "profileName": "{Repository profile name}",
                "repoType": "{Repository profile type (Firmware or Driver}",
                "description": "{Profile description}"
            }
        ],
        "systemProfile": {
            "id": "{System profileID}",
            "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
RepositoryProfiles/{System profile ID}",",
```

```
            "objectType": "BaseProfileMetadata",
            "profileName": "{System profile name}",
            "description": "{System profile description}"
        }
    },
    "driftJob": {
        "id": "{Drift detection job ID}",
        "href": "https://{OMIVV IP}/Spectre/api/rest/v1/Services/DriftDetectionService/
Jobs/{drift detection job ID}",
        "objectType": "JOB",
        "status": "{Status of drift detection job}"
    }

}
```

For more information, see Response body on page 102.

10

Firmware repository inventory management

Topics:

- Get firmware repository inventory details

Get firmware repository inventory details

Description: Gets the details of firmware repository inventory. Ensure that the repository is successfully downloaded.

The driver repository inventory is not supported.

You can view the details like bundle ID, list of components of the particular bundle ID for the specific server model.

Command or URL: `/Services/RepositoryManagementService/RepositoryData?repoProfileID={repoProfileID}&bundleId={bundleId} &systemId={systemId}`

Mandatory query parameter: repoProfileID

Optional query parameter: bundleId, systemId

Method: GET

Authorization: Bearer authentication

Parameters:

Table 26. Parameters

Parameter	Value	Description	Parameter type	Data type
repoProfileID	(required)	Related firmware repository profile ID. Use the repository profile ID received from the **Get list of repository profiles** API. For more information, see Get list of repository profiles on page 48	query	String
systemId	Optional	SystemID of the server to filter the bundles. To get the system ID, run `/Services/ConsoleService/Consoles/{id}` or `/Services/ConsoleService/Clusters/{cluster-id}`. For more information, see Get vCenter tree view of data center on page 43 and Get cluster details on page 45.	query	String
bundleId	Optional	Bundle ID to retrieve software components. To get the bundle ID, run `/Services/RepositoryManagementService/RepositoryData?repoProfileID={repoProfileID}&bundleId={bundleId}`. It is mandatory to enter bundleId to get the component list.	query	String

vCenter privileges required: Dell.Configuration.Firmware Update

HTTP response code:

Table 27. HTTP response code

Code	Description or response object
200	OK
400	Invalid parameters
401	Authorization failure
403	Failed to log in to vCenter

Table 27. HTTP response code (continued)

Code	Description or response object
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "repoProfileID": "{Repository profile ID}",
     "bundleID": "{Bundle ID}",
     "bundles": [
        {
          "bundleID": "{Bundle ID}",
          "name": "{Bundle name}",
          "description": "{Bundle description}",
          "model": "{Server model name}",
          "targetOS": "{Target host OS name}",
          "systemId": "{Server unique ID}"
         }
 ],
"components": [
   {
      "packageID": "{Component unique package ID}",
      "rebootRequired":"{host reboot required (True or False)}",
      "releaseDate": "{Component release date and month}",
      "name": "{Component name}",
      "description": "{Component description}",
      "supportedHWComponents": [
          {
             "componentID": "{Component ID}",
             "pcieVariants": [
                {
                   "deviceID": "{Device ID}",
                   "subDeviceID": "{Sub device ID}",
                   "subVendorID": "{Sub vendor ID}",
                   "vendorID": "{Vendor ID}"
                 }
              ]
           },
        ]
   }],
            "revisionHistory": "{Revison history}",
            "vendorVersion": "{Vendor Version}",
            "criticality": "{importance of component update}"
        },
```

NOTE: After you upgrade OMIVV to 5.2 (RPM Upgrade or backup and restore), the value of model, revisionHistory, and description are displayed as null until firmware repository is refreshed. To refresh the repository profile, on the repository profile page, click **Edit** and complete the wizard without doing any modification.

For more information, see Response body on page 102.

11

Firmware inventory management

Topics:

Create host level firmware inventory report

Description: Creates the host level firmware inventory report. You can view the host component details that is associated to the given bundle ID before performing firmware update.

Ensure that the host is managed by OMIVV and management complaint.

Use the bundle ID received from the `/Services/RepositoryManagementService/RepositoryData?repoProfileID={repoProfileID}&systemId={systemId}&bundleId={bundleId}` API.

Use the system ID received from the `/Services/ConsoleService/Clusters/{cluster-id}` API.

For hosts that are managed at data center level, use the system ID received from the /Services/ConsoleService/Consoles/{id} API.

Command or URL: `/Services/UpdateService/FWReport`

Method: POST

Authorization: Bearer authentication

Request Body:

```
{
   "reportType":"HOST",
  "hostID":"{host ID}",
  "repoProfileID":"{Repository profile ID}"
   "bundleAssociation":[
      {
           "systemId":"{system ID}",
         "bundleID":"{Bundle ID}"
      }
   ],

 }
```

For more information about request body parameters, see Request body on page 98.

Parameters: None

vCenter privileges required: Dell.Configuration.Firmware Update

HTTP response code:

Table 28. HTTP response code

Code	Description or response object
200	OK
400	Invalid parameters
401	Authorization failure
403	Failed to login to vCenter
429	Too many requests

Table 28. HTTP response code (continued)

Code	Description or response object
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
   "reportType": "HOST",
    "clusterReport":NULL,
    "hostReports":
                {
                    "host": {
                        "id": "{Host ID}",
                         "href": "",
                         "objectType": "Host",
                          "hostip": "{Host IPv4 or FQDN}",
                          "hostName": "",
                          "managementIP": "{iDRAC IP}",
                          "serviceTag": "{Host Service Tag}",
                          "model": "{Server model name}",
                          "systemId": "{System ID}"
                 },
                 "applicableComponents": [
                     {
                        "componentType": "{Component Type}",
                        "packageId": "{Package ID}",
                        "component": "{Component Name}",
                        "currentVersion": "{Current installed componet verson}",
                        "availableVersion": "{Available component version}",
                        "criticality": "{Importance of component update}",
                        "updateAction": "{Component update status}",
                        "scheduled": {Component update job scheduled},
                        "rebootRequired": {Host reboot required (True or False)},
                        "releaseDate": "{Component release date and month}"
                       },
            ]
  }
}
```

For more information, see Response body on page 102.

Create cluster level firmware inventory report

Description: Creates the cluster-level firmware inventory report. You can create vSAN and vSphere cluster level firmware inventory reports.

Use the bundle ID received from the `/Services/RepositoryManagementService/RepositoryData?repoProfileID={repoProfileID}&systemId={systemId}&bundleId={bundleId}` API.

Use the system ID received from the `/Services/ConsoleService/Clusters/{cluster-id}` API.

Command or URL: `/Services/UpdateService/FWReport`

Method: POST

Authorization: Bearer authentication

Request Body:

```
 {
   "reportType":"{CLUSTER}",
   "clusterID":"{Cluster ID}",

   "bundleAssociation":[
      {
         "systemId":"{System ID}",
```

```
            "bundleID":"{Bundle ID}"
        },
        {
            "systemId":"{System ID}",
            "bundleID":"{Bundle ID}"
        }


    ],
    "repoProfileID":"{Repository profile ID}"
}
```

For more information about request body parameters, see Request body on page 98.

Parameters: None

vCenter privileges required: Dell.Configuration.Firmware Update

HTTP response code:

Table 29. HTTP response code

Code	Description or response object
200	OK
400	Invalid parameters
401	Authorization failure
403	Failed to login to vCenter
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
     "reportType": "{CLUSTER}",
     "clusterReport": {
         "cluster": {
            "id": "{Cluster ID}",
            "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/ConsoleService/
Clusters/{Cluster ID}",
            "objectType": "clusterMetadata",
            "name": "{Cluster name}"
          },
         "hostReports": [
            {
              "host": {
                 "id": "{Host ID}",
                  "href": "",
                  "objectType": "Host",
                  "hostip": "{Host IP}",
                  "hostName": "",
                  "managementIP": "{iDRAC IP}",
                  "serviceTag": "{Host Service Tag}",
                  "model": "{Server model name}",
                  "systemId": "{System ID}"
           },
          "applicableComponents": [
              {
                 "componentType": "{Componenet Type}",
                 "packageId": "{Package ID}",
                 "component": "{Component Name}",
                 "currentVersion": "{Currently installed component version}",
                 "availableVersion": "{Available component version}",
                 "criticality": "{Importance of component update}",
                 "updateAction": "{Component update status}",
                 "scheduled": {Component update job scheduled},
```

```
                    "rebootRequired": {Host reboot required},
                    "releaseDate": "{Component release date and month}"
                },
            ]
        }
    ]
  },
   "hostReport": null
}
```

For more information, see Response body on page 102.

12

Firmware update management

Topics:

Create host level firmware update jobs

Description: Creates the firmware update job for a host managed by OMIVV. You can update both vSAN and vSphere host.

Driver update is not supported.

Chassis and single DUP firmware updates are not supported.

It may take few seconds to create firmware update job for large number of hosts.

Command or URL: `Services/UpdateService/Jobs`

Method: POST

Request body:

```
{
   "jobname":"{Firmware update job name}",
   "jobdesc":"{Job description}",
   "updateType":"{Update type (FIRMWARE)}",
   "updateTargetType":"HOST",
   "schedule":{
              "runLater":{Scheduled to run at a specified time (true or false)},
              "dateTime":"{Firmware update job schedule (date and time format:YYYY-MM-
DDTHH:MM:SSZ, 24 hour UTC time}"
              "runNow":{Run firmware update job now (true or false)},

  },
   "firmwareRepoProfileID":"{Firmware repository profile ID}",
   "rebootOptions":"{Reboot options}",
   "preCheck":{Check prerequisites before upate(true or false)},
   "firmwareUpdateTargets":[
              {
                           "hostId":"{Host ID}",
                           "bundleId":"{Bundle ID}",
                           "packageIDs":["Package ID1", "Package ID2"]
              }],
      "jobSpecificCustomConfiguration":{
              "exitMaintenanceMode":{true or false},
              "migratePoweredOffAndSuspendedVMs":{true or false},
              "resetIDracAndDeleteJobs":{true or false},
              "enterMaintenanceModetimeout":60,
              "enterMaintenanceModeOption": "{Enter maintenance mode option}"

      }
}
```

For more information about request body parameters, see Request body on page 98.

