REST API Guide–OpenManage Essentials

Abstract

This technical white paper provides assistance for integrating with OpenManage Essentials by using Representational State Transfer (REST) APIs. It provides examples of using REST to perform common tasks based on integration use cases with other products such as OpenManage Mobile, Repository Manager, and the Dell Software group portfolio.

This document is not intended to be an introduction to REST. There are several publicly available documents (for example, Fielding's dissertation and RESTful Web services books) that provide the necessary background information on REST.

October 2018

Dell EMC Technical White Paper

Revisions

Date	Description
March 2014	Initial release
September 2014	Additional features included with OpenManage Essentials version 2.0
September 2015	Revisions included with OpenManage Essentials version 2.1
July 2016	Additional features included with OpenManage Essentials version 2.2
June 2017	Additional features included with OpenManage Essentials version 2.3
January 2018	Additional features included with OpenManage Essentials version 2.4
October 2018	Additional features included with OpenManage Essentials version 2.5

Acknowledgements

This paper was produced by the following members of the Dell EMC storage engineering team:

Authors

Nikhil S — Software Senior Engineer in the Enterprise Systems Management programs

The information in this publication is provided "as is." Dell Inc. makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any software described in this publication requires an applicable software license.

Dell believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

Contents

Acronyms

Acronym	Expansion
REST	Representational State Transfer
RO	Resource Oriented
ROA	Resource Oriented Architecture
OME	OpenManage Essentials
OMM	OpenManage Mobile

1 Key integration concepts

This section covers key integration concepts that are applicable to all of the use cases that are addressed in the next section.

1.1 Client Integration Overview

The REST client makes standard HTTP(S) requests to the REST API end-point. Each request is sent using a HTTP verb (for example, PUT, GET, POST, DELETE, HEAD, and OPTIONS) and includes a message body in XML format. The response uses a standard HTTP status code.

The REST client can be developed in a wide variety of programming languages (for example, C#, Java, and so on) or scripting languages (for example, Python, Perl, and so on) that have basic support for HTTPS communication and are capable of constructing and parsing XML payloads.

1.2 Base URI

The base URI for a REST request must be in the following format:

```
https://<HOST_ADDRESS>:<PORT_NUM>/api/<PRODUCT_NAMESPACE>
```

The parameters used in the base URI are described in the following table:

Parameter	Description
HOST_ADDRESS	The network address of the system on which the product is installed. The network address could be a DNS host name or an IP address.
PORT_NUM	The port number that has been specifically opened on the web stack for API communication. This should be the standard port used for SSL communications or a port that is registered with Internet Assigned Numbers Authority (IANA) for that product.
PRODUCT_NAMESPACE	The registered namespace for that product. For example, `OME.svc`.

An example of a base URI for OpenManage Essentials is as follows:

https://192.168.0.1:2607/api/OME.svc/

The rest of the document will reference this example as <`BASE_URI`>.

1.3 Security

The REST services will be exposed only through HTTPS to ensure that the common threats associated with HTTP traffic are mitigated. Administrator will have the option of updating the SSL self-signed certificate with a customer-provided certificate (for example, by uploading a PKCS-12 certificate or by signing an application-generated CSR request).

1.4 Authentication mechanisms

There are several common schemes for enabling authentication of REST requests. The following is a summary of the most common schemes:

Basic authentication

The authorization header in the request has the base-64 encoding of the credentials (username and password). If the credentials are not provided, a 401 (Authorization Failure) error is returned. Because the encoding is weak, this mechanism is only supported when SSL/TLS is used for the transport.

Digest authentication

The authorization header in the request has a digest computed from the user credentials. The server verifies if the digest matches a digest it computes. This is more secure than basic authentication since the credentials are not encoded over the wire.

1.5 Resource addressing

Each resource that is managed needs to be uniquely addressable using a distinct URI. The URI syntax needs to be intuitive and indicative of any associative relationship with a parent resource.

There are several acceptable ways to address specific resources either as instances of specific resource classes or within an associated parent context.

1) Pattern: <`BASE_URI>/<resource class identifier>/<resource instance identifier>`

   Example: `<BASE_URI>/Devices/1234` (where, 1234 is the unique identifier for the device)

2) Pattern: <`BASE_URI>/<parent resource identifier>/<resource class identifier>`

   Example: `<BASE_URI>/DeviceGroups/5/Devices/1234/Firmware`

   (To retrieve all the firmware resources for a specific device)

1.6 Resource operations

The standard HTTP methods are used for performing create, retrieve, update, and delete operations on the resources. The mapping of the HTTP methods to operational semantics is described in the following table:

HTTP method	Description	Example

`GET`	Used to retrieve the resource representation. This method does not modify the resource across repeated invocations. The query parameters are appended to the URI to appropriately filter the resource instances.	`GET <BASE_URI>/Alerts?Severity=16`
		To retrieve only critical alerts.
		`GET <BASE_URI>/Alerts?Severity=16&Status=2`
		To retrieve only acknowledged critical alerts.

`POST`	Used to create a new instance of a resource or append to the existing resource reference.	`POST <BASE_URI>/Tasks/PowerControl` To create a new power control task. If the service generates the identifier, the resource identifier is returned in the Location response header.
`PUT`	Used to update a specific instance or create a specific resource instance with a specific identifier.	`PUT <BASE_URI>/Alerts/1234` The entity body has the new state representation. For example, changing the acknowledge flag to update the alert.
`DELETE`	Used to remove a specific resource. If the resource does not exist, success is returned.	`DELETE <BASE_URI>/Alerts/1234` Deletes a specific alert with an instance identifier 1234.
`HEAD`	Used to determine if a resource is present or to return meta-data corresponding to the resource.	`HEAD <BASE_URI>/Alerts` The meta-data information is populated using custom HTTP response header options. For example, X-dell-alerttypes: storage, LC X-dell-alertURL:/alerts/storage, /alerts/LC
`OPTIONS`	Used to determine the method-level support for a specific resource.	`OPTIONS <BASE_URI>/Alerts/1234` The Allow HTTP response header indicates which methods are permitted.

1.7 Security considerations

The OpenManage Essentials REST PUT and DELETE operations requires the user IDs used to run these commands have the ‘write’ access to the base directory where the OpenManage Essentials REST services are installed. If the ‘write’ permission on this directory is not enabled for the user IDs used to access the DELETE and PUT operations, an error is returned indicating that the access is “Unauthorized”.

Role-based access to REST API methods are subjected to following restrictions:

Is user local administrator?	Role	HTTP Method			
		GET	POST	PUT	DELETE
Yes	OmeAdministrators	Yes	Yes	Yes	Yes
	OmePowerUsers	Yes	Yes	Yes	Yes
	OmeUsers	Yes	No	No	No
No	OmeAdministrators	Yes	Yes	No	No
	OmePowerUsers	Yes	Yes	No	No
	OmeUsers	Yes	No*	No	No
***For Site Administrators not having local administrator privilege, the POST method is permitted if the specified operation is allowed by using the GUI.**					

1.8 Data filtering and sorting

Data filtering can be accomplished by associating queries with the GET request. In the simplest form, it is a set of parameter name/value pairs that are appended to the request. The following is an example of data filtering:

```
<BASE_URI>/Alerts?Severity=16&Status=2
```

(To get the list of critical acknowledged alerts 16: Critical Severity, 2: Status Acknowledged)

Data sorting can be achieved by providing column names and corresponding directions in sort clause. Sort priority is implicit in the ordering of the column names. The following is an example of data sorting:

```
<BASE_URI>/Alerts/$sort(column=Id,Status;direction=0,1)
```

(To get list of alerts sorted in ascending by Id and then in descending by Status)

1.9 Data pagination

Data pagination can be accomplished by providing additional information in the GET request corresponding to a chunk of data. The response returns information on the total number of entries that could be used to reset any page numbering if the data has changed since the last call.

```
<BASE_URI>/Alerts/$top=250
```

This query will return the first chunk of data corresponding to the top 250 records.

```
<BASE_URI>/Alerts/$skip=250&$top=250
```

This query will skip the first 250 records and return effectively the next 250 records.

Note: The pagination keywords (`$top` and `$skip`) are case-sensitive.

Data pagination may be combined with data filtering. However, the data pagination key-value pairs must appear first. For example, if you want to retrieve the top 100 devices with critical status, use the following command:

```
<BASE_URI>/Devices/$top=100?GlobalStatus=16
```

The devices with a health status identified by 16 (Critical) are returned. The number of devices returned could be less than the maximum of 100 that was requested.

1.10 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 service will indicate the supported preference in the response header. The following table includes a few examples of request headers. For an extensive list of request headers, see List of HTTP header fields.

Request Header	Description	Example
Accept	Format of the data that is requested by the client. This could be one or more values that are comma-separated.	Accept: `application/xml`
Accept-Encoding	Encoding scheme. For example, to compress data to save bandwidth.	
Accept-Language	Choice of language that can be requested by the client.	Accept-Language: en
x-dell-api-version	Version of the API that is requested by the client.	x-dell-api-version:1.1

1.11 Response codes

For synchronous operations, the server returns HTTP response codes 200 or 204 based on the request. For long-running operations, the server returns a status code of 202 along with a HTTP response header (Location), corresponding to the URI of the temporary resource that can be used to monitor the operation. The following table includes a few examples of response codes. For an extensive list of response codes, see List of HTTP status codes.

Request	Response Code
Success Codes	
GET	200 – OK with message body 204 – OK with no message body 206 – OK with partial message body
POST	201 – Resource created (operation complete) 202 – Resource accepted (operation pending)
PUT	202 – Accepted (operation pending) 204 – Success (operation complete)
DELETE	202 – Accepted (operation pending) 204 – Success (operation complete)
Failure Codes	
Invalid parameter	400 – Invalid parameter
Authorization	401 – Authorization failure
Permission denied	403 – Permission denied
Not found	404 – Resource not found
Invalid request method	405 – Invalid request method
Internal server error	500 – Internal server error
Service unavailable	503 – Service unavailable

1.12 Response headers

The following table includes a few examples of response headers. For an extensive list of response headers, see List of HTTP header fields.

Response Header	Description	Example
Content-Type	Specifies the format of the content that is returned by the server. If there are multiple formats that could be accepted in the client request (using the Accept header), the server chooses the appropriate supported format.	Content-Type: `application/xml; charset=utf-8`
Content-Encoding	Specifies the optional supported encoding format.	Content-Encoding: `gzip`
Last-Modified	Specifies the last time the server modified the representation of the resource.	Last-Modified: Thu, 02 Apr 2009 11:11:28 GMT
Allow	Specifies what operations are permitted on the resource instance.	Allow: PUT,DELETE
Location	Defines the final URI of the resource being created or a temporary URI that can be used to monitor progress for long-duration transactions.	Location: `<BASE_URI>/DiscoveryRange/1234`
x-dell-api-version	Returns the version of the APIs	x-dell-api-version: 1.1

2 OpenManage Essentials-Specific Resource Model

The following sub-sections represent a subset of the use cases that OpenManage Essentials supports. The REST API support and the operation support will be incrementally refined based on consumer feedback over multiple OpenManage Essentials releases. Also, see the data filtering or sorting and pagination sections for patterns that can be used for performing paged retrieval of large result sets in OpenManage Essentials (for example, alerts, and devices).

Also, the following sections have information about the minimal set of attributes required for data filtering and sorting operations. The filtering can be enabled by using a suffix such as:

```
<Resource URI>?<Attribute Name1>=<Value1>&<Attribute Name2>=<Value2>
```

2.1 Device Groups

This section describes URIs which deals with operations on device groups.

2.1.1 Access Device Groups

The following resource URI can be used to access groups in OpenManage Essentials:

<BASE_URI>/DeviceGroups

Operation: GET

Attributes (filtering): Id, Type, Name, RollupHealth, and DeviceCount

A GET operation on an URI retrieves the root group of the hierarchy (the ‘All Devices’ group). The hierarchy can be recursively retrieved by retrieving the child groups and the devices under that root group.

Note: Only system and custom groups can be returned (device group types 1 and 4 respectively).

The attributes for each group are described in the following table.

Attribute Name	Description
Id	The unique identifier for the group.
Type	The type enumeration for the group.
Name	The name of the group.
Description	Description of the group.
RollupHealth	The rollup health enumeration for the group.
DeviceCount	The total number of devices in the group.