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: Dell.Configuration.Firmware Update

HTTP response code:

Table 30. HTTP response code

Code	Description
202	OK
400	Operational Context not set
401	Authorization failure
403	VCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{JOB ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/UpdateService/Jobs/{job ID}",
    "objectType": "Job",
    "status": "{Job status}"
}
```

For more information, see Response body on page 102.

Create cluster level firmware update jobs

Description: Creates the firmware update job for a cluster managed by OMIVV. You can update both vSAN and vSphere clusters.

Driver update is not supported.

It may take few seconds to create the firmware update job for large number of hosts.

Command or URL: `Services/UpdateService/Jobs`

Method: POST

Request body:

```
{
   "jobname":"{job name}",
   "jobdesc":"{job description}",
   "updateType":"{Update type (firmware)}",
   "updateTargetType":"CLUSTER",
   "schedule":{
              "runLater":{Scheduled to run at a specified time (true or false),
              "dateTime":"{Firmware update job schedule (Firmware update job schedule
(date and time format:YYYY-MM-DDTHH:MM:SSZ, 24 hour UTC time)}"
              "runNow":{Run firmware update job now (true or false)},

  },
   "firmwareRepoProfileID":"{Firmware repository profile ID}",
   "rebootOptions":"{Reboot options}",
   "preCheck":{Check prerequisites before upate(true or false),
   "firmwareUpdateTargets":[
              {
                           "hostId":"{Host ID}",
                           "clusterId":"{Cluster ID, only for cluster update}",
                           "bundleId":"{Bundle ID}",
                           "packageIDs":["Package ID1", "Package ID2"]
              }],
         "jobSpecificCustomConfiguration":{
```

```
                "exitMaintenanceMode":{true or false},
                "migratePoweredOffAndSuspendedVMs":{true or false},
                "resetIDracAndDeleteJobs":{true or false},
                "enterMaintenanceModetimeout":60,
                "enterMaintenanceModeOption": "{enter maintenance mode option}"

        }
}
```

For more information about request body parameters, see Request body on page 98.

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: Dell.Configuration.Firmware Update

HTTP response code:

Table 31. HTTP response code

Code	Description or response object
202	OK
400	Operational Context not set
401	Authorization failure
403	VCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{JOB ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/UpdateService/Jobs/{job ID}",
    "objectType": "Job",
    "status": "Scheduled"
}
```

For more information, see Response body on page 102.

Get list of firmware update jobs (Host and Cluster)

Description: Gets all the host and cluster firmware update jobs.

Command or URL: `/Services/UpdateService/Jobs`

Method: GET

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: Dell.Configuration.Firmware Update

HTTP response code:

Table 32. HTTP response code

Code	Description or response object
200	OK

Table 32. HTTP response code (continued)

Code	Description or response object
400	Operational Context not set
401	Authorization failure
403	VCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "id": "{Firmware update job ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1Services/UpdateService/Jobs/
{firmware update job ID}",
        "objectType": "JOB",
        "status": "{job status}"
    }
]
```

For more information, see Response body on page 102.

Get firmware update job details (Host or Cluster)

Description: Gets the details of the given host firmware update job ID.

Command or URL: `/Services/UpdateService/Jobs/{id}`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 33. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Firmware update job ID. Use the firmware update job ID received from the **Get list of host firmware update jobs** API. For more information, see Get list of firmware update jobs (Host and Cluster) on page 64.	N/A	Path	String

vCenter privileges required: Dell.Configuration.Firmware Update

HTTP response code:

Table 34. HTTP response code

Code	Description or response object
200	OK
400	Operational Context is not set
401	Authorization failure
403	vCenter permission denied
404	Resource not found
429	Too many requests

Table 34. HTTP response code (continued)

Code	Description or response object
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{Job ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/UpdateService/Jobs/{job ID}",
    "objectType": "UpdateJob",
    "status": "{job status}",
    "data": {
        "jobname": "{Job name}",
        "jobdesc": "{Job description}",
        "updateType": "{Update type}",
        "schedule": {
            "runNow": false,
            "runLater": true,
            "dateTime": "{Update job schedule date and time}"
        },
        "repoProfileID": "{Repository profile ID}",
        "associatedEntities": [
            {
                "host": {
                    "id": "{Host ID}",
                    "href": "",
                    "objectType": "Host",
                    "hostip": "{Host IP}",
                    "hostName": "{Hostname}",
                    "managementIP": "{iDRAC IP}",
                    "serviceTag": "{Host Service Tag}",
                    "model": "{Server model name}",
                    "systemId": "{System ID}"
                },
                "bundleId": null,
                "packageIDs": [
                    "{Package ID1}",
                    "{Package ID2}"
                ]
            }
        ],
        "rebootOptions": "{reboot options}",
         "jobSpecificCustomConfiguration": {
            "exitMaintenanceMode":{true or false},
              "migratePoweredOffAndSuspendedVMs":{true or false},
              "resetIDracAndDeleteJobs":{true or false},
              "enterMaintenanceModetimeout":60,
              "enterMaintenanceModeOption": "{Enter maintenance mode option}"
        }
    }
}
```

For more information, see Response body on page 102.

Delete firmware update job

Description: Deletes specific firmware update job. You can delete only one job at a time.

This API enables you to delete only Successful, Failed, and Canceled firmware update jobs.

Command or URL: `/Services/UpdateService/Jobs/{id}`

Method: DELETE

Authorization: Bearer authentication

vCenter privileges required: Dell.Configuration.Firmware Update

Table 35. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Firmware update job ID. Use the firmware update job ID received from the **Get list of host firmware update jobs** API. For more information, see Get list of firmware update jobs (Host and Cluster) on page 64.	N/A	Path	String

HTTP response code:

Table 36. HTTP response code

Code	Description or response object
200	Deleted
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
Deleted
```

For more information, see Response body on page 102.

Cancel firmware update job

Description: Cancels scheduled and in progress firmware update jobs.

If you stop a firmware update job that is already submitted to iDRAC, the firmware might still get updated on the host. OMIVV reports the job as canceled.

If the job is in scheduled state in OMIVV, job is not submitted to iDRAC. It will get submitted in iDRAC only when it is submitted in OMIVV.

Command or URL: `/Services/UpdateService/Jobs/{id}/Cancel`

Method: PUT

Authorization: Bearer authentication

vCenter privileges required: Dell.Configuration.Firmware Update

Table 37. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Firmware update job ID. Use the firmware update job ID received from the **Get list of host firmware update jobs** API. For more information, see Get list of firmware update jobs (Host and Cluster) on page 64.	N/A	Path	String

HTTP response code:

Table 38. HTTP response code

Code	Description or response object
200	Cancelled
202	Cancelling
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
The scheduled firmware update job <job ID> is cancelled.
```

For more information, see Response body on page 102.

13

System profile management

Topics:

Get list of system profiles

Description: Gets the list of all system profiles that are created in OMIVV UI.

Command or URL: `/Services/PluginProfileService/SystemProfiles`

Method: GET

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: None

HTTP response code:

Table 39. HTTP response code

Code	Description or response object
200	OK
400	Operation context is not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "id": "{System profile ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
SystemProfiles/{SystemProfileID}",
        "objectType": "SystemProfileMetadata",
        "profileName": "{Profile name}",
        "description": "{Profile description}",
        "systemProfileType": "{System profile Type (Basic or Advanced)}",
        "referenceServer": "{Reference server IP}",
        "serverModel": "{Server model name}"
    }
]
```

For more information, see Response body on page 102.

Get system profile details

Description: Gets the details of the given system profile ID.

You can use only 12G and later PowerEdge servers and bare-metal servers as a reference server.

Command or URL: `/Services/PluginProfileService/SystemProfiles`

Method: GET

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: None

HTTP response code:

Table 40. HTTP response code

Code	Description or response object
200	OK
400	Operation context is not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "id": "{system profile ID}",
    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
SystemProfiles/{systemprofileID}",
    "objectType": "SystemProfile",
    "data": {
        "name": "{profile name}",
        "description": "{profile description}",
        "systemProfileType": "{Profile Type (Basic or Advanced)}",
        "referenceESXiHostNameOrIP": "{Reference server hostname or hostIP}",
        "referenceManagementIP": "{iDRAC IP}",
        "referenceiDRACType": "{iDRAC license type}",
        "serviceTag": "{Service Tag of host}",
        "serverModel": "{Server model name}",
        "dateCreated": "{date and time when the profile is created}",
        "dateModified": "{date and time when the profile is modified}",
        "lastModifiedBy": "{Details of the user who modified the profile}"
    }
}
```

For more information, see Response body on page 102.

14

Drift management

Topics:

Get all drift detection job

Description: Gets all drift detection jobs.

Command or URL: `/Services/DriftDetectionService/Jobs`

Method: GET

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 41. HTTP response code

Code	Description or response object
200	Drift job MetaData
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "id": "{Drift job ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/DriftDetectionService/
Jobs/{drift job ID}",
        "objectType": "JOB",
        "status": "{Job status}"
    },

]
```

For more information, see Response body on page 102.

Get specified drift detection job

Description: Gets specified drift detection job.

Command or URL: `/Services/DriftDetectionService/Jobs/{id}`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 42. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Resource ID. Use the vCenter ID received from the **Get all drift detection jobs** API. For more information, see Get all drift detection job on page 71.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 43. HTTP response code

Code	Description or response object
200	Drift job MetaData
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "jobname": "{Drift detection job name}",
    "lastrun": "{Drift detection job last run}",
    "nextrun": "{Drift detection job next run}",
    "collectionSize": "0",
    "associatedClusterProfile": {
        "id": "{drift detection job ID}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
ClusterProfiles/{job ID}",
        "objectType": "ClusterProfile",
        "data": {
            "profileName": "{cluster profile name}",
            "description": "{Profile description}",
            "clusters": [
                {
                    "id": "{cluster ID}",
                    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
ConsoleService/Clusters/{cluster ID}",
                    "objectType": "ClusterMetadata",
                    "name": "{cluster name}"
                }
            ],
            "repo": [
                {
                    "id": "{repository profile ID}",
                    "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
```

```
PluginProfileService/RepositoryProfiles/{profile ID}",
                    "objectType": "RepositoryProfileMetadata",
                    "profileName": "{Repository profile name}",
                    "description": "{Profile description}",
                    "repoType": "{Repository type}"
                }
            ],
            "systemProfile": {
                "id": "{system profile name}",
                "href": "",
                "objectType": "",
                "profileName": "{system profile name}",
                "description": "{profile description}"
            }
        },
        "driftJob": {
            "id": "{drift job ID}",
            "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
DriftDetectionService/Jobs/{drift job ID}",
            "objectType": "JOB",
            "status": "{Job status}"
        }
    }
}
```

For more information, see Response body on page 102.

Get drift report for all clusters

Description: Gets the drift report of all the clusters that are non-compliant.

Command or URL: `/Services/DriftDetectionService/DriftReport`

Method: GET

Authorization: Bearer authentication

Parameters: None

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 44. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "cluster": {
            "id": "{cluster ID}",
            "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/ConsoleService/
Clusters/{cluster ID}",
            "objectType": "ClusterMetadata",
            "name": "{Cluster name}"
        },
        "clusterComplianceStatus": "{Compliance status}",
        "clusterProfileDetails": {
```

```
            "id": "{cluster profile ID}",
            "name": "{cluster profile name}",
            "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
ClusterProfiles/{cluster profile ID}"
        },
        "href": "https://{OMIVV IP}/Spectre/api/rest/v1/Services/DriftDetectionService/
DriftReport/{cluster ID}",
        "systemProfileName": "{system profile name}",
        "fmRepoProfileName": "{firmware respository profile name}",
        "driverRepoProfileName": "{driver repository profile name}"
    }
]
```

For more information, see Response body on page 102.

Get drift report for specific cluster

Description: Gets the drift report of a specific cluster ID. The drift details includes firmware, driver, and configuration details of the given cluster ID.

Command or URL: `/Services/DriftDetectionService/DriftReport/{cluster-id}`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 45. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Resource ID. Use the cluster ID retrieved from `/Services/DriftDetectionService/DriftReport` or `/Services/ConsoleService/Consoles/{id}` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 46. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
   "cluster": {
        "id": "{Cluster ID}",
         "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/ConsoleService/Clusters/
{Cluster ID}",
         "objectType": "ClusterMetadata",
         "name": "{Cluster Name}"
```

```
    },
   "clusterComplianceStatus": "{Cluster Compliance Status}",
    "clusterProfileDetails": {
         "id": "{Cluster Profile ID}",
         "name": "{Cluster Profile Name}",
         "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/PluginProfileService/
ClusterProfiles/{Cluster profile ID}"
  },
   "driftDetails": {
        "fwDriftDetails": {
              "complianceStatus": "{Firmware compliance status of the cluster}",
              "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
DriftDetectionService/DriftReport/{Cluster ID}/FwDriftDetails",
              "objectType": "FirmwareDriftDetails",
              "nonCompliantHostList": [
                    "{Host ID}"
               ],
               "notApplicableHostList": [],
                "compliantHostList": []
  },
   "driverDriftDetails": {
        "complianceStatus": "{Driver Compliance status of the cluster}",
        "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
DriftDetectionService/DriftReport/{Cluster ID}/DriverDriftDetails",
        "objectType": "DriverDriftDetails",
         "nonCompliantHostList": [],
         "notApplicableHostList": [
             "{Host ID}"
          ],
           "compliantHostList": []
     },
     "configurationDriftDetails": {
         "complianceStatus": "{Configuration compliance status of the cluster}",
          "href": "https://{OMIVVIP}/Spectre/api/rest/v1/Services/
DriftDetectionService/DriftReport/{Cluster ID}/ConfigurationDriftDetails",
          "objectType": "ConfigurationDriftDetails",
          "nonCompliantHostList": [],
          "notApplicableHostList": [
            "{Host ID}"
           ],
           "compliantHostList": []
      }
   }
}
```

For more information, see Response body on page 102.

Get firmware drift report

Description: Gets the firmware drift report for specific cluster.

This API helps you to get the drifted components details before running firmware update.

Command or URL: `/Services/DriftDetectionService/DriftReport/{id}/FwDriftDetails`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 47. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Resource ID. Use the cluster ID retrieved from `/Services/ConsoleService/Consoles/{id}`.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 48. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "host": {
            "id": "{Host ID}",
            "href": "",
            "objectType": "Host",
            "hostip": "{Host IP}",
            "hostName": "{Hostname}",
            "managementIP": "{iDRAC IP}",
            "serviceTag": "{Service tag}",
            "model": "{Server Model name}",
            "systemId": "{System ID}"
        },
        "overallSummary": {
            "compliantStatus": "{Compliance Status}",
            "nonCompliantType": "{Non-compliant Type}",
            "noncompliantTypeDescription": "{Firmware version is different for N number
of component (s)}"
        },
        "componentDriftList": [
            {
                "componentDetails": {
                    "componentID": "{Component ID}",
                    "packageID": "{Package ID}",
                    "bundleID": "{Bundle ID}",
                    "instanceID": "{Instance ID}",
                    "componentName": "{Component Name}",
                    "componentType": "{Component Type}",
                    "componentTypeDisplay": "{Component Type Display}",
                    "upgrade": "Component Update required True or False",
                    "criticality": "{Importance of component update}",
                    "rebootRequired": "Host reboot required (True or False)",
                    "pciDeviceInfo": {
                        "deviceID": "{PCI device ID}",
                        "subDeviceID": "{PCI sub device ID}",
                        "subVendorID": "{PCI sub vendor ID}",
                        "vendorID": "{PCI vendor ID}"
                    }
                },
                "complianceSummary": {
                    "compliantStatus": "{Compliance Status}",
                    "nonCompliantType": "{non-compliance type}",
                    "noncompliantTypeDescription": "{Reason for non-compliance}"
                },
                "complianceDetails": {
                    "driftedVersionInfo": "{Component Version in Host}",
                    "baselineVersionInfo": "{Component version in Repository}"
                }
            }


        ]
    }
```

```
]
```

NOTE: After you upgrade OMIVV to 5.2 (RPM Upgrade or backup and restore), the value of model, revisionHistory, and description are displayed as null until firmware repository is refreshed. To refresh the repository profile, on the repository profile page, click **Edit** and complete the wizard without doing any modification. After repository profile is updated, update the associated cluster profile. Drift detection job runs after you update the cluster profile.

For more information, see Response body on page 102.

Get driver drift report

Description: Gets the driver drift details of a specific vSAN cluster ID.

Command or URL: `/Services/DriftDetectionService/DriftReport/{id}/DriverDriftDetails`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 49. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Cluster ID. Use the cluster ID retrieved from `/Services/DriftDetectionService/DriftReport` or `/Services/ConsoleService/Consoles/{id}` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 50. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "overallSummary": {
            "compliantStatus": "{Compliance status of the cluster}",
            "nonCompliantType": "{Non-compliance Type}",
            "noncompliantTypeDescription": "{Reason for non-comliance}"
        },
        "componentDriftList": [
        {
        "complianceDetails": {
                    "driftedVersionInfo": "{Driver version in Host}",
                    "baselineVersionInfo": "{Component version in Repo}"
                            },
                "complianceSummary": {
```

```
                    "compliantStatus": "{Component compliance status}",
                    "nonCompliantType": "{Non-Compliance Type}",
                    "noncompliantTypeDescription": "{Non-Compliance Reason}"
                },
    "componentDetails": {
                    "vendorName": "{Vendor name}"
                    "componentName": "{Component Name}",
                                "driverName": "{Driver Name}",
                    "packageType": "{Package Type}",
                    "rebootRequired": "{Host reboot required (True or False)}",
                                "recommendation": "{Recommendation (Yes or No)}"

                }
        }
        ],
        "host": {
            "id": "{Host ID}",
            "href": "",
            "objectType": "Host",
            "hostip": "{Host IP}",
            "hostName": "{Hostname}",
            "managementIP": "{iDRAC IP}",
            "serviceTag": "{Service Tag}",
            "model": "{Server Model Name}",
            "systemId": "{System ID}"
        }
    }
]
```

For more information, see Response body on page 102.

Get configuration drift report

Description: Gets the configuration drift (system profile drift) details of a specific cluster ID. Ensure that system profile is created using OMIVV UI.

Command or URL: `Services/DriftDetectionService/DriftReport/{id}/ConfigurationDriftDetails`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 51. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Cluster ID. Use the cluster ID retrieved from `/Services/DriftDetectionService/DriftReport` or `/Services/ConsoleService/Consoles/{id}` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 52. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests

Table 52. HTTP response code (continued)

Code	Description or response object
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "host": {
            "id": "{Host ID}",
            "href": "",
            "objectType": "Host",
            "hostip": "{Host IP}",
            "hostName": "{hostname}",
            "managementIP": "{iDRAC IP}",
            "serviceTag": "{Service Tag}",
            "model": "{Server Model Name}",
            "systemId": "{System ID}"
        },
        "overallSummary": {
            "compliantStatus": "{Compliance Status}",
            "nonCompliantType": "{Non-compliant Type}",
            "noncompliantTypeDescription": "{Reason for non-compliance}"
        },
        "componentDriftInfo": {
            "missingComponents": [
                {
                    "componentName": "{Component name}",
                    "componentFqdd": "{Component FQDD}"
                },
                {
                    "componentName": "{Component name}",
                    "componentFqdd": "{Component FQDD}"
                }
            ],
            "mismatchedComponents": [
                {
                    "componentName": "{Component name}",
                    "componentFqdd": "{Component FQDD}",
                    "subComponents": [],
                    "containsSubComponent": false,
                    "missingGroups": [],
                    "misMatchedGroups": [
                        {
                            "name": "{Component name}",
                            "missingAttributes": [],
                            "misMatchedAttirbutes": [
                                {
                                    "attributeName": "{Attribute Name}",
                                    "attributeLctValue": "{Attribute Value}",
                                    "attributeDeviceValue": "{Attribute value configured
in server (enabled or disabled)"
                                }
                            ]
                        }
                    ]
                }
            ]
        }
    }
]
```

For more information, see Response body on page 102.

Gets host details on specified drift detection jobs

Description: Gets host details on specified drift detection jobs

Command or URL: `/Services/DriftDetectionService/Jobs/{id}/Details`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Table 53. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Drift detection job ID. Use the job ID received from `/Services/DriftDetectionService/Jobs`. For more information, see Get all drift detection job on page 71.	N/A	Path	String

HTTP response code:

Table 54. HTTP response code

Code	Description or response object
200	DriftDetectionJobHostData
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
[
    {
        "status": "{Drift detection job status}",
        "startTime": "{Drift detection job start time}",
        "endTime": "{Drift detection job end time}",
        "host": "{Host IP}",
        "serviceTag": "{Host Service Tag}",
        "managementIP": "{iDRAC IP}",
        "cluster": "{Cluster name}",
        "vcenter": "{vCenter IP}"
    }
]
```

For more information, see Response body on page 102.

15

Get subsystem health report (OMIVV Host Health)

Description: Gets the health of the server components such as Fan, Power Supply, and Memory. The information displayed here is real-time information.

Command or URL: `Services/InventoryService/Hosts/{id}/SubSystemHealth`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 55. Parameters

Parameter	Value	Description	Parameter type	Data type
id	(required)	Host ID. Use the host ID received from `/Services/ConsoleService/Consoles/{id}` or `/Services/ConsoleService/Consoles/{Console-ID}/Hosts` or `/Services/ConsoleService/Consoles/{id}/Hosts?serviceTags={service Tags 1}& serviceTags={service Tags 2}`	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 56. HTTP response code

Code	Description or response object
200	OK
401	Authorization failure
404	Resource not found
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "overallHealth": "{{Overall health of a server}}",
    "subSystemHealth": [
        {
            "subSystem": "{Fan}",
            "rollUpStatus": "CRITICAL"
        },
        {
            "subSystem": "{Memory}",
            "rollUpStatus": "NORMAL"
        },
        {
            "subSystem": "{PSU}",
            "rollUpStatus": "NORMAL"
        },
```

```
        {
            "subSystem": "{Processor}",
            "rollUpStatus": "CRITICAL"
        },
        {
            "subSystem": "{Temperature}",
            "rollUpStatus": "NORMAL"
        },
        {
            "subSystem": "{Voltage}",
            "rollUpStatus": "NORMAL"
        },
        {
            "subSystem": "{Battery}",
            "rollUpStatus": "NORMAL"
        },
        {
            "subSystem": "{PowerConsumption}",
            "rollUpStatus": "NORMAL"
        },

    ]
}
```


For more information, see Response body on page 102.

16

Host management

Topics:

Get the host overview

Description: Gets the host overview of successfully inventoried host.

Command or URL: `/Services/InventoryService/Hosts/{id}/HostOverview`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Table 57. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Get the host ID from `/Services/ConsoleService/Consoles/{Console-id}/Hosts` API. For more information, see Get list of all hosts in vCenter on page 42.	N/A	Path	String

HTTP response code:

Table 58. HTTP response code

Code	Description or response object
200	HostOverview
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests

Table 58. HTTP response code (continued)

Code	Description or response object
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "modelName": "{Server model name}",
    "hostName": "{Host name}",
    "serviceTag": "{Host Service Tag}",
    "powerState": "{Power state (on or off)",
    "assetTag": "{Asset tag}",
    "warrantyDaysleft": {Number of warranty days left},
    "warrantyStatus": {Warranty status},
    "warrantyNotificationEnabled": {warranty notification enabled on the Settings page
(True or False)},
    "operatingSystem": "{OS name}",
    "operatingSystemVersion": "{OS version}",
    "HostcredentialProfileName": "{Host credential profile name}",
    "lastInventoryScan": "{Date and time when invetory is run last time}",
    "systemId": "{System ID}",
    "isBlade": {Blade Server (True or false)} ,
    "chassisInfo": null,
    "frmState": "{firmware state}",
    "lockDownState": "{System Lockdown status (Enabled or Disabled)}",
    "biosFirmware": "{BIOS Firmware version}",
    "idracFirmware": "{iDRAC firmware version}",
    "idracIpaddress": "{iDRAC IP address}",
    "managementURI": "{CMC or OME-M URL}",
    "vcenterIpAddress": "{vCenter IP address}"
}
```

For more information, see Response body on page 102.