The following resource URI can be used to retrieve the child groups of a specific group:

<BASE_URI>/DeviceGroups/<ID>/ChildGroups

The following resource URIs can be used to retrieve the devices associated with a specific group:

<BASE_URI>/DeviceGroups/<ID>/ChildDevices (to retrieve all the immediate child devices)

<BASE_URI>/DeviceGroups/<ID>/Devices (to retrieve all the leaf devices for the group by performing a recursive traversal of the hierarchy rooted at the group identified by <ID>)

The following resource URI can be used to retrieve information about a specific group:

<BASE_URI>/DeviceGroups/<ID>

This URI returns the group attributes information described in the earlier table.

The following resource URI can be used to retrieve health summary information for a specific group:

<BASE_URI>/DeviceGroups/<ID>/Summary

The attributes of the group health summary information are described in the following table.

Attribute Name	Description
DeviceCount	The total number of devices in the group.
CriticalCount	The number of devices in the group that are in a critical state.
WarningCount	The number of devices in the group that are in a warning state.
NormalCount	The number of devices in the group that are in a normal state.
UnknownCount	The number of devices in the group that are in an unknown state.

Rollup Health Enumeration

The rollup health enumeration values are defined in the following table:

Enum Value	Description
0	None – Health status is not available.
2	Unknown – Health status is unknown.
4	Normal – Health status is normal.
8	Warning – Health status is warning.
16	Critical – Health status is critical.

Device Group Type Enumeration

The device group type enumeration values are defined in the following table:

Enum Value	Description
-1	AllGroups
0	Unknown
1	System
2	Factory
4	User (custom group)
8	Query
16	Pseudo
64	Internal
128	Tasks
256	ComputePool

2.1.2 Create Devices Group

The following resource URI can be used to create device groups in OpenManage Essentials:

<BASE_URI>/DeviceGroups

Operation: POST

In this POST operation, user has to provide the name and description of the group to be created in the payload. Optionally, user can provide a list of device IDs to be added to the group in the payload.

Attribute Name	Description
Name	Name of the device group
Description	A short description of the device group
AddDevices	List of DeviceIDs to be added to the group

A sample payload is given here:

```
<RESTDeviceGroupEntry>
   <Name>Group 1</Name>
   <Description>Description of group</Description>
   <AddDevices>
         <int>1</int>
         <int>2</int>
         <int>3</int>
   </AddDevices>
</RESTDeviceGroupEntry>
```

After successful completion, a new group is added with specified parameters and the newly added group appears under ‘All Devices’ in the device tree. If either the group name is empty or a group with same name already exist, a corresponding error message is returned. User can add a group only under root node (All Devices). Creation of subgroup and query driven groups are not possible. The devices bearing the IDs provided (if any) in the payload will be added to the newly created group if they are already discovered in OME. If the user provides device IDs which are invalid, an error message is returned only for those IDs and remaining devices will be added to the group.

Only OmeAdministrators, OmePowerUsers, and OmeSiteAdministrators will have the permission to add new device groups. In case of insufficient permissions, the operation will fail and returns an error message.

2.1.3 Modify Devices Group

The following URI allows user to add devices to and remove devices from a custom group. The devices to be added to the group must be discovered in OME prior to adding to custom group. User can use this API to rename the group and change description of the group.

<BASE_URI>/DeviceGroups/{groupid}

Operation: PUT

For this operation, a user must enter group ID in URI and list of IDs of devices to be added or removed in the payload XML.

Attribute Name	Description
Name	Name of the device group
Description	A short description of the device group
AddDevices	List of DeviceIDs to be added to the group
RemoveDevices	List of DeviceIDs to be removed from the group

The payload for this PUT operation is as follows:

```
<RESTDeviceGroupEntry>
   <Name>Group 2</Name>
   <Description>This is new group description</Description>
   <AddDevices>
         <int>4</int>
         <int>5</int>
         <int>6</int>
   </AddDevices>
   <RemoveDevices>
         <int>1</int>
         <int>2</int>
         <int>3</int>
   </RemoveDevices>
</RESTDeviceGroupEntry>
```

User can add the devices already discovered in OME to a custom group. User cannot add devices to any other types of groups other than custom group. All devices meant to be added will be added to the specified group if devices bearing those IDs exist in OME. Those devices which are meant to be removed from the group will be removed from the group. If the user enters device IDs which are invalid, an error response will be returned only for those IDs and remaining devices will be added/removed to/from the group. If the fields for groupname/description is not empty then those values will overwrite current name and description of the specified group subjected to data validation. If a group other than the specified group has the mentioned group name then modify operation fails and an appropriate error will be thrown. User will not be able to modify query driven groups or groups having subgroups with this API.

Only OmeAdministrators and OmePowerUsers will have the permission to add devices to group provided they also have local admin privilege. In case of insufficient permissions, the operation will fail and returns an error message. Users not having local admin privilege will encounter a generic ‘Access is denied’ error.

2.1.4 Delete Devices Group

The following URI allows user to delete a custom device group.

<BASE_URI>/DeviceGroups/{groupid}

Operation: DELETE

This operation deletes the specified custom group if it exists in OME. If the specified group is not present in OME, a corresponding error is returned. Any groups other than custom groups cannot be deleted.

Only OmeAdministrators, OmePowerUsers, and OmeSiteAdministrators will have the permission to delete a custom device group, provided they also have the local admin privilege. In case of insufficient permissions, the operation will fail and returns an error message. Users not having local admin privilege will get a generic ‘Access is denied’ error.

2.2 Devices

The URI for accessing devices in OpenManage Essentials could be either based on the parent group resource or by directly accessing the devices. For example:

Devices by group (immediate child devices): **<BASE_URI>/DeviceGroups/<ID>/ChildDevices**

Devices by group (all devices): **<BASE_URI>/DeviceGroups/<ID>/Devices**

Devices by direct access: **<BASE_URI>/Devices**

Any of the URIs specified earlier will be referenced as the <DEVICE_BASE_URI> in the subsequent sections of this document.

Devices (sorted): **<BASE_URI>/Devices/$sort(column=<attribute name(s)>;direction=<sort direction(s)>)**

Operations supported: GET

Attributes (filtering): Id, Type, Name, DNSName, and GlobalStatus

Attributes (sorting): Name Asc (by default), Id, and GlobalStatus

The attributes associated with each device are described in the following table:

Attribute Name	Description
Id	The unique identifier for the device.
Type	The type enumeration for the device.
Name	The name of the device.
DNSName	The DNS name of the device.
iDRACName	The name of the iDRAC on the device.
SystemId	The system identifier of the device.
GlobalStatus	The global HW status of the device.
PowerStatus	The power status of the device.
ServiceTag	The Service Tag of the device.
NodeId	The specific node identifier rather than the Service Tag that is used to identify a device such as the PowerEdge FM120x4 sled.
LaunchURL	The URL for the portal page for the iDRAC, if the device is a server.
AssetTag	The asset tag of the device.
SystemModel	The model of the device.
ExpressServiceCode	The service code associated with the device.
DiscoveryTime	The time when the device was last discovered.
InventoryTime	The time when the device was last inventoried.
StatusTime	The time when the device was last statused.
OSName	The name of the installed operating system.
OSRevision	The version of the operating system.
NICS	A collection of NIC elements identifying all the NICs associated with the particular device.
IsIdrac	True or false indicating whether the device instance is representing an iDRAC device.

The type enumeration for the device (**Type**) is defined in the following table:

Enum Value	Description
2	Unclassified
4	Server

Enum Value	Description
8	Switch
9	Chassis
10	KVM
11	Printer
12	Tape
13	EMC
14	FC SWITCH
15	MD Storage
16	EqualLogic Group
18	PDU
19	UPS
20	Client
21	PowerEdge C
22	Compellent
23	NAS Appliance
24	Network Appliance
26	EqualLogic Member
28	VxRail
29	XC Series
30	MX Chassis

The type enumeration for the **GlobalStatus** is defined in the following table:

Enum Value	Description
0	None
2	Unknown
4	Normal
8	Warning
16	Critical

For sorting devices, enter attribute names and sort direction separated by commas in the corresponding place holders. Priority of sorting is implicit in ordering of attributes.

The enumerated values for the sort direction are described in the following table:

Enum Value	Description
0	Ascending - smallest/earliest first (Default)
1	Descending - largest/latest first

Sample URIs for sorting devices are described in the following table:

URI	Result
<BASE_URI>/Devices/$sort(column=Name,Id;direction=0,1)	Devices sorted by Name (Ascending) then by Id (Descending)
<BASE_URI>/Devices/$top=5&$sort(column=Name,Id;direction=0,1)	First 5 devices from the list of devices sorted by Name (Ascending) then by Id (Descending)
<BASE_URI>/Devices/$skip=10&$top=5&$sort(column=Name,Id;direction=0,1)	5 devices skipping first 10 from the list of devices sorted by Name (Ascending) then by Id (Descending)

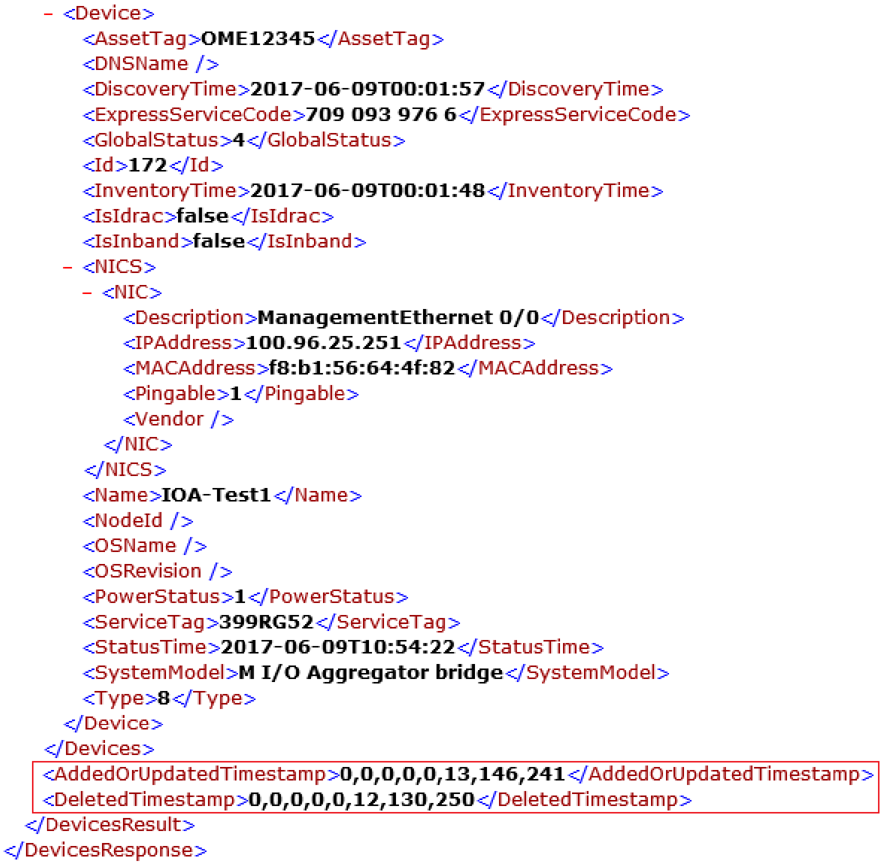
<BASE_URI>/Devices/$sort(column=Name,Id;direction=0,1)?GlobalStatus=2&Type=2	All Unknown and Unclassified Devices sorted by Name (Ascending) and then by Id (Descending)

2.2.1 Devices by direct access

<BASE_URI>/Devices

The resulting payload from this API is slightly different that the payload returned by the other device-related REST API calls. After all devices are enumerated in the resulting XML payload, two additional pieces of information are returned as well:

- AddedOrUpdatedTimestamp
- DeletedTimestamp



```
  - <Device>
      <AssetTag>OME12345</AssetTag>
      <DNSName />
      <DiscoveryTime>2017-06-09T00:01:57</DiscoveryTime>
      <ExpressServiceCode>709 093 976 6</ExpressServiceCode>
      <GlobalStatus>4</GlobalStatus>
      <Id>172</Id>
      <InventoryTime>2017-06-09T00:01:48</InventoryTime>
      <IsIdrac>false</IsIdrac>
      <IsInband>false</IsInband>
    - <NICS>
      - <NIC>
          <Description>ManagementEthernet 0/0</Description>
          <IPAddress>100.96.25.251</IPAddress>
          <MACAddress>f8:b1:56:64:4f:82</MACAddress>
          <Pingable>1</Pingable>
          <Vendor />
        </NIC>
      </NICS>
      <Name>IOA-Test1</Name>
      <NodeId />
      <OSName />
      <OSRevision />
      <PowerStatus>1</PowerStatus>
      <ServiceTag>399RG52</ServiceTag>
      <StatusTime>2017-06-09T10:54:22</StatusTime>
      <SystemModel>M I/O Aggregator bridge</SystemModel>
      <Type>8</Type>
    </Device>
  </Devices>
  <AddedOrUpdatedTimestamp>0,0,0,0,0,13,146,241</AddedOrUpdatedTimestamp>
  <DeletedTimestamp>0,0,0,0,0,12,130,250</DeletedTimestamp>
</DevicesResult>
</DevicesResponse>
```

Figure 1 Timestamps

The **AddedOrUpdatedTimestamp** and **DeletedTimestamp** denote the latest timestamps associated with the device payload just returned. These timestamps can be used to retrieve delta information by a consumer.