Get host firmware inventory

Description: Gets the host firmware inventory details. The information displayed here is real-time information.

Command or URL: `/Services/InventoryService/Hosts/{id}/FirmwareInventory`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Table 59. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Get the host ID from `/Services/ConsoleService/Consoles/{Console-id}/Hosts` API. For more information, see Get list of all hosts in vCenter on page 42.	N/A	Path	String

HTTP response code:

Table 60. HTTP response code

Code	Description or response object
200	FirmwareInventory

Table 60. HTTP response code (continued)

Code	Description or response object
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
        "componentType": "{component type}",
        "elementName": "{Name of the software component}",
        "installationDate": "{component installation date}",
        "versionString": "{firmware version}",
        "componentId": "{componenent ID}",
        "deviceId": "{device ID}",
        "identityInfoValue": "{component type:component ID}",
        "instanceId": "{Unique identifier}",
        "isEntity": "{true or false}",
        "majorVersion": "{component major version}",
        "minorVersion": "{component minor version}",
        "releaseDate": null,
        "revisionNumber": "{revison number}",
        "serialNumber": "{serial number}",
        "status": "{installation status}",
        "subDeviceId": "{sub device ID}",
        "subVendorId": "{sub vendor ID}",
        "vendorId": "{vendor ID}"
```

For more information, see Response body on page 102.

Get host warranty

Description: Gets the host warranty.

Command or URL: `/Services/InventoryService/Hosts/{id}/Warranty`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Table 61. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Get the host ID from `/Services/ConsoleService/Consoles/{Console-id}/Hosts` API. For more information, see Get list of all hosts in vCenter on page 42.	N/A	Path	String

HTTP response code:

Table 62. HTTP response code

Code	Description or response object
200	WarrantyInfo

Table 62. HTTP response code (continued)

Code	Description or response object
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "overAllStatus": "{overall status of warranty}",
    "warranty": [
        {
            "hostName": "{host name}",
            "serviceTag": "{host Service Tag}",
            "id": {Host ID},
            "serviceLevelCode": "{Types of warranty}",
            "serviceLevelDescription": "{Warranty description}",
            "provider": "{Provider}",
            "startDate": "{Warranty start date}",
            "endDate": "{warranty end date}",
            "daysLeft": "{number of days left}",
            "entitlementType": "{entitlement type}",
            "lastUpdated": "{The date and time when the warranty job is run}",
            "status": "{Warranty status of particular warranty component(active or
expired)}"
        }
    ]
}
```

For more information, see Response body on page 102.

Get alarms and events

Description: Get alarms and events.

Command or URL: `/Services/ConsoleService/AlarmsAndEvents`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Parameters: None

HTTP response code:

Table 63. HTTP response code

Code	Description or response object
200	Get Alarms and Event
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
{
    "alarmStatus": {hosts and chassis alarm status (True or false)},
    "mprAlarmStatus": {MPR forecast status (true or false)},
    "eventPostType": "{event posting level option}"
}
```

For more information, see Response body on page 102.

Set alarms and events

Description: Sets alarms and events.

Command or URL: `/Services/ConsoleService/AlarmsAndEvents`

Method: PUT

Request Body:

```
{
    "alarmStatus":  {Host and chassis alarm status (true or false)},
    "mprAlarmStatus": {MPR forecast alarm status(true or false)},
    "eventPostType": "{event posting level options}"
}
```

For more information about request body parameters, see Request body on page 98.

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Parameters: None

HTTP response code:

Table 64. HTTP response code

Code	Description or response object
200	Updated Alarms and Event
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
    Alerts and Event updated successfully
```

For more information, see Response body on page 102.

Get host power supply information

Description: Gets the information about the host power supplies.

Command or URL: `/Services/InventoryService/Hosts/{id}/Hardware/PowerSupplies`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Parameters:

Table 65. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Get the host ID from `/Services/ConsoleService/Consoles/{Console-id}/Hosts` API. For more information, see Get list of all hosts in vCenter on page 42.	N/A	Path	String

HTTP response code:

Table 66. HTTP response code

Code	Description or response object
200	Powersupply info
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
PowerSupplyInfo:
{
    "powerSupplyCount": {PSU count} ,
    "powerSupplies": [
        {
            "type": "{Power supply type}",
            "location": "{PSU location}",
            "outputWatts": "{Output in watts}",
            "status": "{PSU status}"
        },
        {
            "type": "{Power supply type}",
            "location": "{PSU location}",
            "outputWatts": {Output in watts},
            "status": "{PSU status}"
        }
    ]
}
```

For more information, see Response body on page 102.

Get host memory information

Description: Gets the information about the host memories.

Command or URL: `/Services/InventoryService/Hosts/{id}/Hardware/Memories`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Table 67. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Get the host ID from `/Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

HTTP response code:

Table 68. HTTP response code

Code	Description or response object
200	MemoryInfo
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
MemoryInfo:
{
    "maxSizeMB": "{Max size}",
    "numSocketsTotal": {total number of sockets},
    "numSocketsUsed": {Total number of sockets in use},
    "memoryList": [
        {
            "slot": "{Memory slot}",
            "size": "{Memory size}",
            "type": "{Memory type}"
        },
        {
            "slot": "{Memory slot}",
            "size": "{Memory size}",
            "type": "{Memory type}"
        },
```

For more information, see Response body on page 102.

Get host processors information

Description: Gets the information about host processors.

Command or URL: `/Services/InventoryService/Hosts/{id}/Hardware/Processors`

Method: GET

Authorization: Bearer authentication

vCenter privileges required: Dell.Inventory.Configure Inventory

Table 69. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Get the host ID from `/Services/ConsoleService/Consoles/{Console-id}/Hosts` API. For more information, see Get list of all hosts in vCenter on page 42.	N/A	Path	String

HTTP response code:

Table 70. HTTP response code

Code	Description or response object
200	Processor
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

For OMIVV-Specific error codes, see OMIVV-Specific error codes on page 121.

Example Response:

```
Processor:
    {
        "socket": "{CPU socket information}",
        "currentSpeed": {Current speed of CPU},
        "brand": "{CPU brand}",
        "version": "{CPU version}",
        "coreCount": {core count}
    }
```

For more information, see Response body on page 102.

Get host hardware FRUs

Description: Gets the host hardware FRUs.

Command or URL: `Services/InventoryService/Hosts/{id}/Hardware/Frus`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 71. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Use the host ID retrieved from `//Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 72. HTTP response code

Code	Description or response object
200	FRU
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
[
    {
        "partNumber": "{FRU part number}",
        "partName": "{FRU part name}",
        "manufacturer": "{Manufacturer name}",
        "serialNumber": "{Serial number of the manufacturer}",
        "manufacturerDate": "{Manufacturer Date}",
    },

]
```

For more information, see Response body on page 102.

Get host hardware NICs

Description: Gets the host hardware NICs.

Command or URL: `/Services/InventoryService/Hosts/{id}/Hardware/Nics`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 73. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Use the host ID retrieved from `//Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 74. HTTP response code

Code	Description or response object
200	NIC
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied

Table 74. HTTP response code (continued)

Code	Description or response object
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
{
    "nicCount": {Total number of NICs},
    "nicList": [
        {
            "name": "{NIC name}",
            "manufacturer": "{NIC manufacturer name}",
            "macAddress": "{NIC MAC address}",
        }
```

For more information, see Response body on page 102.

Get host hardware PCI slots

Description: Get the host hardware PCI slots information.

NOTE:

Command or URL: `/Services/InventoryService/Hosts/{id}/Hardware/PciSlots`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 75. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Use the host ID retrieved from `//Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 76. HTTP response code

Code	Description or response object
200	PciSlots
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
{
    "pciSlotsTotal": {Total number of PCI slots},
    "pciSlotsUsed": {Total number of PCI slots used},
    "pciSlots": [
        {
            "fqdd": "{FQDD of PCI}",
            "manufacturer": "{PCI manufacturer}",
            "description": "{PCI description}",
            "type": "{PCI type}",
            "width": "{PCI width}"
        }
     ]
}
```

NOTE: Along with PCI slot information, other slot information such as Power Supply, Fan, DIMM, Processor, SD card, IDSDM, Physical Disk, Enclosure Fan, and Enclosure Power Supply are displayed in the API response for modular servers.

For more information, see Response body on page 102.

Get host hardware remote access card

Description: Get the host hardware remote access card.

Command or URL: `/Services/InventoryService/Hosts/{id}/Hardware/RemoteAccessCard`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 77. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Use the host ID retrieved from `//Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 78. HTTP response code

Code	Description or response object
200	RemoteAccessCard
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
{
    "ipAddress": "{Remote access card IP address}",
    "macAddress": "{Remote access card MAC address}",
    "racType": "{Remote access card type}",
```

```
    "url": "{https://<iDRAC IP>:<portnumber>}"
}
```

For more information, see Response body on page 102.

Get host storage details

Description: Gets the host storage details. The storage API response is an array of controller object. Each controller object displays information such as enclosure, physical disk, virtual disk, and controller information. Each enclosure displays information such as physical disk and enclosure information.

If physical disk is not part of enclosure, the response displays as null.

Command or URL: `/Services/InventoryService/Hosts/{id}/Storage`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 79. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Use the host ID retrieved from `//Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 80. HTTP response code

Code	Description or response object
200	Storage
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
[
    {
        "enclosures": {Enclosure ID},
        "physicalDisks": {Array of physical disks},
        "virtualDisks": {Array of virtual disks},
        "controllerId": "{Controller ID}",
        "fqdd": "{Controller FQDD}",
        "name": "{Controller name}",
        "firmwareVersion": "{Controller firmware version}",
        "minimumRequiredFirmwareVersion": "{Minimum required firmware version}",
        "driverVersion": "{Minimum driver version}",
        "cacheMemorySize": "{Cache memory size}",
        "patrolReadState": "{Patrol Read State}"
    },
    {
        "enclosures": {Enclosure ID},
        "physicalDisks": [
```

```
            {
                "name": "{Physical disk name}",
                "fqdd": "{Physcial disk FQDD}",
                "deviceId": "{Physical disk device ID}",
                "capacity": "{Physical disk capacity}",
                "partNum": "{Physical disk part number}",
                "vendorName": "{Vendor name}",
                "serialNum": "{Physical disk serial Number}",
                "modelNum": "{Model number}",
                "diskStatus": "{Physical disk status}",
                "hotSpare": "{Hot spare type}",
                "busProtocol": "{Bus protocol}",
                "contollerId": "{Controller ID}",
                "enclosureId": "{Enclosure ID}",
                "connectorId": "{Connector ID}",
                "mediaType": "{Media Type (SSD or HDD or SAS)}",
                "remainingRatedWriteEndurance": "{remaining write endurance}"
            },
        ],
        "virtualDisks": [
            {
                "deviceId": "{Virtual disk device ID}",
                "fqdd": "{Virtual disk FQDD}",
                "name": "{Virtual disk name}",
                "deviceName": "{Virtual disk device name}",
                "size": "{Virtual disk device size}",
                "layout": "{Layout type of the virtual storage}",
                "mediaType": "{Media Type (SSD or HDD or SAS)}",
                "busProtocol": "{Bus protocol}",
                "defaultReadPolicy":"{the default read policy that is supported by the
controller}",
                "defaultWritePolicy":"{the default write policy that is supported by the
controller}",
                "diskCachePolicy": "{Cache policy if enabled}",
                "stripeSize": "{Stripe size}",
                "physicalDiskKey": [
                    "ControllerId:"{Controller ID:Connector ID:Physical disk name}",
                    "ControllerId:"{Controller ID:Connector ID:Physical disk name}"
                ],
                "controllerId": "{Controller ID}"
            }
        ],
        "controllerId": "{Controller ID}",
        "fqdd": "{Controller FQDD}",
        "name": "{Controller name}",
        "firmwareVersion": "{Controller firmware version}",
        "minimumRequiredFirmwareVersion": "{Minimum required firmware version}",
        "driverVersion": "{Driver version}",
        "cacheMemorySize": "{Cache memory size}",
        "patrolReadState": "{Patrol Read State}"
    }
]
```


For more information, see Response body on page 102.

Get host power monitoring details

Description: Gets the host power monitoring details

Command or URL: `/Services/InventoryService/Hosts/{id}/PowerMonitoring`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 81. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Use the host ID retrieved from `//Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 82. HTTP response code

Code	Description or response object
200	PowerMonitoring
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
    {
    "powerBudget": {Power profile},
    "warningThreshold": "{Warning thresholds in watts}",
    "currentProfile": "{Power profile to maximize performance of your system and
conserve energy}",
    "failureThreshold": "{Failure thresholds in watts}",
    "energyStatistics": {
        "energyConsumption": "{The energy consumption of the host}",
        "energyConsumptionStartDateTime": "{The date and time when the host began to
consume power}",
        "energyConsumptionEndDateTime":"{The date and time when the host stopped to
consume power}",
        "systemPeakPower": "{The host peak power}",
        "systemPeakPowerStartDateTime":"{The date and time when the host peak power
started}",
        "systemPeakPowerEndDateTime": "{the date and time when the host peak power
ended}",
        "systemPeakAmps": "{The hosts peak amps}",
        "peakAmpsStartDateTime": "{The starting date and time of the host peak amps}",
        "peakAmpsEndDateTime": "{The end date and time of the host peak amps}"
    },
    "reservePowerCapacity": {
        "instant": "{Instant power}",
        "peak": "{Peak power}"
    }
}
```

For more information, see Response body on page 102.

Get host system event log

Description: Gets the host system event log.

Command or URL: `Services/InventoryService/Hosts/{id}/SystemEventLog`

Method: GET

Authorization: Bearer authentication

Parameters:

Table 83. Parameters

Parameter	Value	Description	Default value	Parameter type	Data type
id	(required)	Host ID. Use the host ID retrieved from `//Services/ConsoleService/Consoles/{Console-id}/Hosts` API.	N/A	Path	String

vCenter privileges required: Dell.Inventory.Configure Inventory

HTTP response code:

Table 84. HTTP response code

Code	Description or response object
200	SystemEventLog
400	Operational Context not set
401	Authorization failure
403	vCenter permission denied
429	Too many requests
500	Internal Server error / timeout

Example Response:

```
{
    "selLog": [
        {
            "index": "{serial number}",
            "status": "{Severity levels (Normal, Warning, and Critical)}",
            "timeStampString": "{Date with time for the specific event in UTC format}",
            "description": "{Event description}"
        }
]
 "noOfRecords": {Total number of system event logs}
}
```

For more information, see Response body on page 102.

A

Request body

Start an OMIVV session

Table 85. Start an OMIVV session API

Parameter	Description
username	OMIVV username. Only admin user is allowed to login to API.
password	OMIVV password

Set vCenter context

Table 86. Set vCenter context API

Parameter	Description
consoleId	vCenter ID
username	vCenter username
domain	vCenter domain
password	vCenter password

Create and modify repository profile

Table 87. Create and modify repository profile

Parameter	Description
name	Repository profile name
description	Repository profile description
globalDefault	Dell default repository profile. Currently, this parameter is not supported.
repoType	Repository profile type (Firmware and Driver). You cannot edit this parameter while modifying the profile.
protocolType	Protocol Type (NFS or CIFS). For NFS, credential is not required.For CIFS credentials are required. domain parameter is not mandatory.
uri	Catalog location. Catalog file path for firmware: `host:/share/filename.xml`, `host:/share/filename.gz`, `\\\\host\\share\\filename.xml`, or `\\host\\share\\filename.gz`. Catalog file path for driver: `host:/share/foldername`, `\\\\host\\share\\foldername`
username	share username
domain	Domain

Table 87. Create and modify repository profile (continued)

Parameter	Description
password	share password
synchronizeRepository	Synchronize with current repository location (always set to true)

Create host or cluster level firmware inventory report

Table 88. Host or cluster level firmware inventory report

Parameter	Description
reportType	Inventory report type (Host or Cluster)
hostID	Host ID
clusterID	Cluster ID
repoProfieID	Repository profile ID
systemId	System ID
bundleID	Bundle ID

Create cluster or host level firmware update job

Table 89. Cluster or host level firmware update job

Parameter	Description
jobname	Firmware update job name
jobdesc	Firmware update job description
updateType	Update type (FIRMWARE). The value for update type is case-sensitive. Enter the value in upper case. OMIVV supports only FIRMWARE update type.
updateTartgetType	Update Target Type (host or cluster). Host—used to update single host Cluster—used to update the hosts under cluster
runLater	Scheduled to run at a specified time (true or false)
runNow	Run firmware update job now (true or false)
dateTime	Firmware update job schedule (date and time format: YYYY-MM-DDTHH:MM:SSZ) for runLater. Enter date and time in 24 hour UTC format.
firmwareRepoProfileID	Repository profile ID. To get the repository profile details, run `/Services/PluginProfileService/RepositoryProfiles`.
rebootOptions	Reboot options. Safe reboot: Apply Updates and Reboot after entering Maintenance mode. Applicable for cluster level, vSphere, vSAN, and datacenter host level. Next reboot: Apply updates on next reboot. Applicable for vSphere, datacenter, and vSAN host level.

Table 89. Cluster or host level firmware update job (continued)

Parameter	Description
	Force reboot: Apply updates and force reboot without entering maintenance mode. Applicable for vSphere and datacenter host level. **NOTE:** If you select the Next reboot or Force reboot options, the following parameters are not applicable: exitMaintenanceMode, migratePoweredOffAndSuspendedVMs, enterMaintenanceModetimeout, enterMaintenanceModeOption
preCheck	Prerequisites check before firmware update. Ensure that prerequisites checks are met in your environment: For vSAN host and cluster: DRS is enabled Host is not already in maintenance mode vSAN data objects are healthy For vSphere host and cluster: DRS is enabled. The possible input value is true or false. Default value is true if the required value is not entered. For cluster-level firmware update, the only allowed value is true.
hostId	Host ID. You can update only 64 hosts. To get the host ID, run `/Services/ConsoleService/Consoles/{id}`
clusterId	Cluster ID. To get the cluster ID, run `/Services/ConsoleService/Consoles/{id}`.
bundleId	Bundle ID. To get the bundle ID, run `/Services/RepositoryManagementService/RepositoryData?bundleId={bundleId}`
packageIDs	package ID. To get the package ID, run `/Services/UpdateService/FWReport`. Package ID of the up-to-date components are not supported and firmware update job fails.
exitMaintenanceMode	Exit maintenance mode after firmware update completes. If you disable this option, host remains in maintenance mode.
migratePoweredOffAndSuspendedVMs	Move powered-off and suspended virtual machines to other hosts in cluster. Disabling this option disconnects VM until the host device is online.
resetIDracAndDeleteJobs	Clears all the iDRAC jobs present in the Job Queue followed by iDRAC reset before updating firmware on the host. If you do not mention any value while running API, OMIVV considers the settings configured on the **Settings** > **Firmware Update Settings** on the User Interface.
enterMaintenanceModetimeout	Enter the Maintenance Mode timeout value between 60–1440 minutes. If the wait time goes beyond the specified time, the update jobs fail and enter maintenance task will be canceled

Table 89. Cluster or host level firmware update job (continued)

Parameter	Description
	or timed out. However, the components may get updated automatically when the host is restarted.
enterMaintenanceModeOption	Enter maintenance mode option. This option is applicable for vSAN host and cluster. Supported options are: Ensure accessibility Full Data migration No data migration

Upload license to OMIVV

Table 90. Upload license to OMIVV

Parameter	Description
sharetype	Repository share location (NFS/CIFS).
path	License file path. The format of the license file path is `\\\\IP-hostname-fqdn\\pathfolder1\\pathfolder2\\..\\pathfolderN\\<filename.xml>`.
credential	Username, domain, and password. For NFS, credential is not required. For CIFS credentials are required. domain parameter is not mandatory.

Set alarms and events

Table 91. Set alarms and events

Parameter	Description
alarmStatus	Host and chassis alarm status (true or false)
mprAlarmStatus	MPR alarm status(true or false)
eventPostType	Event posting level options. Available options are Do not post any events, Post all events, Post only critical and warning events, and Post only virtualization-related critical, and warning events.

B

Response body

This topic describes each parameter in the example response.

Start an OMIVV session

Table 92. Start an OMIVV session

Attribute	Type	Description
accessToken	String	Token ID
tokenType	String	Token type
expiresAt	Integer and string	Expiry date and time for token ID.