2.2.2 Delete Device

The following URI allows user to delete device from OME until it is discovered next time. This will remove the device from the device tree and it will not be shown under any group.

<BASE_URI>/Devices/{deviceid}

Operation: DELETE

Upon successful completion, the specified device will be deleted from OME. If the user provides devices ID which are invalid, an error response will be returned.

Only OmeAdministrators, OmePowerUsers, and OmeSiteAdministrators will have the permission to delete device from OME through REST, provided they also have local admin privilege. In case of insufficient permissions, the operation will fail and returns an error message. Users not having local admin privilege will get a generic ‘Access is denied’ error.

2.2.3 Changed device information

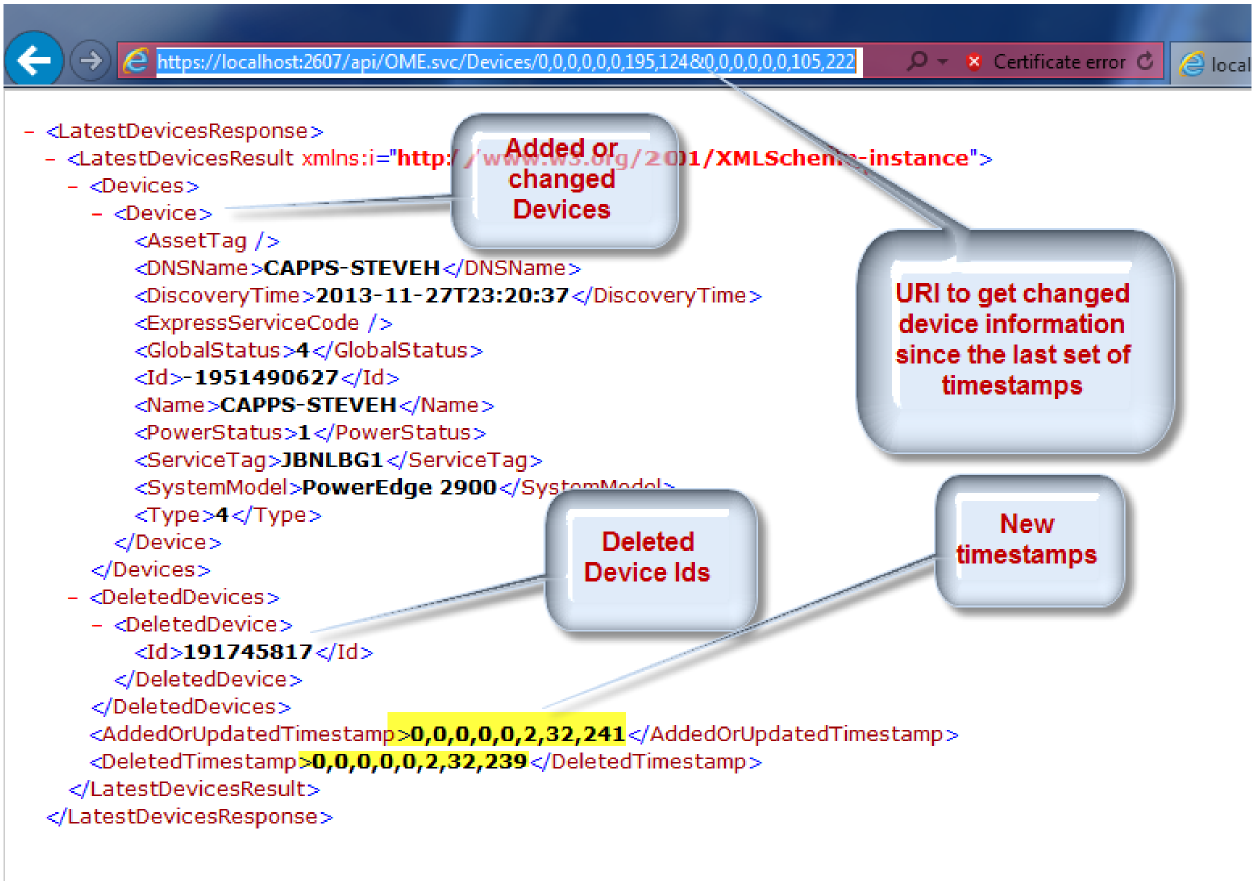
The changed device information can be obtained by the following REST API:

<BASE_URI>/Devices/{AddedOrUpdatedTimestamp}&{DeletedTimestamp}

The values of the `AddedOrUpdatedTimestamp` and `DeletedTimestamp` would have been obtained by a previous call to the **<BASE_URI>/Devices** REST API.

This particular REST API will typically be used by a consumer that does its own device management and tracking periodically based on a particular polling cycle. Instead of getting the whole list of devices to determine the changes, the consumer will be able to retrieve only the changed device information based on the timestamps obtained in a previous call.

For example, if the device listing was retrieved by an OpenManage Essentials REST consumer at time `t1` and the consumer needs to get information about the devices that were added or modified or deleted at time `t2` where `t2 > t1`, then the consumer, using the `AddedOrUpdatedTimestamp` and `DeletedTimestamp` obtained from the last call at `t1`, can get a list of only the changed and deleted devices.


https://localhost:2607/api/OME.svc/Devices/0,0,0,0,0,0,195,124&0,0,0,0,0,0,105,222
Certificate error
local
Added or changed Devices
URI to get changed device information since the last set of timestamps
Deleted Device Ids
New timestamps

```
- <LatestDevicesResponse>
  - <LatestDevicesResult xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    - <Devices>
      - <Device>
          <AssetTag />
          <DNSName>CAPPS-STEVEH</DNSName>
          <DiscoveryTime>2013-11-27T23:20:37</DiscoveryTime>
          <ExpressServiceCode />
          <GlobalStatus>4</GlobalStatus>
          <Id>-1951490627</Id>
          <Name>CAPPS-STEVEH</Name>
          <PowerStatus>1</PowerStatus>
          <ServiceTag>JBNLBG1</ServiceTag>
          <SystemModel>PowerEdge 2900</SystemModel>
          <Type>4</Type>
        </Device>
      </Devices>
    - <DeletedDevices>
      - <DeletedDevice>
          <Id>191745817</Id>
        </DeletedDevice>
      </DeletedDevices>
      <AddedOrUpdatedTimestamp>0,0,0,0,0,2,32,241</AddedOrUpdatedTimestamp>
      <DeletedTimestamp>0,0,0,0,0,2,32,239</DeletedTimestamp>
    </LatestDevicesResult>
  </LatestDevicesResponse>
```

Figure 2 Retrieving changed device information

2.3 Device inventory

The URIs described in this section can be used to retrieve more detailed inventory for each device. The DEVICE_BASE_URI is based on the parent group scope or the device scope.

Operations supported: `GET` (for all inventory operations)

2.3.1 Firmware

The following resource URI can be used to retrieve firmware information for a specific device identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/Firmware

The attributes associated with each firmware instance are described in the following table:

Attribute Name	Description
Type	The type of firmware.
Name	The name of the firmware.
Version	The version of the firmware.

2.3.2 Processor

The following resource URI can be used to retrieve information about the CPUs associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/Processor

The attributes associated with each processor instance are described in the following table:

Attribute Name	Description
MaxSpeed	The maximum speed of the processor.
CurSpeed	The current speed of the processor.
Cores	The number of cores in the processor.
Brand	The brand name of the processor.
Model	The model of the processor.

2.3.3 NICs

The following resource URI can be used to retrieve network interface information associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/NIC

The attributes associated with each NIC instance are described in the following table:

Attribute Name	Description
Description	The description of the NIC.
IPAddress	The IP address of the NIC.
MACAddress	The MAC address of the NIC.
Pingable	A numeric indicator for whether the device can be pinged or not. (1–In-band; 2–out-of-band)
Vendor	The NIC vendor.

2.3.4 Operating systems

The following resource URI can be used to retrieve operating system information associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/OS

The attributes associated with each operating system instance are described in the following table:

Attribute Name	Description
Type	The operating system type.
Name	The name of the operating system.
Revision	The operating system revision.
ServicePackVersion	The service pack version.

2.3.5 Memory

The following resource URI can be used to retrieve memory information associated with a specific device with identified by <`DEVICE_ID`>:

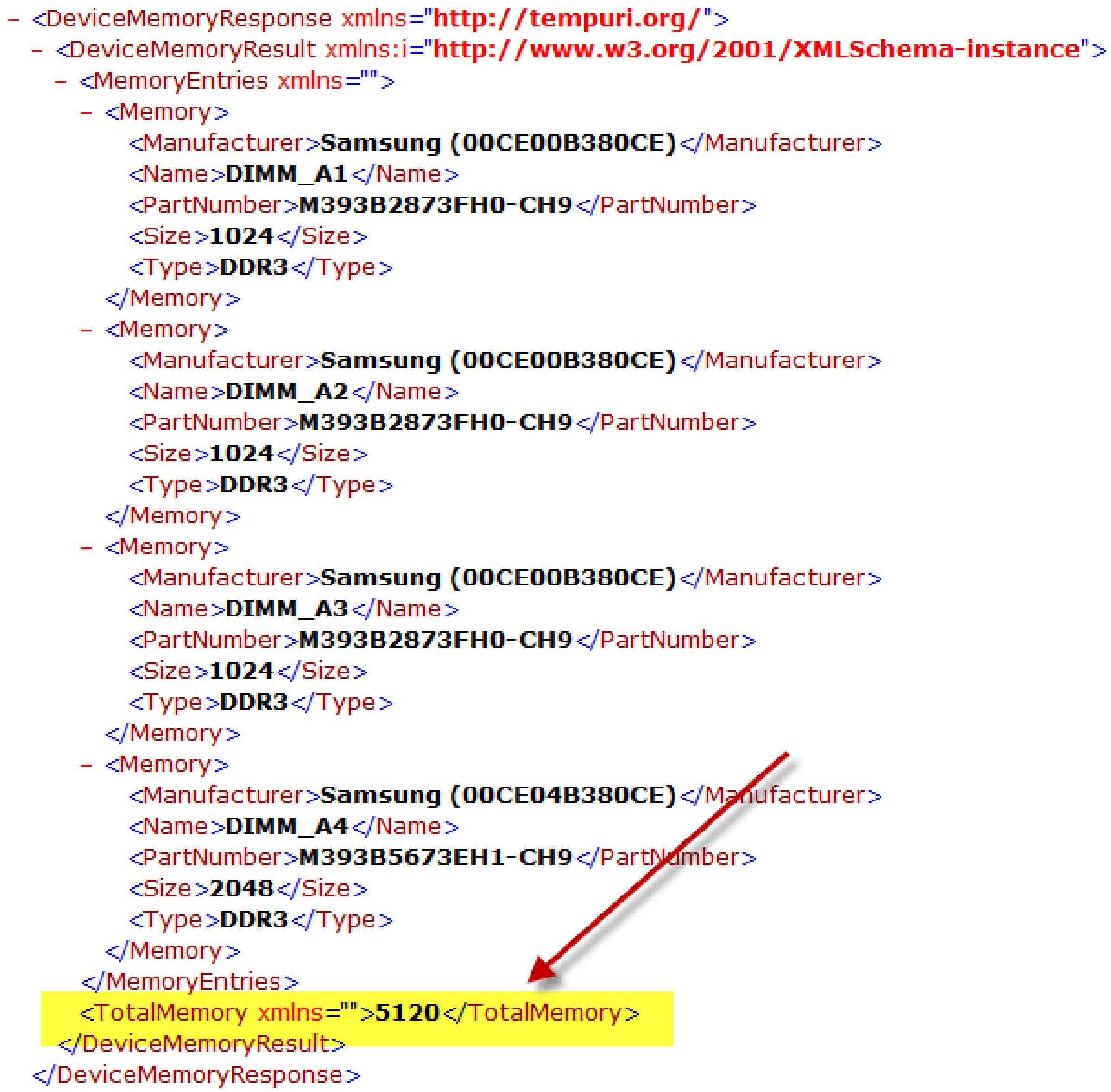
<DEVICE_BASE_URI>/<DEVICE_ID>/Memory

The attributes associated with each DIMM instance are described in the following table:

Attribute Name	Description
Size	The size of the DIMM in MB.
Type	The type of the DIMM.