Get list of vCenter and host license

Table 93. Get list of vCenter and host license

Attribute	Type	Description
id	String	License ID
entitlementID	String	Entitlement ID
licenseType	Integer and string	License Type (Standard or Evaluation)
maxHosts	String	Number of maximum hosts allowed
startDate	Integer and string	The date when the license is uploaded to OMIVV
expirationDate	Integer and string	License expiry date
duration	String	License duration
licenseStatus	String	License status

Get host and vCenter license details, upload vCenter license

Table 94. Get host and vCenter license details, upload vCenter license

Attribute	Type	Description
id	String	License ID
entitlementID	String	Entitlement ID
licenseType	String	License Type (Standard or Evaluation)
maxHosts	String	Number of maximum hosts allowed
startDate	Integer and string	The date when the license is uploaded to OMIVV
expiryDate	Integer and string	License expiry date
duration	String	Duration of the license

Table 94. Get host and vCenter license details, upload vCenter license (continued)

Attribute	Type	Description
licenseStatus	String	Status of license (ACTIVE, INACTIVE, and EXPIRED)

Get list of registered vCenters

Table 95. Get list of registered vCenters

Attribute	Type	Description
id	String	vCenter ID
hostname	String	vCenter Hostname or FQDN
ip	String	vCenter IP

Get list of all hosts in vCenter

Table 96. Get list of all hosts in vCenter

Attribute	Type	Description
id	String	Host ID
servicetag	String	service tag of a host

Get host details

Table 97. Get host details

Attribute	Type	Description
id	String	Host ID
serviceTag	String	Service Tag of host
hostip	String	Host IP
hostName	String	Hostname
managementIP	String	iDRAC IP
model	String	Server model name
systemId	String	System ID

Get vCenter tree view of datacenter

Table 98. Get vCenter tree view of datacenter

Attribute	Type	Description
id	String	vCenter ID or datacenter ID or cluster ID or Host ID
hostname	String	vCenter hostname or FQDN. The hostname value is displayed in the response only if the hostname

Table 98. Get vCenter tree view of datacenter (continued)

Attribute	Type	Description
		is configured properly using DNS. Otherwise, hostname value displayed as null or blank.
ip	String	vCenter IP
name	String	Datacenter name or cluster name
registeredWithVlcm	Boolean	vSphere Lifecycle Manager (vLCM) registration status (true or false)

Get cluster details

Table 99. Get cluster details

Attribute	Type	Description
id	String	Cluster ID or host ID
name	String	Cluster name
cluster type	String	Cluster type (vSAN or vSphere)
hostip	String	Host IPv4
managementIP	String	iDRAC IP
hostname	String	vCenter host name
serviceTag	String	Service Tag of the server
model	String	Server model name
systemId	String	System ID

Get cluster health

Table 100. Get cluster health

Attribute	Type	Description
clusterId	String	Cluster ID
drsState	String	DRS state (Enabled or Disabled)
vsanObjectHealth	String	vSAN object health status (Healthy, Unhealthy, or N/A)

Get list of cluster profiles

Table 101. Get list of cluster profiles

Attribute	Type	Description
id	String	Cluster profile ID
profileName	String	Cluster profile name
description	String	Cluster profile description

Get details of cluster profiles

Table 102. Get details of cluster profiles

Attribute	Type	Description
id	String	Cluster ID or cluster profile ID or repository profile ID or system profile ID or drift detection job ID
profileName	String	Cluster profile name or repository profile name or system profile name
description	String	Description of cluster profile or repository profile or system profile
name	String	Cluster name
repoType	String	Repository profile type (Driver or Firmware)
status	String	Drift detection job status

Get list of repository profiles

Table 103. Get list of repository profiles

Attribute	Type	Description
id	String	vCenter ID
profileName	String	Repository profile name

Get repository profile details

Table 104. Get repository profile details

Attribute	Type	Description
id	String	Repository profile ID
name	String	Repository profile name
description	String	Repository profile description.
globalDefault	Boolean	Dell EMC default catalog repository profile (True). Factory-created firmware repository profile.
repoType	String	Repository Type (Firmware or Driver)
protocolType	String	Protocol Type (NFS, CIFS). For NFS, credential is not required.For CIFS credentials are required. domain parameter is not mandatory.
uri	String	Catalog location
synchronzeRepository	Boolean	Synchronize with currently repository location (Enabled or Disabled)
username	String	share username
domain	String	domain name
password	String	share password

Create and modify repository profile

Table 105. Create and modify repository profile

Parameter	Description
name	Repository profile name
description	Repository profile description
globalDefault	Dell default repository profile. Currently, this parameter is not supported.
repoType	Repository profile type (Firmware and Driver). You cannot edit this parameter while modifying the profile. Ensure that you enter value for repository type in upper case letters.
protocolType	Protocol Type (NFS or CIFS). For NFS, credential is not required.For CIFS credentials are required. domain parameter is not mandatory.
uri	Catalog location
username	share username
domain	Domain
password	share password
synchronizeRepository	Synchronize with current repository location (always set to true)

Get firmware repository inventory details

Table 106. Get firmware repository inventory details

Attribute	Type	Description
repoProfileID	String	Repository profile ID
bundleID	String	Bundle ID
name	String	Bundle name
description	String	Bundle description
model	String	Server Model
targetOS	String	Target host OS name. Windows target OS used for firmware update using iDRAC.
systemId	String	System ID
packageID	String	Component unique package ID
rebootRequired	Boolean	Host reboot required (True or False)
releaseDate	Integer and string	Component release date and month
componentID	String	Supported hardware component ID
deviceID	String	Supported hardware device ID
subDeviceID	String	Supported hardware sub device ID
subVendorID	String	Supported hardware sub vendor ID
vendorID	String	Supported hardware sub vendor ID

Create host level firmware inventory report

Table 107. Create host level firmware inventory report

Attribute	Type	Description
reportType	String	Inventory report type
id	String	Host ID
hostip	String	Host IPv4 or FQDN
hostName	String	Hostname
managementIP	String	iDRAC IP
serviceTag	String	Host Service Tag
model	String	Server model
systemId	String	System ID
componentType	String	Component type
packageId	String	Component unique package ID
component	String	Component name
currentVersion	String	Current installed component version
availableVersion	String	Available component version
criticality	String	Importance of component update (Urgent, Recommended, Optional, Security, Performance)
updateAction	String	Component update status (Up to date, downgrade, upgrade)
scheduled	String	Component update job scheduled
rebootRequired	Boolean	Host reboot required (True or False)
releaseDate	Integer and String	Component release date and month

Create cluster level firmware inventory report

Table 108. Create cluster level firmware inventory report

Attribute	Type	Description
reportType	String	Inventory report type
id	String	Host ID or cluster ID
hostip	String	Host IPv4 or FQDN
hostName	String	Hostname
managementIP	String	iDRAC IP
serviceTag	String	Host Service Tag
model	String	Server model
systemId	String	System ID
componentType	String	Component type
packageId	String	Component unique package ID
component	String	Component name

Table 108. Create cluster level firmware inventory report (continued)

Attribute	Type	Description
currentVersion	String	Current installed component version
availableVersion	String	Available component version
criticality	String	Importance of component update (Urgent, Recommended, Optional, Security, Performance)
updateAction	String	Component update status (up to date, downgrade, upgrade)
scheduled	String	Component update job scheduled
rebootRequired	Boolean	Host reboot required (True or False)
releaseDate	Integer and String	Component release date and month

Create host or cluster level firmware update job

Table 109. Create host or cluster level firmware update job

Attribute	Type	Description
id	String	Firmware update job ID
Status	String	Firmware update job status.

Get list of host firmware update jobs

Table 110. Get list of host firmware update jobs

Attribute	Type	Description
id	String	Firmware update job ID
status	String	Status of the firmware update job (Scheduled, Cancelled, Staging, Success, and Failed)

Get host or cluster firmware update job details

Table 111. Get host or cluster firmware update job details

Attribute	Type	Description
id	String	Firmware update job ID or host ID
status	String	Status of the firmware update job
jobname	String	Firmware update job name
jobdesc	String	Firmware update job description
updateType	String	Update type
runNow	Boolean	Run update job now
runLater	Boolean	Run update job later
dateTime	Integer and String	Update job schedule date and time
repoProfileID	String	Firmware repository profile ID

Table 111. Get host or cluster firmware update job details (continued)

Attribute	Type	Description
hostip	String	Host IPv4 or FQDN
hostName	String	Hostname
managementIP	String	IDRAC IP
serviceTag	String	Host Service Tag
model	String	Server Model
systemId	String	System ID
bundleId	String	Bundle ID
packageIDs	String	Component unique package ID
rebootOptions	String	Reboot options. Safe reboot: Apply Updates and Reboot after entering Maintenance mode. Applicable for cluster level, vSphere, vSAN, and datacenter host level. Next reboot: Apply updates on next reboot. Applicable for vSphere, datacenter, and vSAN host level. Force reboot: Apply updates and force reboot without entering maintenance mode. Applicable for vSphere and datacenter host level.
precheck	Boolean	Prerequisites check before firmware update. Ensure that prerequisites checks are met in your environment: For vSAN host and cluster: DRS is enabled Host is not already in maintenance mode vSAN data objects are healthy For vSphere host and cluster: DRS is enabled
migratePoweredOffAndSuspendedVMs	Boolean	Move powered-off and suspended virtual machines to other hosts in cluster. Disabling this option disconnects VM until the host device is online.
resetiDracAndDeleteJobs	Boolean	Clears all the iDRAC jobs present in the Job Queue followed by iDRAC reset before updating firmware on the host. If you do not mention any value while running API, OMIVV considers the settings configured on the **Settings** > **Firmware Update Settings** on the User Interface.
enterMaintenanceModetimeout	String	Enter the Maintenance Mode timeout value between 60–1440 minutes.

Table 111. Get host or cluster firmware update job details (continued)

Attribute	Type	Description
		If the wait time goes beyond the specified time, the update jobs fail and enter maintenance task will be canceled or timed out. However, the components may get updated automatically when the host is restarted
enterMaintenanceModeOption	String	Enter maintenance mode option. This option is applicable for vSAN host and cluster. Supported options are: Ensure accessibility Full Data migration No data migration

Get list of system profiles and get system profile details

Table 112. Get list of system profiles and get system profile details

Attribute	Type	Description
id	String	system profile ID
name	String	system profile name
description	String	profile description
systemProfileType	String	system profile type (basic or advanced)
referenceESXiHostNameOrIP	String	Reference server hostname or hostIP
referenceManagementIP	String	iDRAC IP
referenceiDRACType	String	iDRAC license type
serviceTag	String	Service Tag of a host
serverModel	String	Server model name
dateCreated	String	Date and time when the profile is created
dateModified	String	Date and time when the profile is modified
lastModifiedBy	String	Details of the user who modified the profile

Get firmware drift report

Table 113. Get firmware drift report

Attribute	Type	Description
id	String	Host ID
hostip	String	Host IP
hostname	String	Hostname

Table 113. Get firmware drift report (continued)

Attribute	Type	Description
managementIP	String	iDRAC IP
Service Tag	String	Service Tag of the host
model	String	Server Model Name
systemID	String	System ID
complianceStatus	String	Compliance Status. You may get any of the following values for compliant status: VERSION_MISMATCH NEW COMPONENT COMPONENT MISSING/ATTRIBUTE_MISMATCH NOT_APPLICABLE
nonCompliantType	String	Reason for non-compliance
noncompliantTypeDescription	String	Detailed description for non-compliance
componentID	String	Component ID
packageID	String	Package ID
bundleID	String	Bundle ID
instanceID	String	FQDD
componentName	String	Component Name
componentType	String	Component Type
componentTypeDisplay	String	Component Type Display. For example, firmware is displayed as FRMW.
upgrade	Boolean	Component Upgrade required (True or False)
criticality	String	Importance of component update
rebootRequired	Boolean	Host reboot required (true or false)
deviceID	String	Device ID
subDeviceID	String	Sub device ID
subVendorID	String	Sub Vendor ID
vendorID	String	Vendor ID
driftedVersionInfo	String	Component version in host
baselineVersionInfo"	String	component version in repository

Get drift report for all clusters

Table 114. Get drift report for all clusters

Attribute	Type	Description
id	String	Cluster ID or cluster profile ID.
name	String	Cluster name or cluster profile name

Table 114. Get drift report for all clusters (continued)

Attribute	Type	Description
clusterComplianceStatus	String	Compliance status of the cluster (Complaint or non-compliant)
systemprofileName	String	System profile name. If any system profile is not associated with cluster profile, profile name is displayed as null.
fmRepoProfileName	String	Firmware repository profile name. If any firmware repository profile is not associated with cluster profile, profile name is displayed as null.
driverRepoProfileName	String	Driver repository profile name. If any driver repository profile is not associated with cluster profile, profile name is displayed as null.

Get drift report for specific cluster

Table 115. Get drift report for specific cluster

Attribute	Type	Description
id	String	Cluster ID or cluster profile ID
name	String	Cluster name or cluster profile name
clusterComplianceStatus	String	Cluster compliance status
complianceStatus	String	Firmware or driver or configuration drift compliance status
fwDriftDetails	String	Firmware drift details. If any firmware repository profile is not associated with cluster profile, fwDriftDetails value is displayed as null.
driverDriftDetails	String	Driver drift details. If any driver repository profile is not associated with cluster profile, driverDriftDetails parameter value is displayed as null.
configurationDriftDetails	String	Configuration drift details. If any system profile is not associated with cluster profile, configurationDriftDetails parameter value is displayed as null.
nonCompliantHostList	String	List of non-complaint host
notApplicableHostList	String	List of not applicable hosts
compliantHostList	String	List of compliant hosts

Get driver report

Table 116. Get driver report

Attribute	Type	Description
compliantStatus	String	Compliance status
nonCompliantType	String	Non-compliance type

Table 116. Get driver report (continued)

Attribute	Type	Description
noncompliantTypeDescription	String	Reason for non-compliance
vendorName	String	Component vendor name
componentName	String	Component name
driverName	String	Driver name
packageType	String	Package type (Component, Bulletin)
rebootRequired	Boolean	Host reboot required (true or false)
recommendation	Boolean	Recommendation (Yes or No)
compliantStatus	String	Complaint status
nonCompliantType	String	non-compliance type. If the driver is not installed, the noncomplaintType displays **NEW_COMPONENT**.
noncompliantTypeDescription	String	Reason for non-compliance
driftedVersionInfo	String	Installed driver version. If driver is not installed, driftedVersionInfo parameter value is displayed as N/A
baselineVersionInfo	String	Driver component version in repository
id	String	Host ID
hostip	String	Host IP
hostname	String	Hostname
managementIP	String	iDRAC IP
serviceTag	String	Service Tag of host
model	String	Server model name
systemId	String	System ID

Get configuration drift report

Table 117. Get configuration drift report

Attribute	Type	Description
id	String	Host ID
hostip	String	Host IP
hostName	String	Hostname
managementIP	String	iDRAC IP
serviceTag	String	Service Tag of host
model	String	Server Model Name
systemId	String	System ID
compliantStatus	String	Compppliance Status
nonCompliantType	String	Non-complaint Type
noncompliantTypeDescription	String	Reason for non-compliance
componentName	String	Mismatched component name

Table 117. Get configuration drift report (continued)

Attribute	Type	Description
componentFqdd	String	Component FQDD
subComponents	String	Sub components
containSubComponent	Boolean	Presence of child components
missingGroups	String	Missing groups from source and target system profile
misMatchedGroups	String	List of mismatched group names
name	String	Attribute name
missingAtributes	String	Attributes which are present in source but not present in target
misMatchedAtributes	String	List of mismatched attributes
attributeName	String	Attribute Name
attributeLctValue	String	Baseline value
attributeDeviceValue	String	Attribute value configured in server (Enabled or Disabled)

Get subsystem health report (OMIVV Host Health)

Table 118. Get subsystem health report (OMIVV Host Health)

Attribute	Type	Description
overallHealth	String	Overall health status of servers
subsystem	String	Subsystem name (For example, Fan, Memory, PSU, Processor, Temperature, Voltage, Battery, PowerConnection)
rollUpStatus	String	Status of the components (Normal, Critical, and Warning)

Get the host overview

Table 119. Get the host overview

Attribute	Type	Description
modelName	String	server model name
hostName	String	host name
seriveTag	String	Host service tag
powerState	Boolean	Power State (on or off)
assetTag	String	Asset tag
warrantyDaysleft	Integer	Number of warranty days left
warrantyStatus	String	Warranty status
warrantyNotificationEnabled	Boolean	warranty notification enabled on the Settings page (True or False)
operatingSystem	String	OS name

Table 119. Get the host overview (continued)

Attribute	Type	Description
operatingSystemVersion	String	OS version
HostcredentialProfileName	String	Host credential profile name
lastInventoryScan	Integer	Date and time when inventory is run last time
systemId	String	System ID
isBlade	Boolean	Blade server (true or false)
chassisInfo	String	chassis information
frmState	String	firmware state
lockDownState	Boolean	system lookdown mode status (enabled or disabled)
biosFirmware	String	BIOS firmware version
idracFirmware	String	iDRAC firmware version
idracIpaddress	String	iDRAC IP address
managementURI	String	CMC or OME-M URL
vcenterIpAddress	String	vCenter IP address

Get host firmware inventory

Table 120. Get host firmware inventory

Attribute	Type	Description
componentType	String	component type
elementName	String	name of the software component
installationDate	String	component installation date
versionString	String	firmware version
componentId	String	component ID
deviceId	String	device ID
identityInfoValue	String	component type: component ID
instanceId	String	unique identifier
isEntity	Boolean	The IsEntity property is used to indicate whether the SoftwareIdentity corresponds to a discrete copy of the software component or is being used to convey descriptive and identifying information about software that is not present in the management domain. A value of TRUE indicates that the SoftwareIdentity instance corresponds to a discrete copy of the software component. A value of FALSE indicates that the SoftwareIdentity instance does not correspond to a discrete copy of the Software
majorVersion	String	component major version

Table 120. Get host firmware inventory (continued)

Attribute	Type	Description
minorVersion	String	component minor version
releaseDate	Integer	component release date
revisionNumber	String	revison number
serialNumber	String	serial number
status	String	installation status
subDeviceId	String	sub device ID
subVendorId	String	sub vendor ID
vendorId	String	vendor ID

Get host warranty

Table 121. Get host warranty

Attribute	Type	Description
overAllStatus	String	overall status of warranty
hostName	String	host name
serviceTag	String	host service tag
id	number	host ID
serviceLevelCode	String	types of warranty
serviceLevelDescription	String	warranty description
provider	String	provider
startDate	number	warranty start date
endDate	number	warranty end date
daysLeft	number	number of days left
entitlementType	String	entitlement type
lastUpdated	number	the date and time when the warranty job is run
status	String	Warranty status of particular warranty component(active or expired)

Get alarms and events, set alarm and events

Table 122. Get alarms and events, Set alarm and events

Attribute	Type	Description
alarmStatus	boolean	Host and chassis alarm status (true or false)
mprAlarmStatus	Boolean	Memory Page Retire (MPR) forecast alarm status (true or false)
eventPostType	String	Event posting level options. Available options are Do not post any events, Post all events, Post only critical and warning

Table 122. Get alarms and events, Set alarm and events (continued)

Attribute	Type	Description
		events, and Post only virtualization-related critical, and warning events.

Get host power supply information

Table 123. Get host power supply information

Attribute	Type	Description
powerSupplyCount	String	PSU count
type	String	PSU type
location	String	PSU location
outputWatts	String	Output in watts
status	String	PSU status

Get host memory information

Table 124. Get host memory information

Attribute	Type	Description
maxSizeMB	String	Maximum memory size
numSocketsTotal	String	total number of sockets
numSocketsUsed	String	total number of sockets in use
slot	String	memory slot
size	String	memory size
type	String	memory type

Get host processors information

Table 125. Get host processors information

Attribute	Type	Description
socket	String	CPU socket information
currentSpeed	String	Current speed of CPU
brand	String	CPU brand
version	String	CPU version
coreCount	String	core count

Get host hardware FRUs

Table 126. Response body

Attribute	Type	Description
partNumber	String	FRU part number

Table 126. Response body (continued)

Attribute	Type	Description
partName	String	FRU part name
manufacturer	String	Manufacturer name
serialNumber	String	Serial number of the manufacturer
manufacturerDate	String	Manufacturer Date. The date is displayed in UTC format.

Get host hardware NICs

Table 127. Response body

Attribute	Type	Description
nicCount	String	Total number of NICs
name	String	NIC name
manufacturer	String	NIC manufacturer name
macAddress	String	NIC MAC address

Get host hardware PCI slots

Table 128. Response body

Attribute	Type	Description
pciSlotsTotal	String	Total number of PCI slots
pciSlotsUsed	String	Total number of PCI slots used
fqdd	String	FQDD of PCI
manufacturer	String	PCI manufacturer
description	String	PCI description
type	String	PCI type
width	String	PCI width

Get host hardware remote access card

Table 129. Response body

Attribute	Type	Description
ipAddress	String	Remote access card IP address
macAddress	String	Remote access card MAC address
racType	String	Remote access card type
url	String	The live URL for the iDRAC associated with this host. The format of the URL is https://<iDRAC IP>:<Port number>

Get host power monitoring details

Table 130. Response body

Attribute	Type	Description
powerBudget	String	The Instant and Peak reserve power capacity in watts
warningThreshold	String	Warning thresholds in watts
currentProfile	String	Power profile to maximize performance of your system and conserve energy
failureThreshold	String	Failure thresholds in watts
energyConsumption	String	The energy consumption of the host
energyConsumptionStartDateTime	String	The date and time when the host began to consume power. The date and time will be displayed in UTC format.
energyConsumptionEndDateTime	String	The date and time when the host stopped to consume power. The date and time will be displayed in UTC format.
systemPeakPower	String	The host peak power
systemPeakPowerStartDateTime	String	The date and time when the host peak power started. The date and time will be displayed in UTC format.
systemPeakPowerEndDateTime	String	The date and time when the host peak power ended. The date and time will be displayed in UTC format.
systemPeakAmps	String	The hosts peak amps
peakAmpsStartDateTime	String	The starting date and time of the host peak amps. The date and time will be displayed in UTC format.
peakAmpsEndDateTime	String	The end date and time of the host peak amps. The date and time will be displayed in UTC format.
instant	String	Instant power
peak	String	Peak power

Get host system event log

Table 131. Response body

Attribute	Type	Description
index	String	Serial number
status	String	Severity levels (Normal, Warning, and Critical)
timeStampString	String	Date with time for the specific event in UTC format
description	String	Event description
noOfRecords	String	Total number of system event logs

Get host storage details

Table 132. Response body

Attribute	Type	Description
enclosures	String	Enclosure ID
physicalDisks	String	Array of physical disks
virtualDisks	String	Array of virtual disks
controllerId	String	Controller ID
fqdd	String	Enclosure FQDD or physical disk FQDD or virtual Disk FQDD
firmwareVersion	String	Controller firmware version
minimumRequiredFirmwareVersion	String	Minimum required firmware version
driverVersion	String	Minimum driver version
cacheMemorySize	String	Cache memory size
patrolReadState	String	Patrol Read State
deviceId	String	Physical disk device ID or virtual disk device ID
name	String	Virtual disk name or virtual disk name or controller name
capacity	String	Physical disk capacity
partNum	String	Physical disk part number
vendorName	String	Vendor name
serialNum	String	Serial number
modelNum	String	Model number
diskStatus	String	Physical disk status
hotSpare	String	Hot spare type
busProtocol	String	Bus protocol
enclosureId	String	Enclosure ID
mediaType	String	Media Type (SSD or HDD or SAS)
connectorId	String	Connector ID
remainingRatedWriteEndurance	String	Remaining write endurance
size	String	Virtual disk device size
layout	String	Layout type of the virtual storage
defaultReadPolicy	String	The default read policy that is supported by the controller
defaultWritePolicy	String	The default write policy that is supported by the controller
diskCachePolicy	String	Cache policy if enabled
stripeSize	String	Stripe size

C

OMIVV-Specific error codes

Table 133. OMIVV-specific error codes

Code	Description
11501	Number of active sessions limit exceeded.
11502	Number of active client limit exceeded.
11503	Input user data is not valid to process the login request.
11504	Exception occurred in token generation.
11505	User is not authorized to process the login request.
11506	Not a valid API Action specified.
11507	Account is locked.
11508	Request limit per minute exceeded.
11509	Empty Action request. Connection refused.
11510	User is not authorized to process the login request. Account Locked.
11601	Invalid Token
11602	Invalid path
11603	Internal Server Error
11604	Public key not found
11605	Token has already expired
10001	vCenter is not registered with appliance.
10002	This instance of OpenManage Integration is already registered with vCenter.
10003	Make sure vCenter is reachable and services are running.
10004	Failed to login to vCenter due to incorrect username or password.
10005	Invalid Parameters. vCenter registration data is not proper.
10006	Invalid Parameter.Provide valid vCenter IP/Hostname.
10101	Invalid console id {0}
10102	No vCenter is registered with appliance for the given id {0}
10103	vCenter tree is empty for id {0}
10201	Invalid http header does not contain authorization token.
10202	Invalid http header is empty.
10203	Invalid Parameters. vCenter context request parameters is not proper.
10205	Failed to login to vCenter. Make sure vCenter is running and credentials are valid.