Attribute Name	Description
Name	The name associated with the DIMM.
Manufacturer	The manufacturer of the DIMM.
PartNumber	The part number of the DIMM.

All DIMMS that are currently installed in the device will be returned. At the end of the envelope, the total memory installed will be returned. The following is an example of the returned XML payload:



```
- <DeviceMemoryResponse xmlns="http://tempuri.org/">
  - <DeviceMemoryResult xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    - <MemoryEntries xmlns="">
      - <Memory>
          <Manufacturer>Samsung (00CE00B380CE)</Manufacturer>
          <Name>DIMM_A1</Name>
          <PartNumber>M393B2873FH0-CH9</PartNumber>
          <Size>1024</Size>
          <Type>DDR3</Type>
        </Memory>
      - <Memory>
          <Manufacturer>Samsung (00CE00B380CE)</Manufacturer>
          <Name>DIMM_A2</Name>
          <PartNumber>M393B2873FH0-CH9</PartNumber>
          <Size>1024</Size>
          <Type>DDR3</Type>
        </Memory>
      - <Memory>
          <Manufacturer>Samsung (00CE00B380CE)</Manufacturer>
          <Name>DIMM_A3</Name>
          <PartNumber>M393B2873FH0-CH9</PartNumber>
          <Size>1024</Size>
          <Type>DDR3</Type>
        </Memory>
      - <Memory>
          <Manufacturer>Samsung (00CE04B380CE)</Manufacturer>
          <Name>DIMM_A4</Name>
          <PartNumber>M393B5673EH1-CH9</PartNumber>
          <Size>2048</Size>
          <Type>DDR3</Type>
        </Memory>
      </MemoryEntries>
      <TotalMemory xmlns="">5120</TotalMemory>
    </DeviceMemoryResult>
  </DeviceMemoryResponse>
```

Figure 3 Total memory

2.3.6 Software inventory

The following resource URI can be used to retrieve software inventory information associated with a specific device identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/Software

The attributes associated with each software entity instance are described in the following table:

Attribute Name	Description
Description	The description of the software entity.
Version	The version of the software entity.
Type	The type of the software entity.

The software type is described in the following table:

Type	Description
APAC	Application
APP	Systems management application
BIOS	BIOS
DRVR	Driver
FRMW	Firmware

2.3.7 Agent information

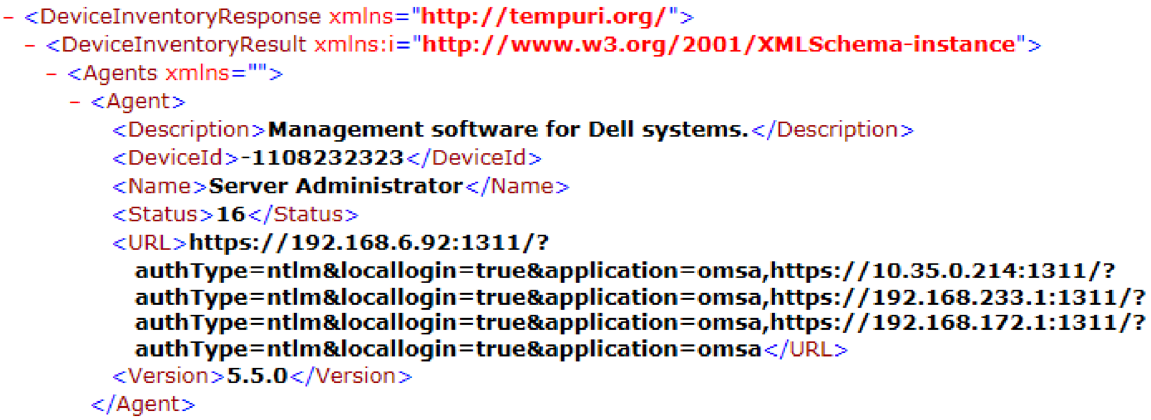
The following resource URI can be used to retrieve agent information associated with a specific device identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/Agents

The attributes associated with the agent information are described in the following table:

Attribute Name	Description
AgentId	The unique identifier for the agent
Description	The description of the agent.
DeviceId	The device identifier associated with the agent.
Name	The name of the agent.
Status	The latest status reported by the agent.
URL	The management URL of the agent.
Version	The version of the agent.

Note—In the case of an OpenManage Server Administrator (OMSA) agent, the value of the URL is a concatenated list of various URLs separated by the comma ',' character. The consumer of this API will have to tokenize this output and attempt a web connection to each of the URLs to determine which URL is responsive before surfacing the URL to the actual consumer. The following is an example of the URL for OMSA as returned in the REST XML payload:



```
- <DeviceInventoryResponse xmlns="http://tempuri.org/">
  - <DeviceInventoryResult xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    - <Agents xmlns="">
      - <Agent>
          <Description>Management software for Dell systems.</Description>
          <DeviceId>-1108232323</DeviceId>
          <Name>Server Administrator</Name>
          <Status>16</Status>
          <URL>https://192.168.6.92:1311/?
            authType=ntlm&locallogin=true&application=omsa,https://10.35.0.214:1311/?
            authType=ntlm&locallogin=true&application=omsa,https://192.168.233.1:1311/?
            authType=ntlm&locallogin=true&application=omsa,https://192.168.172.1:1311/?
            authType=ntlm&locallogin=true&application=omsa</URL>
          <Version>5.5.0</Version>
        </Agent>
```

Figure 4 Agent information

The values for the **Status** reported by the agent are defined in the following table:

Enum Value	Description
0	None – The status information is not available.
2	Unknown – The status is unknown.
4	Normal – The status is normal.
8	Warning – Warning status reported by the agent.
16	Critical – Critical status reported by the agent.

2.3.8 Contact information

The following resource URI can be used to retrieve contact information associated with a specific device identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/ContactInfo

The attributes associated with the contact information are described in the following table:

Attribute Name	Description
DeviceId	The device identifier associated with the contact information.
ContactName	The name of the contact associated with the device.
ContactInformation	The information about the contact associated with the device.
ContactDescription	The description of the contact associated with the device.

2.3.9 Device capabilities

The following resource URI can be used to retrieve the list of device capabilities associated with a specific device identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/DeviceCapabilities

The attributes associated with each device capability entity instance is described in the following table:

Attribute Name	Description
DeviceId	The device identifier.
DeviceCapabilityId	The identifier of the device capability.
DeviceCapabilityName	The name of the device capability.

The device capability values are defined in the following table:

Enum Value	Description
0	None
1	PowerCycle
2	PowerON
3	Shutdown
4	IPMICmdLine
5	BMC
6	RemoteAdministrator
7	RACADM
13	OOBPatch
14	iDRACMinVersion
15	iDRACLicense
16	iDRACInventory
17	NoOMSA
18	NoInvCollector
19	UnLicensedPlasma
20	LicensedPlasma
21	PlasmaPatch
22	PlasmaInventory
23	DellOEMServer
24	iDRACSixtyFourBit
25	DeviceConfigurationLicensed
26	DeviceConfigurationLicensable
27	iDRACMinVersionForISM
28	OSSixtyFourBit
29	FluidCache
30	ESXi
31	OOBAgent
32	InbandAgent
33	IOASupportsConfiguration
34	DellStorageNXtoNASApplicance
35	UnLicensedFX2
36	LicensedFX2
37	SwitchIsSupportedIoaModel
38	CmcSupportsXmlConfiguration
39	IoaSupportsVlanPortConfiguration
40	CmcDiscoveredWithWsMan
41	iDRACSupportsRedfishStreaming
42	NGMDiscoveredWithREST
43	NGMIsMCMDomainLead

2.3.10 Warranty information

The following resource URI can be used to retrieve the warranty information of a device identified by <DEVICE_ID>:

<DEVICE_BASE_URI>/<DEVICE_ID>/Warranty

Operations Supported: GET

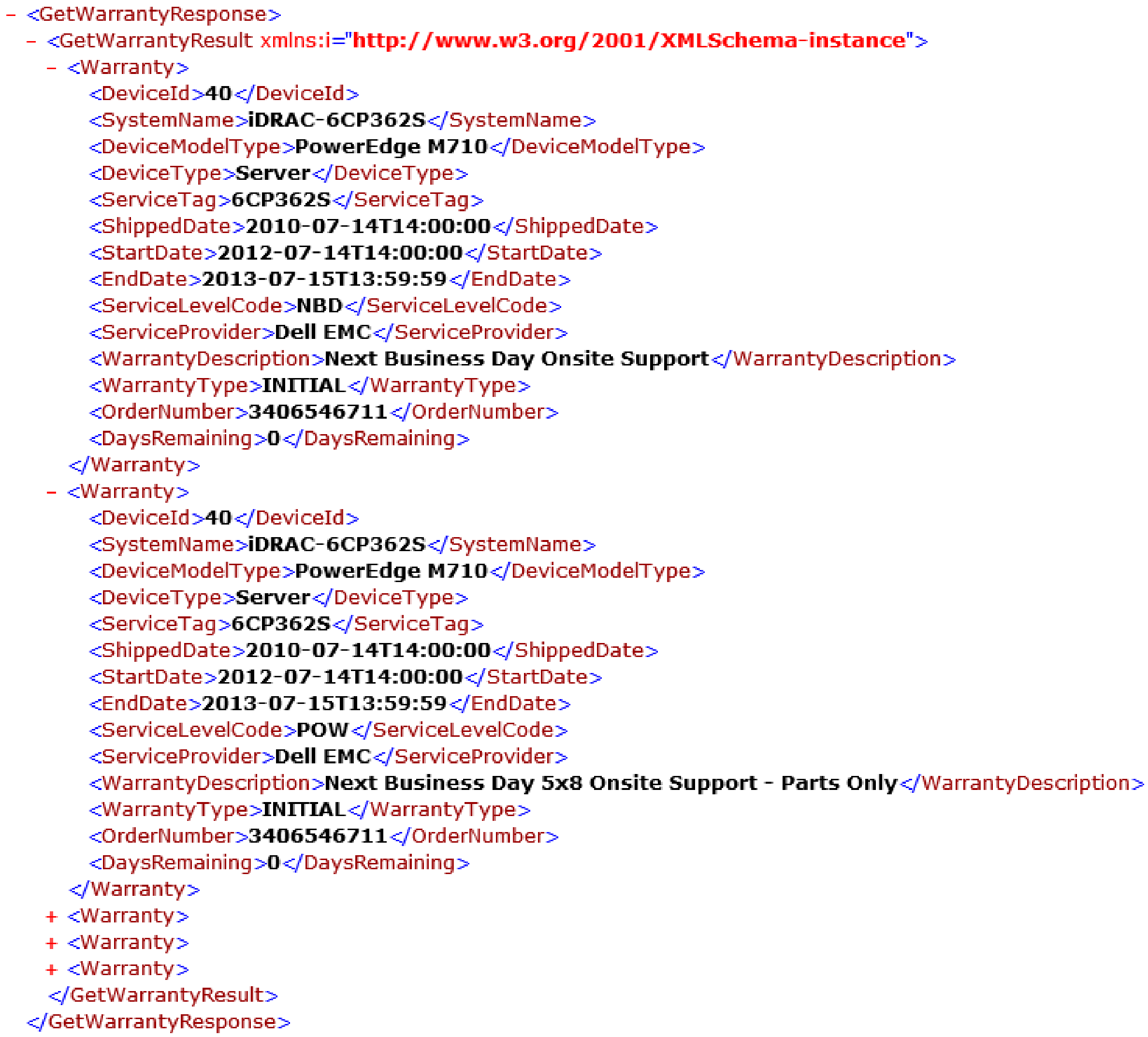
The warranty information is defined in the following table:

Attribute Name	Description
DeviceId	The device identifier
SystemName	Name of the device or system

DeviceModelType	Model type of the device
DeviceType	Type of the device
ServiceTag	Service Tag of the device
ShippedDate	The date the device originally shipped
StartDate	Date when the warranty started
EndDate	End date of the warranty
ServiceLevelCode	Code to describe what level of service the warranty provides
ServiceProvider	Name of the provider for the warranty
WarrantyDescription	Description of the warranty’s term of service
WarrantyType	Type of the warranty
OrderNumber	Order number of the warranty
DaysRemaining	Number of days remaining on the warranty

Each warranty record that is returned is per warranty and not per device. So if the warranty for a device is requested, there may be more than one warranty for that device. As a result more than one record may be returned.