Table 133. OMIVV-specific error codes (continued)

Code	Description
10206	Invalid Parameters. No vCenter is registered with the appliance.
10301	Cluster does not exist.
10302	The cluster with name {0} and id {1} has been deleted or has zero hosts.
10303	The cluster with name {0} and id {1} has been deleted or has zero OMIVV managed hosts.
10304	The cluster with name {0} and id {1} is not associated with Cluster Profile.
10305	The {0} is invalid host-id.
10401	The requestinfo is null or request does not contain serverIP details.
10402	Unable to find the cluster details because cluster is not part of the selected vCenter: {0}
10601	Repository profile does not exist.
10603	The NFS share path is invalid. The supported format is host:/share/file or host:/share/folder
10604	Repository profile already exists.
10605	Repository profile name is invalid.
10606	Unable to access the NFS share specified. The supported format is host:/share/folder
10607	Authorization failed to access the remote share. Check the provided credentials and/or write permissions on the share.
10608	Profile name must be Dell Default Catalog for Global firmware repository.
10609	Validated MX stack Catalog online URI must be https://downloads.dell.com/catalog/ValidatedMXstack_Catalog.xml.gz
10610	Authorization failed to access the remote share. Check the provided credentials and/or write permissions on the share. The supported format is \\\\host\\share\\filename.xml or \\\\host\\share\\filename.gz
10611	Unable to access the NFS share specified.
10612	Unable to access the HTTP/HTTPS share {0}
10613	Unsupported Protocol
10614	Global Repository is of type FIRMWARE only.
10615	Repository type can not be changed.
10616	Profile name must be Validated MX stack Catalog
10617	Protocol should be HTTP for the specified URI.
10618	Protocol should be HTTPS for the specified URI.
10619	Name is limited to only 250 characters.
10620	Description is limited to only 400 characters.
10901	Cluster profile is not created.

Table 133. OMIVV-specific error codes (continued)

Code	Description
11001	Cluster profile does not exist.
11002	Drift detection Job does not exist.
11353	Inventory information is not available for the host : {0}
11352	The entered host ID {0} is invalid or the host is not part of host credential profile.
11354	Could not connect to host ID {0}
11355	Could not connect to iDRAC of host {0}
10701	Invalid repository profile {0}.
10702	The repository {0} is not downloaded or refreshed.
10703	An exception occurred while loading the repository {0} : {1}.
10704	Invalid bundleId {0}.
10705	Invalid systemId {0}.
10706	Driver repository profiles {0} are not supported.
10802	For vSAN-enabled cluster, the cluster or host level firmware inventory is not supported for Dell Default Catalog and Validated MX Stack Catalog firmware repository profiles.
10804	Unable to find the cluster details: {0}.
10808	Firmware inventory report at host level does not allow empty or null hostId value.
10809	System BundleID does not allow empty or null value.
10810	vCenter with id {0} not found in registered list.
10812	Unable to find the cluster details using: {0}.
10814	The host is not compliant because the Hypervisor status is non- compliant.
10815	Only HOST/CLUSTER firmware inventoryType is allowed.
10816	Firmware inventory report does not allow empty or null BundleAssociations value.
10817	SystemId does not allow empty or null value.
10818	Unable to find the applicable files components using BundleAssociations.
10820	Unable to find the HostName using hostAddress {0}.
10821	Firmware inventory report at cluster level does not allow empty or null clusterId.
10822	Unable to retrieved vcenter tree using vCenterAddress {0}.
10823	The host is not compliant because the host is not associated to Host Credential Profile.
10824	The host is not compliant because the CSIOR status is non-compliant.
10825	Unable to find the Host managementIP using hostAddress {0}.
11301	preCheck is not applicable for rebootOptions with values FORCEREBOOT or NEXTREBOOT.

Table 133. OMIVV-specific error codes (continued)

Code	Description
11302	enterMaintenanceModeOption, exitMaintenanceMode, migratePoweredOffAndSuspendedVMs, and enterMaintenanceModetimeout are not applicable for rebootOptions with values FORCEREBOOT or NEXTREBOOT.
11303	firmwareUpdateTargets cannot be null or empty.
11304	More than one firmwareUpdateTargets is not supported for HOST firmware update.
11305	Invalid job schedule.
11306	Invalid jobSpecificCustomConfiguration details : {0}.
11307	rebootOptions cannot be null or empty.
11308	Invalid jobSpecificCustomConfiguration details. Enter the enterMaintenanceModetimeout value between 60 to 1440 minutes.
11309	Invalid input. The value for preCheck should be true for cluster update.
11310	exitMaintenanceMode value should be true for cluster update.
11312	Invalid enterMaintenanceModeOption.
11313	enterMaintenanceModeOption is not applicable for non-vSAN host or cluster.
11315	packageIDs cannot be null or empty.
11316	Unsupported jobSpecificCustomConfigurations. Only 5 job specific configurations are supported.
11317	Duplicate Host id : {0}.
11318	Component is up to date with package id: {0}.
11319	Invalid packageId {0}.
11320	Invalid packageId.
11321	Invalid input preCheck is applicable only for host managed under cluster.
11322	Unable to find the host details in inventory for host : {0}.
11323	The host with id {0} is not associated to credential profile.
11324	The host is not compliant {0}.
11325	The following firmware update jobs are currently scheduled {0} for host/cluster {1}.
11326	Invalid dateTime: {0}.
11327	An exception occurred while setting date and time schedule : {0}.
11328	Empty or null value is not allowed for job name.
11329	The job name is too long. Job name allows only 255 characters.
11330	The entered job name already exists {0}.
11331	The job Description is too long. Job Description allows only 2000 characters.

Table 133. OMIVV-specific error codes (continued)

Code	Description
11332	Do not enter true or false or null values for both the runNow and runLater job schedules at a time.
11333	Schedule the job at least 30 minutes after the UTC current time: {0}.
11334	The dateTime values for job schedular should not be null or empty.
11335	The dateTime used for job schedular is invalid.
11336	The dateTime format for job schedular should be : yyyy-MM-dd'T'HH:mm:ssZ.
11337	Host {0} is not part of the cluster.
11338	All hosts in the cluster are non-compliant.
11339	The host is not compliant {0}.
11340	rebootOptions with NEXTREBOOT is not applicable for runLater.
11341	rebootOptions with NEXTREBOOT is applicable only for HOST level Update Job.
11342	rebootOptions with FORCEREBOOT is not applicable for runLater.
11343	rebootOptions with FORCEREBOOT is applicable only for HOST level Update Job.
11344	rebootOptions with FORCEREBOOT is applicable only for non-vSAN HOST level Update Job.
11345	Driver updates are not supported.
11346	bundleId cannot be null or empty.
11347	hostId cannot be null or empty.
11349	clusterId cannot be null or empty.
11350	Invalid clusterId {0}.
11351	For vSAN-enabled cluster, the cluster or host level update job is not supported for Dell Default Catalog and Validated MX Stack Catalog firmware repository profiles.
11451	Firmware update job is not present.
11452	Invalid Job id.
11453	Job id {0} not found.
11455	Job ID {0} is not valid to purge the job. The job must be in canceled, or successful, or failed state.
11456	Job id {0} is not a firmware update job.
11457	Task is already triggered to cancel the job {0}.
11458	The cancel job task is already completed for {0}. You cannot create the task to cancel the job for already completed task.
11701	Exception occured while retrieving firmware drift details.
11702	Exception occured while retrieving configuration drift details
11703	Exception occured while retrieving driver drift details

Table 133. OMIVV-specific error codes (continued)

Code	Description
11704	Firmware drift is not calculated for cluster {0}. Ensure drift detection job is completed successfully.
0	Unknown exception occured.
12501	Input request is not valid.
12502	Operational Context not set.
12503	Invalid path parameter.
12504	Invalid Parameters.
12505	vCenter for which the Operational Context was set got unregistered with the appliance.
12506	Could not find the request header information.
12507	Could not find the Bearer Authorization token.
12508	Invalid Server Context.
12509	Error in connecting to vCenter.
12510	vCenter User is not authorized to access this URI.
12511	URI not supported by OMIVV.
12512	vCenter ID {0} is not available in the registered list.
12001	Internal Server Error/Timeout occurred: {0}.
12002	Unable to schedule the firmware update job.
12003	Unable to get applicable files.
12004	Unable to get the applicable software files.
12005	Unable to get the applicable software bundles.
12006	An exception occurred while finding the vSAN cluster details.
12007	Unable to process.
12008	vCenter id {0} is not is not registered with appliance.
12009	Invalid vCenter details.
12010	An unknown exception was recorded in the logs.
13001	No licenses found.
13002	License with id {0} does not exist.
13003	The detected older license format is not supported in this version. To get an updated license, send an email to Download_Software@Dell.com with original order number and contact details.
13004	Invalid license issuer. Use the license issued by Dell and ensure that the license has not been modified.
13005	Installing an evaluation license is not permitted when a standard license exists. Upload only a standard license.
13006	Invalid signature on the license file. Upload a valid license file.
13007	Unable to parse license file. Ensure that the license is issued for OMIVV and has not been modified.
13008	Your license is expired. Upload a new license to use this feature.

Table 133. OMIVV-specific error codes (continued)

Code	Description
13009	License contains invalid values. To get an valid license, send an email to Download_Software@Dell.com with original order number and contact details.
13010	License already in use. Select a different license file, and try again.
13011	License is not for this application. Select a different license file, and try again.
13012	Parameters are empty. Enter proper input values.
13013	File path is not matching the mentioned sharetype parameter. Enter proper license file path.
13014	Unable to read the file.
13015	File content is empty. Upload a proper license file.
13016	Unable to access the file or file has insufficient permission. Enter proper license file and provide permission to read the file.
13017	Failed to authenticate the CIFS share. Enter credentials to read the file.
13018	Unable to access the file.
13019	Upload a valid XML license file.
13020	The domain name format is not correct. Enter domain name in proper format.
13021	The username format is not correct. Enter username in proper format.
13022	Unable to upload the license file because the provided license file is invalid, empty, or modified.
13023	Unable to access the NFS share.
14001	Parameters are empty. Enter proper input values.
14002	Invalid combination of alerts and events. Please enter valid combination of alerts and events.