The following is an example for service tag 6CP362S:



```
- <GetWarrantyResponse>
  - <GetWarrantyResult xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    - <Warranty>
        <DeviceId>40</DeviceId>
        <SystemName>iDRAC-6CP362S</SystemName>
        <DeviceModelType>PowerEdge M710</DeviceModelType>
        <DeviceType>Server</DeviceType>
        <ServiceTag>6CP362S</ServiceTag>
        <ShippedDate>2010-07-14T14:00:00</ShippedDate>
        <StartDate>2012-07-14T14:00:00</StartDate>
        <EndDate>2013-07-15T13:59:59</EndDate>
        <ServiceLevelCode>NBD</ServiceLevelCode>
        <ServiceProvider>Dell EMC</ServiceProvider>
        <WarrantyDescription>Next Business Day Onsite Support</WarrantyDescription>
        <WarrantyType>INITIAL</WarrantyType>
        <OrderNumber>3406546711</OrderNumber>
        <DaysRemaining>0</DaysRemaining>
      </Warranty>
    - <Warranty>
        <DeviceId>40</DeviceId>
        <SystemName>iDRAC-6CP362S</SystemName>
        <DeviceModelType>PowerEdge M710</DeviceModelType>
        <DeviceType>Server</DeviceType>
        <ServiceTag>6CP362S</ServiceTag>
        <ShippedDate>2010-07-14T14:00:00</ShippedDate>
        <StartDate>2012-07-14T14:00:00</StartDate>
        <EndDate>2013-07-15T13:59:59</EndDate>
        <ServiceLevelCode>POW</ServiceLevelCode>
        <ServiceProvider>Dell EMC</ServiceProvider>
        <WarrantyDescription>Next Business Day 5x8 Onsite Support - Parts Only</WarrantyDescription>
        <WarrantyType>INITIAL</WarrantyType>
        <OrderNumber>3406546711</OrderNumber>
        <DaysRemaining>0</DaysRemaining>
      </Warranty>
    + <Warranty>
    + <Warranty>
    + <Warranty>
    </GetWarrantyResult>
  </GetWarrantyResponse>
```

Figure 5 Warranty information

2.3.11 Physical disk drive

The following resource URI can be used to retrieve information about the Physical disk drives associated with a specific server identified by <DEVICE_ID>:

<DEVICE_BASE_URI>/<DEVICE_ID>/PhysicalDisk

The attributes associated with each physical drive instance are described in the following table:

Attribute Name	Description
Name	The name of the physical drive.
EnclosureID	ID of the enclosure.
Channel	Channel number.
TargetID	ID of the target.
LunID	LUN ID.
SizeInGB	Size of the disk (in GB).
BusType	Type of the bus.

Attribute Name	Description
SerialNumber	The serial number of the drive.
ModelNumber	The model number of the drive.
Revision	The revision number of the drive.
PartNumber	Part number.
MediaType	The type of the media.
Vendor	Vendor name of the drive.
RemainingRatedWriteEdurance	Remaining Rated Write Edurance
SupportedEncryptionTypes	Supported Encryption Types

2.3.12 Virtual drive

The following resource URI can be used to retrieve information about the Virtual drives associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/VirtualDisk

The attributes associated with each virtual drive instance are described in the following table:

Attribute Name	Description
Name	The name of the virtual drive.
DeviceName	The name of the device.
TargetID	ID of the target.
SizeInGB	Size of the disk (in GB).
LayOut	Lay-out.
CachePolicy	Cache Policy.
ReadPolicy	Read Policy.
StripeSizeInBytes	Stripe size (in Bytes).
WritePolicy	Write Policy.

2.3.13 Power supply unit (PSU)

The following resource URI can be used to retrieve information about the PSUs associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/PowerSupply

The attributes associated with each PSU instance are described in the following table:

Attribute Name	Description
Location	Location of the PSU.
OutputInWatts	Power supply output (in Watt).
Type	Type of the PSU.
PowerMonitoringCapable	Power monitoring capability.

2.3.14 Device card

The following resource URI can be used to retrieve information about the Device cards associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/DeviceCard

The attributes associated with each device card instance are described in the following table:

Attribute Name	Description
BusSpeed	The speed of the bus.

Attribute Name	Description
DataBusWidth	Width of the data bus.
Description	Description of the card.
Manufacturer	Name of the manufacturer.
SlotLength	Length of the slot.
SlotNumber	Slot number.
SlotType	Type of the slot.

2.3.15 Embedded device

The following resource URI can be used to retrieve information about the embedded devices associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/EmbeddedDevice

The attributes associated with each embedded device instance are described in the following table.

Attribute Name	Description
Description	Description of the device.
Manufacturer	Name of the manufacturer.

2.3.16 Enclosure slot

The following resource URI can be used to retrieve information about the Enclosure slots associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/EnclosureSlot

The attributes associated with each enclosure slot instance are described in the following table:

Attribute Name	Description
Slot	Enclosure slot number.
Subslot	Sub slot of the enclosure.
SlotName	Name of the slot.
SlotContent	Slot content.
ServiceTag	Service tag of the enclosure slot.
NodeID	ID of the node.
Model	Model number.
DeviceName	Name of the device.

2.3.17 Controller

The following resource URI can be used to retrieve information about the Controllers associated with a specific server identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/Controller

The attributes associated with each controller instance are described in the following table:

Attribute Name	Description
Name	The name of the controller.
Number	Controller number.
Type	Type of the controller.
FirmwareVersion	Firmware version of the controller.
ChannelCount	Channel count.
PhysicalDeviceCount	Physical device count.

Attribute Name	Description
LogicalDeviceCount	Logical device count.
CacheSizeInMB	Size of the cache (in MB).
MemorySize	Size of the memory.
Vendor	Vendor name.
CurrentMode	Current mode.

2.3.18 All inventory

The following resource URI can be used to retrieve the entire inventory associated with a specific device identified by <`DEVICE_ID`>:

<DEVICE_BASE_URI>/<DEVICE_ID>/Inventory

This URI would effectively return a concatenation of the individual components listed earlier (Firmware, Processor, NICs, Operating Systems, Memory, Software Inventory, Agents, `ContactInfo`, `DeviceCapabilities`, `WarrantyInfo`, Physical Disk, Virtual Disk, Power Supply, Device Card, Embedded Device, Enclosure Slot, and Controller). Use of this URI should be limited to scenarios which require all of this data to be displayed in a single view. If the inventory information is retrieved incrementally, one would use the individual URIs listed in the previous sections.

In addition to the components specified above, this command will return the following additional information:

Enclosures

There may be one or more enclosure entries. Each entry specifies the following attributes:

Attribute Name	Description
EnclosureProductId	The enclosure product ID.
EnclosureServiceTag	The Service Tag of the enclosure.
EnclosureId	The enclosure ID.

SwitchInfos

There may be one or more SwitchInfo entries. This section is applicable to switch devices only. Each entry specifies the following attributes:

Attributes	Description
Index	Indicates the number of units in the device.
SerialNumber	Hardware identifier.
SwitchRole	Indicates if the unit is a management unit, standby unit, stack unit, and local unit.

2.4 Table inventory

This is a generic API that returns the data of individual tables defined in OpenManage Essentials:

<DEVICE_BASE_URI>/TableInventory/{table id}

To request the table inventory for a specific device:

<DEVICE_BASE_URI>/{device id}/TableInventory/{table id}

The following table describes the table IDs.

Attribute Name	Description
1	Device
2	Agent
3	ArrayDisk
4	CmcSlot
5	ContactInfo
6	Controller
7	CostOfOwnership
8	DeviceCard
9	Enclosure
10	EnclosureManagementModule
11	Firmware
12	FruInformation
13	HyperVGuestNICView
14	HyperVHostGuestView
15	MaserInfo
16	MemoryDevice
17	ModularChassisAssociation
18	NIC
19	OperatingSystem
20	PDUDeviceProperty
21	PowerSupply
22	PrinterCover
23	PrinterInputTray
24	PrinterOutputTray
25	PrinterSupply
26	Processor
27	RACDevice
28	SoftwareInventory
29	StorageBattery
30	StorageGroup
31	SwitchDevice
32	TapeDrive
33	TapeLibrary
34	TPMInfo
35	UPSPhysicalBattery
36	VirtualDisk
37	VirtualFlash
38	VMDeviceInfoView
39	VMHostInfo
40	EventCatalog
41	EventCategory
42	EventSource
43	HyperVGuestInfo
44	HyperVGuestMemoryInfo
45	HyperVGuestNICInfo
46	HyperVHostInfo
47	DeviceGroup
48	DeviceGroupToDevice
49	NASApplianceNode
50	SoftwareInventoryOOB

2.5 Device Group Hierarchy

Some consumers of the OpenManage Essentials REST API services will require the capability of retrieving information that has changed since a particular point in time. The OpenManage Essentials server continuously updates its database with new device information, new hierarchical information, and information that has been deleted. OpenManage Essentials REST API consumers that want to implement a cache subsystem and need the capability of retrieving information that has changed (or deleted) may be able to use the following set of APIs.

The following URI can be used to retrieve the device group hierarchy currently managed by OpenManage Essentials:

<BASE_URI>/DeviceGroupHierarchy

Operations Supported: GET

The attributes associated with the returned payload are described in the following table:

Attribute Name	Description
HierarchyAddedOrUpdatedTimestamp	The last timestamp for the device group to device group hierarchy where an addition or update was performed.
HierarchyDeletedTimestamp	The last timestamp for the device group to device group hierarchy timestamp where a deletion was performed.
DeviceGroupAddedOrUpdatedTimestamp	The last timestamp for the device groups where an addition or updated was performed.
DeviceGroupDeletedTimestamp	The last timestamp for the device groups where a deletion was performed.
DeviceToGroupAddedOrUpdatedTimestamp	The last timestamp for the device to device group hierarchy where an addition or update was performed.
DeviceToGroupDeletedTimestamp	The last timestamp for the device groups where deletion was performed.
DeviceGroups	A collection of the device groups that have changed since the last specified `DeviceGroupAddedOrDeletedTimestamp`. Each entry of this collection corresponds to the definition in the “Device Groups” section in this document.
DeviceGroupToDeviceGroup	A collection of device group to device group relationships. This relationship is defined in a later table.
DeviceGroupToDevice	A collection of device group to device relationships. This relationship is defined in a later table.
DeletedDeviceGroupToDeviceGroup	A collection of integers that define the device group to device group identifiers that have been deleted since the last `HierarchyDeletedTimestamp` was specified.
DeletedDeviceGroups	A collection of integers that define the device group identifiers that have been deleted since the last `DeviceGroupDeletedTimestamp` was specified.
DeletedDeviceGroupToDevice	A collection of integers that define the device group to device identifiers that have been deleted since the last `DeviceToGroupDeletedTimestamp` was specified.

The attributes associated with the individual entries of the **DeviceGroupToDeviceGroup** section of the returned payload are described in the following table:

Attribute Name	Description
Id	The identifier of the relationship.
DeviceGroupId	The device group identifier of the child device group.
ParentDeviceGroupId	The device group identifier of the parent device group.

The attributes associated with the individual entries of the **DeviceGroupToDevice** section of the returned payload are described in the following table:

Attribute Name	Description
Id	The identifier of the relationship.
DeviceId	The device identifier of the child device.
ParentGroupId	The device group identifier of the parent device group.

2.5.1 Retrieving Device Hierarchy Information That Has Changed

To retrieve device hierarchy information that has changed, the **DeviceGroupHierarchy** GET API is used by providing the timestamps as part of the command construction:

```
<BASE_URI>/DeviceGroupHierarchy/{HierarchyAddedOrUpdatedTimestamp}&
{HierarchyDeletedTimestamp}&{ DeviceGroupAddedOrUpdatedTimestamp}&{
DeviceGroupDeletedTimestamp}&{ DeviceToGroupAddedOrUpdatedTimestamp}&{
DeviceToGroupDeletedTimestamp}
```

Operations Supported : GET

For example, to get the complete device hierarchy, the following API can be used:

```
https://<OME_SERVER>:2607/api/OME.svc/DeviceGroupHierarchy
```

To retrieve device hierarchy information that has changed use a command such as the following:

```
https://OME_SERVER>:2607/api/OME.svc/DeviceGroupHierarchy/0,0,0,0,0,2,32,205&0,0
,0,0,0,2,32,205&0,0,0,0,0,2,32,204&0,0,0,0,0,2,32,204&0,0,0,0,0,2,32,207&0,0,0,0
,0,2,32,207
```

2.6 Device Logs

The following resource URI can be used to retrieve the device logs:

<DEVICE_BASE_URI>/<DEVICE_ID>/HWLogs/SEL (to retrieve the SEL logs)

OpenManage Essentials may expose Lifecycle Controller logs in the future and that would require a separate URI construct as the attributes exposed are slightly different (for example, recommended action and detailed description).

Operations supported: GET

The attributes associated with each log entry are represented in the following table:

Attribute Name	Description
Severity	The severity of the log message.
Date	The date and time of the message.
Message	The message associated with the HW log entry.

2.7 Alert Filters

OpenManage Essentials alert filters enable the filtering of alerts by severity, time, event category and source, acknowledged status, and devices/device groups.

The following URI can be used to retrieve all alert filters:

<BASE_URI>/AlertFilters

Operations supported: GET

Attributes (filtering): Id, Type, and IsEnabled

Attributes (sorting): Name Asc (by default)

The attributes associated with an alert filter are described in the following table:

Attribute Name	Description
Id	The unique identifier for the filter.
Name	The name of the filter.
Type	The type of filter (for example, view filter, action filter, and so on).
IsEnabled	A flag indicating whether the filter is enabled or disabled.
IsReadOnly	A flag indicating whether the filter is read-only or editable.

To retrieve alerts associated with a specific alert filter, the following resource URI can be used:

The following URI enables filtering the alerts based on the criteria defined by the alert filter.

<**BASE_URI>/AlertFilters/<ID>/Alerts**

To get a summary by severity for each alert filter, the following URI can be used:

<BASE_URI>/AlertFilters/<ID>/Summary

The attributes associated with the alert filter are described in the following table:

Attribute Name	Description
TotalAlerts	The total number of alerts that match the filter criteria.
CriticalCount	The number if alerts with a critical severity.
WarningCount	The number of alerts with a warning severity.
NormalCount	The number of alerts with a normal severity.
InfoCount	The number of alerts with an informational severity.
UnknownCount	The number of alerts with an unknown severity.

2.8 Alerts

The resource URIs in this section can be used to return information on alert operations. Alerts can either be scoped based on the device or can be accessed globally.

Alerts (per filter) : **<BASE_URI>/AlertFilters/<ID>/Alerts**

Alerts (per device) : **<DEVICE_BASE_URI>/<ID>/Alerts** (where <ID> is the device identifier)

All alerts: **<BASE_URI>/Alerts**

Alerts (sorted): **<BASE_URI>/Alerts/$sort(column=<attribute name(s)>;direction=<sort direction(s)>)**

Operations supported: GET, DELETE, and PUT

Attributes (filtering): Id, Severity, and Status

Attributes (sorting): Timestamp Desc (by default), Id, Severity, Status, Message, and DeviceName

The attributes associated with the alerts are described in the following table:

Attribute Name	Description
Id	The unique identifier for the alert.
Severity	The enumerated severity of the alert.
Status	The status of the alert (for example, to indicate whether the alert is acknowledged or not).
Time	The date/time for the alert.
Message	The message corresponding to the alert.
EventCategory	The event category for the alert.
EventSource	The OpenManage Essentials alert source name for the alert.
DeviceIdentifier	The device identifier associated with the alert.
DeviceName	The name of the associated device.
SourceName	The name of the device associated with the alert when the alert was received (Note: The device may be deleted since the alert was received and the DeviceName will most likely be set to ‘Unknown’). If there was no device associated with the alert at the time the alert was received, this entry displays 0.0.0.0.
DeviceType	The device type of the device initiating the alert (string, user consumable).
DeviceTypeId	The device type id of the device initiating the alert (integer).
DeviceNodeId	The specific node identifier rather than the Service Tag that is used to identify a device such as the PowerEdge FM120x4 sled.
EnterpriseOID	The enterprise OID associated with the alert.
GenericOID	The generic OID associated with the alert.
SpecificOID	The specific OID associated with the alert.
Model	The model of the device that initiated the alert.
OS	The operating system running on the device.
Package	The service pack version.
ServiceTag	The Service Tag of the device that initiated the alert
IsIdrac	‘True’ or ‘False’. It signifies whether the device that generated the alert is an iDRAC device.

The enumerated values for the **Severity** field are described in the following table:

Enum Value	Description
1	Unknown
2	Informational
4	Normal
8	Warning
16	Critical

The enumerated values for the **Status** field are described in the following table:

Enum Value	Description
1	Not Acknowledged
2	Acknowledged

For sorting alerts enter attribute names and sort direction separated by commas in corresponding place holders. Priority of sorting is implicit in ordering of attributes. The enumerated values for the sort direction are described in the following table:

Enum Value	Description
0	Ascending - smallest/earliest first (Default)
1	Descending - largest/latest first

Sample URIs for sort are described in the following table:

URI	Result
<BASE_URI>/Alerts/$sort(column=Id,Status;direction=0,1)	Alerts sorted by Id (Ascending) and then by Status(Descending)
<BASE_URI>/Alerts/$top=5&$sort(column=Id,Status;direction=0,1)	First 5 alerts from the list of alerts sorted by Id (Ascending) and then by Status(Descending)
<BASE_URI>/Alerts/$skip=10&$top=5&$sort(column=Id,Status;direction=0,1)	5 alerts skipping first 10 from the list of alerts sorted by Id (Ascending) and then by Status(Descending)
<BASE_URI>/Alerts/$sort(column=Id,Status;direction=0,1)?Severity=8	All warning alerts sorted by Id (Ascending) and then by Status(Descending)

2.8.1 Delete Alerts

<BASE_URI>/Alerts/{id}{|{id}}

Operation: DELETE

To delete an alert use the standard HTTP DELETE operation.

2.8.2 Acknowledge Alerts

<BASE_URI>/Alerts/{id}{|{id}}

Operation: PUT

To acknowledge an alert or clear the acknowledgement use the standard HTTP PUT operation. For the content of the URL PUT request use plain text (text/plain) and send ‘status=clear’ or ‘status=acknowledge’ to update the status. Requesting an acknowledgement or clearing an acknowledgement can be requested regardless of the current status. The requested status will become the current status. Any other operations will be considered an error.

The requested action can be applied to a list of alerts by adding additional alert ids separated by a pipe (|).

The response will be in the following format:

```
<RESTResponse xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <Result>false</Result>
    <Messages>
        <RESTResponseMessage>
            <Id>56465</Id>
            <Message>Alert with id 56465 does not exist</Message>
        </RESTResponseMessage>
        <RESTResponseMessage>
            <Id>46541</Id>
            <Message>Alert with id 46541 does not exist</Message>
        </RESTResponseMessage>
    </Messages>
</RESTResponse>
```

If the requested action for all of the alerts in the list are successful, the return value will be true. No other information will be in the response.

If the requested action for any of the alerts in the list fails, the return value will be false. Messages for only actions that failed will be in the response. The ID of the alert and the error message will be reported.

Any alerts in the list for the requested action that are not in the response are successful.

2.8.3 Last alert IDs

<BASE_URI>/Alerts/LastAlertId

This API returns the ID of the last alert inserted in the OpenManage Essentials database.

2.8.4 Alerts since

<BASE_URI>/AlertsSince/{id}

This API returns the alerts since the identifier provided ({id}). This identifier will typically be obtained by a call to <BASE_URI>/Alerts/LastAlertId at some time in the past.

2.9 Alert subscriptions

Alert subscriptions provide an option to register or subscribe for alerts that match a filter. The subscription mechanism will allow notifications of alerts to external subscribers.

The following URI can be used to perform alert subscription operations:

Alert subscriptions: **<BASE_URI>/AlertSubscriptions**

Operations supported: GET, POST, PUT, and DELETE

Attributes (filtering): Id

The attributes associated with the alert subscription is described here:

Attribute Name	Description
Id	The unique identifier for the subscription.
IdSubscriberService	The enumerated type or identifier of the subscription service. This enables association with the correct notification mechanism.
UserName	The user name of the subscriber.
Description	The description of the subscriber (for example, mobile device model).
IsEnabled	A flag indicating whether this subscription is enabled or disabled.
AlertFilterId	The filter identifier associated with the subscription.
Token	A string that provides information about the subscriber (for example, address or device token).

The numbers for the various subscriber services are listed in the following table. As of OpenManage Essentials version 1.3, only Google Cloud is supported.

IdSubscriberService number	Description
1	Google Cloud

2.9.1 AlertSubscription GET command

To get the associated subscription that is assigned to a mobile device, you must specify the device ID that is associated to the particular device. The GET command suffix is as follows:

/AlertSubscriptions/<deviceId>

2.9.2 AlertSubscription DELETE command

Deletion of the subscription (or unsubscribe) is accomplished using the standard HTTP DELETE operation.

Note: For the DELETE operation, you have to provide the **DeviceId** value to the alert subscription that you want to delete.

2.9.3 AlertSubscription POST example Data

The payload for the POST command is an XML structure that defines the various fields that are needed to create an Alert Subscription.

Note: All the values in the payload are case sensitive.

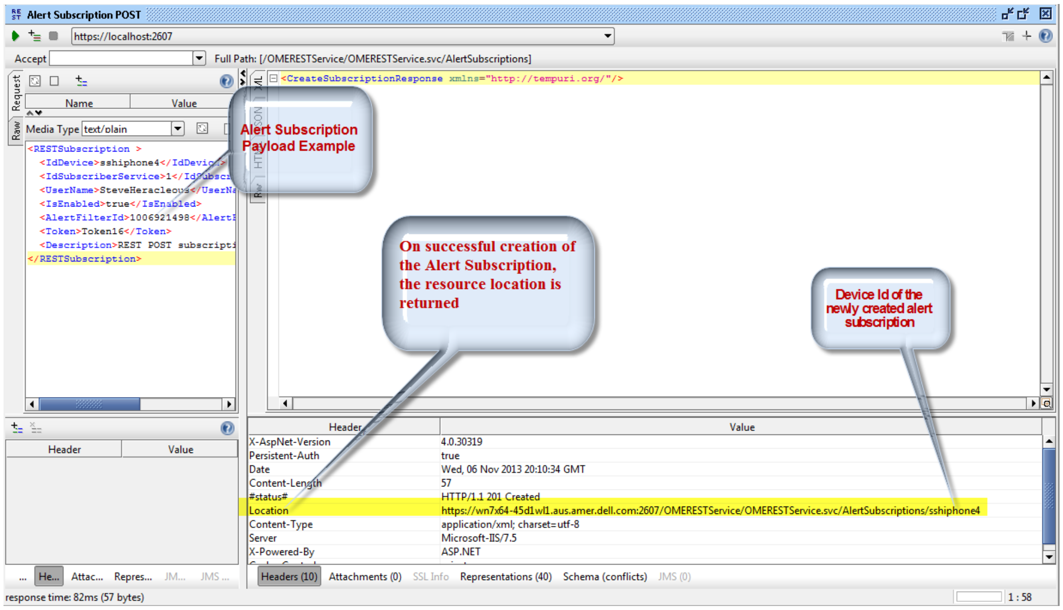
Of particular importance are Boolean values such as the <IsEnabled> included in the following example. This type of attribute takes the values of 'true' or 'false' in lower case. Any other value, including 'True' and 'False' will create an invalid data error when posting.

```
<RESTSubscription>
   <IdDevice>sshiphone4</IdDevice>
   <IdSubscriberService>1</IdSubscriberService>
   <UserName>SteveHeracleous</UserName>
   <IsEnabled>true</IsEnabled>
   <AlertFilterId>1006921498</AlertFilterId>
   <Token>Token16</Token>
   <Description>REST POST subscription creation</Description>
```

```
    </RESTSubscription>
```

Note: The field for Id is read-only and its value is only applicable on a GET operation. For the POST operation, this field is ignored; the recommendation is to set it to 0.

The following screenshot shows the result of executing an **AlertSubscription POST** command using SoapUI 4.5.0.


Alert Subscription POST
https://localhost:2607
Accept
Request
Raw
Name
Value
Media Type text/plain
XML
JSON
HTML
Raw
Alert Subscription Payload Example
On successful creation of the Alert Subscription, the resource location is returned
Device Id of the newly created alert subscription
... He... Attac... Repres... JM... JMS ...
Headers (10) Attachments (0) SSL Info Representations (40) Schema (conflicts) JMS (0)
1 : 58

Full Path: [/OMERESTService/OMERESTService.svc/AlertSubscriptions]

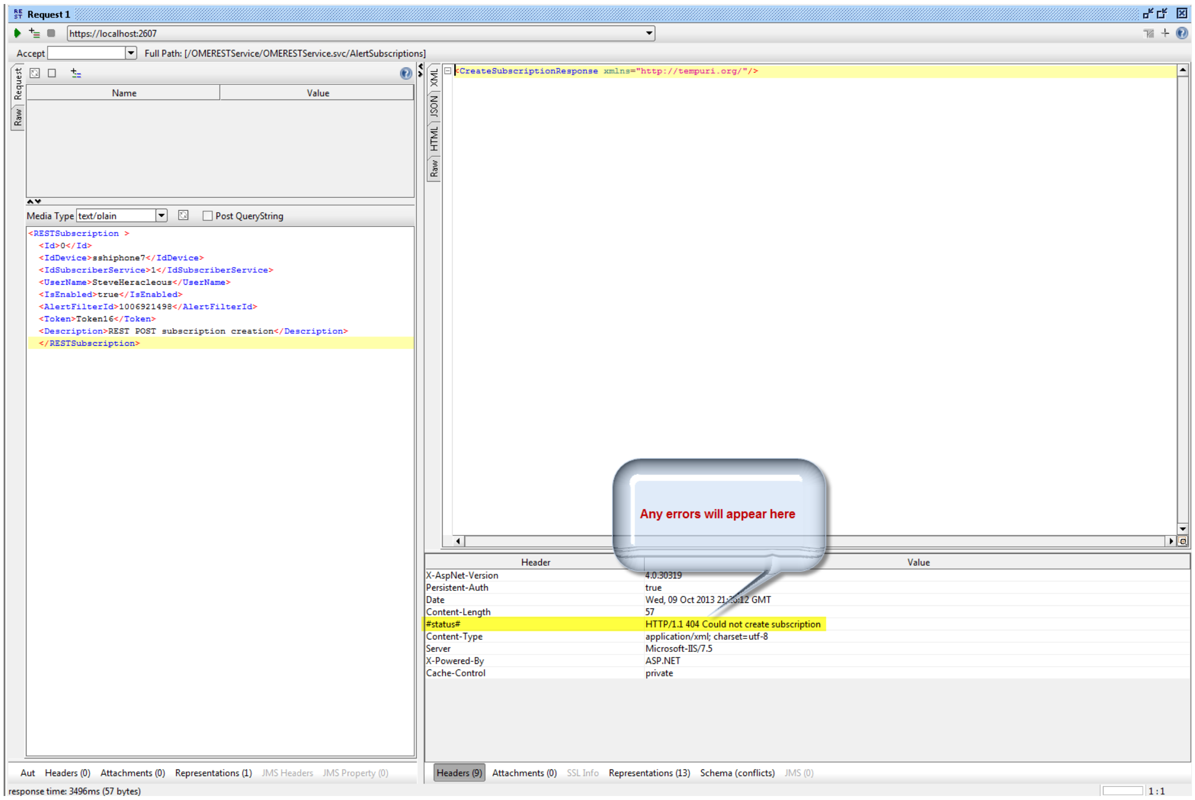
```
<RESTSubscription >
  <IdDevice>sshiphone4</IdDevice>
  <IdSubscriberService>1</IdSubscr
  <UserName>SteveHeracleous</UserNa
  <IsEnabled>true</IsEnabled>
  <AlertFilterId>1006921498</AlertF
  <Token>Token16</Token>
  <Description>REST POST subscripti
</RESTSubscription>
```

```
<CreateSubscriptionResponse xmlns="http://tempuri.org/"/>
```

Header	Value
X-AspNet-Version	4.0.30319
Persistent-Auth	true
Date	Wed, 06 Nov 2013 20:10:34 GMT
Content-Length	57
#status#	HTTP/1.1 201 Created
Location	https://wn7x64-45d1wl1.aus.amer.dell.com:2607/OMERESTService/OMERESTService.svc/AlertSubscriptions/sshiphone4
Content-Type	application/xml; charset=utf-8
Server	Microsoft-IIS/7.5
X-Powered-By	ASP.NET

response time: 82ms (57 bytes)

Figure 6 Alert subscription


Request 1
Accept
Request
Raw
Name
Value
Media Type text/plain
□ Post QueryString
XML
JSON
HTML
Raw
Aut Headers (0) Attachments (0) Representations (1) JMS Headers JMS Property (0)
Headers (9) Attachments (0) SSL Info Representations (13) Schema (conflicts) JMS (0)
1 : 1

https://localhost:2607

Full Path: [/OMERESTService/OMERESTService.svc/AlertSubscriptions]

```
<RESTSubscription >
  <Id>0</Id>
  <IdDevice>sshiphone7</IdDevice>
  <IdSubscriberService>1</IdSubscriberService>
  <UserName>SteveHeracleous</UserName>
  <IsEnabled>true</IsEnabled>
  <AlertFilterId>1006921498</AlertFilterId>
  <Token>Token16</Token>
  <Description>REST POST subscription creation</Description>
  </RESTSubscription>
```

```
<CreateSubscriptionResponse xmlns="http://tempuri.org/"/>
```

Any errors will appear here

Header	Value
X-AspNet-Version	4.0.30319
Persistent-Auth	true
Date	Wed, 09 Oct 2013 21:26:12 GMT
Content-Length	57
#status#	HTTP/1.1 404 Could not create subscription
Content-Type	application/xml; charset=utf-8
Server	Microsoft-IIS/7.5
X-Powered-By	ASP.NET
Cache-Control	private

response time: 3496ms (57 bytes)

Figure 7 Error

2.9.4 AlertSubscription PUT command

An update of the subscription information can be accomplished with a PUT operation.

Note: The payload for the PUT operation is identical to the payload of the POST operation.
Note: The subscription corresponding to the device identifier will be updated (if it exists). If the subscription with the corresponding device identifier does not exist, then the operation will return an error in the response header

2.10 Alert subscription notification

The alert subscription notification enables the mobile user to see the latest status of a notification that is associated with a subscription. This enables the administrator to sync his status with the status that is perceived by the Alert Subscription Manager in OpenManage Essentials.

The following resource URI supports alert subscription notifications:

Alert subscriptions: **<BASE_URI>/AlertSubscriptionNotifications**

Operations supported: GET and PUT

Attributes (filtering): Id, SubscriptionId, and AlertFilterId

The attributes associated with the alert subscription notification are described in the following table.

Attribute Name	Description
Id	The unique identifier for the subscription notification.
SubscriptionId	The subscription identifier.

AlertFilterId	The filter identifier associated with the subscription.
NewAlertCount	The number of new alerts associated with this subscription notification.
LastEventId	The last event identifier that is associated with the notification.
NotificationTimestamp	The timestamp associated with the last time data was pushed.

The PUT operation clears the NewAlertCount to 0 indicating to the Alert Subscription Manager in OpenManage Essentials that the data has been read by the mobile device.

2.12 Tasks

The following URI can be used to create a new task:

Task URI: **< BASE_URI>/Tasks**

Operations supported: GET and POST

Attributes (filtering): TaskId

2.12.1 Power control task

The following resource URI can be used to create a power control task.

URI: **< BASE_URI>/Tasks/PowerControl**

Operations: POST

The attributes associated with the power control task are described in the following table:

Attribute Name	Description
Id	Always set this to 0 when creating a new task.
Name	The name of the task.
Description	A short description of the task.
ControlType	An enumerated value indicating the type of power control (Shutdown, Power Cycle, Reset, or Power On)
IsGraceful	A flag indicating if the power control results in a graceful operating system shutdown or not: 1 – Shutdown the operating system first 0 – Do not shutdown the operating system first Note: The IsGraceful property is applicable for task control types of 1, 2, 3 (Shutdown, PowerCycle, and Reset) and is ignored for control type of 4 (PowerOn). For more information, see the "**Error! Reference source not found.**" section.
Devices	The list of selected device identifiers. If only one device is affected, the list will contain one element. The device identifier is the identifier associated with the device.
Schedule	The schedule for the task. This will be in cron format. If it is a run now task, the schedule will have a special value ("-1").
UserName	The username required to run the task.
Password	The password required to run the task.

For more information about the format of the Schedule field, see cron.

A sample XML that would comprise the payload when initiating a POST operation to create a power control task is as follows:

```
<RESTPowerControlTaskW>
   <Id>0</Id>
   <Name>Power Off - REST</Name>
   <Description>Power Off Task</Description>
   <Schedule>46 22 */1 * * *</Schedule>
   <ControlType>1</ControlType>
   <Devices>
         <int>1</int>
         <int>2</int>
         <int>3</int>
         <int>4</int>
   </Devices>
   <UserName>Administrator</UserName>
   <Password>Dell1234</Password>
   <IsGraceFul>1</IsGraceFul>
</RESTPowerControlTaskW>
```

The attributes associated with the power control type enumeration are defined in the following table.

Enum Value	Description
0	None
1	Shutdown
2	PowerCycle
3	Reset
4	PowerOn

2.12.2 LED control task

LED control task can be utilized for controlling system LED on iDRAC. The following resource URI can be used to create a LED control task:

URI: **< BASE_URI>/Tasks/LEDControl**

Operations: POST

The attributes associated with the LED control task are described in the following table:

Attribute Name	Description
Id	Always set this to 0 when creating a new task.
Name	The name of the task.
Description	A short description of the task.
ControlType	An enumerated value indicating the type of LED control (Start LED, Stop LED)
Devices	The list of selected device identifiers. If only one device is affected, the list will contain one element. The device identifier is the identifier associated with the device.
Schedule	The schedule for the task. This will be in cron format. If it is a run now task, the schedule will have a special value (“-1”).
UserName	The username required to run the task.
Password	The password required to run the task.

A sample XML that would comprise the payload when initiating a POST operation to create a LED control task is as follows:

```
<RESTLEDControlTask>
   <Id>0</Id>
   <Name>Blink LED</Name>
   <Description>Start blinking LED of 1,2,3</Description>
   <Schedule>-1</Schedule>
   <ControlType>1</ControlType>
   <Devices>
         <int>1</int>
         <int>2</int>
         <int>3</int>
   </Devices>
   <UserName>username</UserName>
   <Password>password</Password>
</RESTLEDControlTask>
```

The attributes associated with the LED control type enumeration are defined in the following table.

Enum Value	Description
0	Stop blinking LED
1	Start blinking LED

2.12.3 Task Start Time consideration

If the schedule field is set as part of the payload, then particular attention needs to be paid to the repeating time field of the schedule. For example, if the schedule is specified as 46 22 */1 * * *, it would translate as follows:

46	: minute
22	: hour
*/1	: day of month
*	: weekly
*	: monthly
*	: yearly

Therefore, the example translates to executing the task every day at 22:46 UTC. What this means is that particular attention needs to be paid to the locale where this REST command is executed and the necessary conversion to the hour field needs to be performed (conversion to UTC according to the locale) so that the right execution hour is persisted in the database.

2.12.4 Task status and progress Information

<SPECIFIC_TASK_URI> =<BASE_URI>/Tasks/<SPECIFIC_TASK_ID>

The following resource URI can be used to query the task execution summary:
<SPECIFIC_TASK_URI>/ExecutionSummary

Operations supported: GET

The attributes associated with the task status and progress information are described in the following table.

Attribute Name	Description
StartTime	The start time of the task.
EndTime	The end time of the task.
TaskState	The enumerated state of the task.
PercentComplete	The completion status of the task in percent.
LastUpdated	The time at which the task was last updated.

The values of the **TaskState** are defined in the following table.

Enum Value	Description
1	Pending
2	Running
3	Stopped
4	Complete
5	Failed
6	Scheduled
7	Not Scheduled

The details for task execution on multiple devices can be queried by

<SPECIFIC_TASK_URI>/ExecutionDetails

The attributes associated with task execution on multiple devices are described in the following table.

Attribute Name	Description
StartTime	The start time of the task for the specific device.
EndTime	The end time of the task for the specific device.
Status	The status enumeration of the task for this device.
DeviceId	The device identifier for the task.
Summary	The summary of the task execution for the specific device.
Details	The details of the task execution for the specific device.

The device **Status** enumeration is defined in the following table.

Enum Value	Description
0	In progress
1	Pending
2	Running
4	Complete
8	Warning
16	Failed
32	Stopped

2.13 Application information

The following resource URI can be used to access information about the OME application:
<BASE_URI>/Application

Operations supported: GET

The attributes associated with the application information are described in the following table:

Attribute Name	Description
Name	The name of the application.
Description	Description of the application.
Vendor	The vendor of the application.
Version	The software version.
Build Number	The build number.
URL	The URL for the web-based management application.
GUID	A unique identifier for the OpenManage Essentials instance.

2.14 Current user information

The following resource URI can be used to access information about the currently authenticated user:

<BASE_URI>/CurrentUser

Operations supported: GET

The attributes associated with the current user information is described in the following table.

Attribute Name	Description
UserName	The name of the currently authenticated user
UserType	The type of the user

The type of user is defined in the following table.

User Type	Description
0	Not defined (unknown)
1	OMEAdministrator
2	OMEDeveloper
3	OMEGuest
4	OMEUser
5	OMEPowerUser
6	OMESiteAdministrator

2.15 User permissions

The following resource URI can be used for accessing the user permissions of the currently logged-in user:

<BASE_URI>/UserPermissions/{SID of user}

This is typically an API that will be used by third party consumers programmatically. The {SID} must be provided as parameter.

This API returns the permissions associated with the loggedin user ID. The permissions are represented as a collection of integers.

2.16 Reports

The currently available reports in OpenManage Essentials are provided through the REST API. These reports are identical to the reports found in OpenManage Essentials, including column names and sort order.

2.16.1 Report types

The following resource URI can be used for accessing a list of currently available reports:

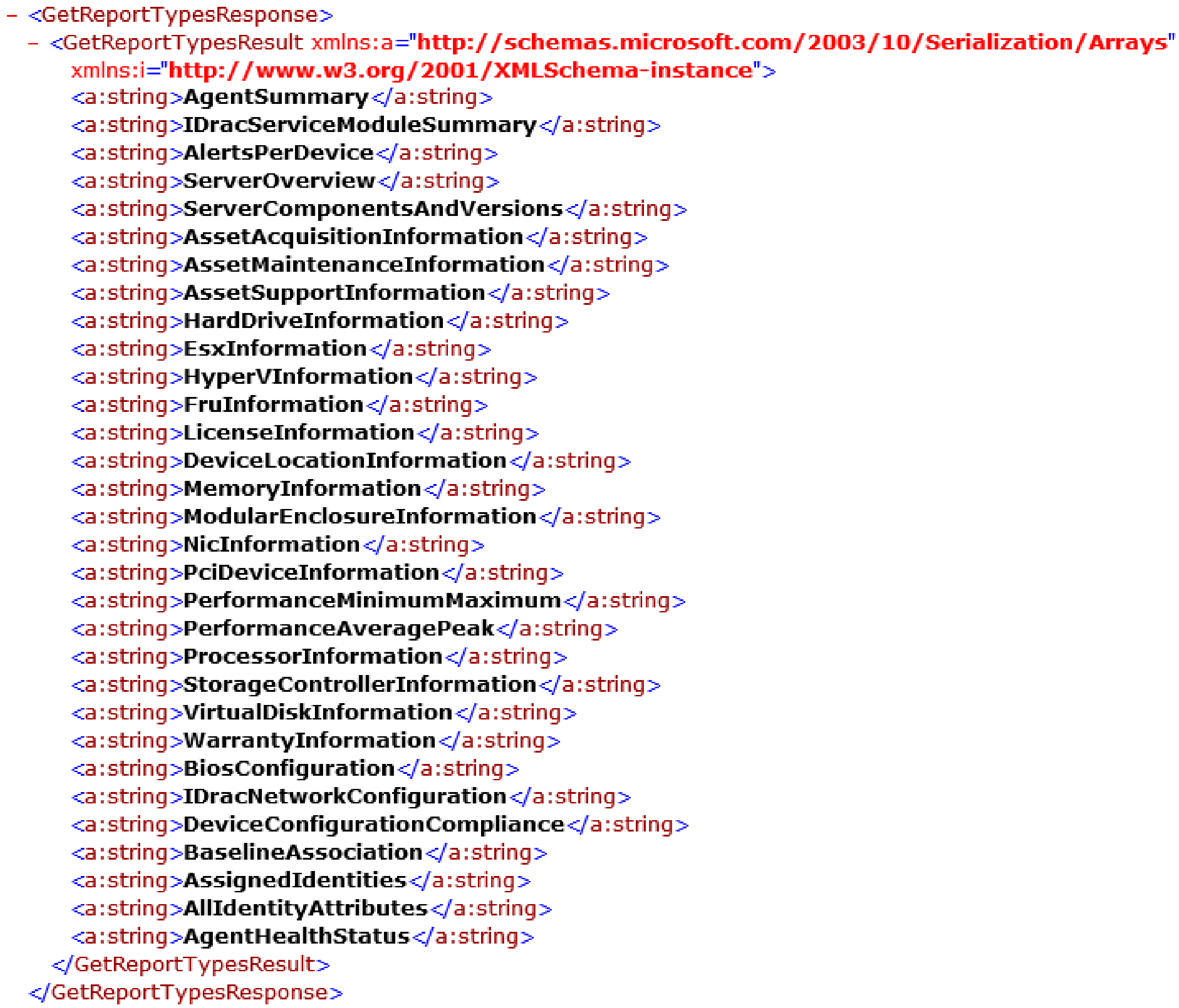
<BASE_URI>/Reports

Operations supported: GET

The types of reports are defined in the following table:

Report ID	Report Type
1	AgentSummary
2	IDracServiceModuleSummary
3	AlertsPerDevice
4	ServerOverview
5	ServerComponentsAndVersions
6	AssetAcquisitionInformation
7	AssetMaintenanceInformation
8	AssetSupportInformation
9	HardDriveInformation
10	EsxInformation
11	HyperVInformation
14	FruInformation
15	LicenseInformation
16	DeviceLocationInformation
17	MemoryInformation
18	ModularEnclosureInformation
19	NicInformation
20	PciDeviceInformation
21	PerformanceMinimumMaximum
22	PerformanceAveragePeak
23	ProcessorInformation
24	StorageControllerInformation
25	VirtualDiskInformation
26	WarrantyInformation
27	BiosConfiguration
28	IDracNetworkConfiguration
29	DeviceConfigurationCompliance
30	BaselineAssociation
31	AssignedIdentities
32	AllIdentityAttributes
33	AgentHealthStatus

The format of the XML output with the report types is as follows:



```
- <GetReportTypesResponse>
  - <GetReportTypesResult xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
      xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
      <a:string>AgentSummary</a:string>
      <a:string>IDracServiceModuleSummary</a:string>
      <a:string>AlertsPerDevice</a:string>
      <a:string>ServerOverview</a:string>
      <a:string>ServerComponentsAndVersions</a:string>
      <a:string>AssetAcquisitionInformation</a:string>
      <a:string>AssetMaintenanceInformation</a:string>
      <a:string>AssetSupportInformation</a:string>
      <a:string>HardDriveInformation</a:string>
      <a:string>EsxInformation</a:string>
      <a:string>HyperVInformation</a:string>
      <a:string>FruInformation</a:string>
      <a:string>LicenseInformation</a:string>
      <a:string>DeviceLocationInformation</a:string>
      <a:string>MemoryInformation</a:string>
      <a:string>ModularEnclosureInformation</a:string>
      <a:string>NicInformation</a:string>
      <a:string>PciDeviceInformation</a:string>
      <a:string>PerformanceMinimumMaximum</a:string>
      <a:string>PerformanceAveragePeak</a:string>
      <a:string>ProcessorInformation</a:string>
      <a:string>StorageControllerInformation</a:string>
      <a:string>VirtualDiskInformation</a:string>
      <a:string>WarrantyInformation</a:string>
      <a:string>BiosConfiguration</a:string>
      <a:string>IDracNetworkConfiguration</a:string>
      <a:string>DeviceConfigurationCompliance</a:string>
      <a:string>BaselineAssociation</a:string>
      <a:string>AssignedIdentities</a:string>
      <a:string>AllIdentityAttributes</a:string>
      <a:string>AgentHealthStatus</a:string>
    </GetReportTypesResult>
  </GetReportTypesResponse>
```

Figure 8 Report types

Note: As new reports are added to OpenManage Essentials, they will be added to the list of report types. Using the **<BASE_URI>/Reports** operation provides the current list of supported reports.

2.16.2 Getting reports

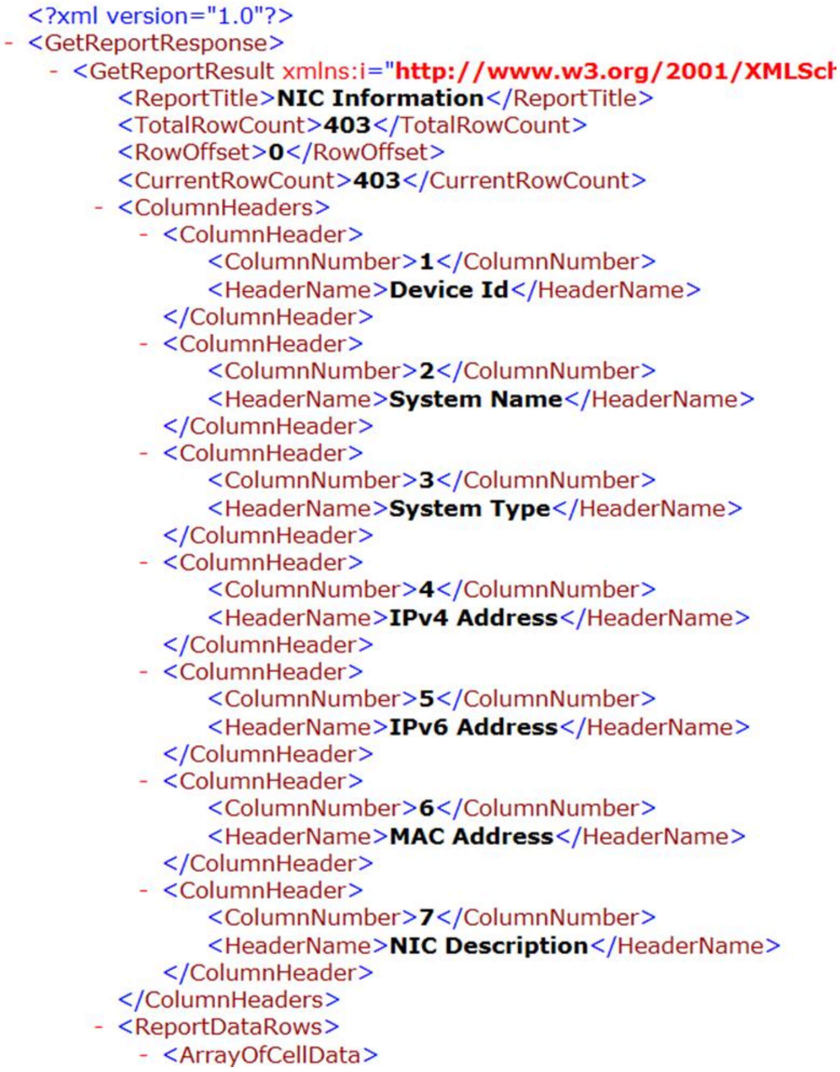
The following resource URI can be used for getting reports:

<BASE_URI>/Reports/{Report Type}

For the report types, see the Report Types section.

2.16.3 Generic report format

The following is an example of a generic report:



```
  <?xml version="1.0"?>
- <GetReportResponse>
  - <GetReportResult xmlns:i="http://www.w3.org/2001/XMLSch
      <ReportTitle>NIC Information</ReportTitle>
      <TotalRowCount>403</TotalRowCount>
      <RowOffset>0</RowOffset>
      <CurrentRowCount>403</CurrentRowCount>
    - <ColumnHeaders>
      - <ColumnHeader>
          <ColumnNumber>1</ColumnNumber>
          <HeaderName>Device Id</HeaderName>
        </ColumnHeader>
      - <ColumnHeader>
          <ColumnNumber>2</ColumnNumber>
          <HeaderName>System Name</HeaderName>
        </ColumnHeader>
      - <ColumnHeader>
          <ColumnNumber>3</ColumnNumber>
          <HeaderName>System Type</HeaderName>
        </ColumnHeader>
      - <ColumnHeader>
          <ColumnNumber>4</ColumnNumber>
          <HeaderName>IPv4 Address</HeaderName>
        </ColumnHeader>
      - <ColumnHeader>
          <ColumnNumber>5</ColumnNumber>
          <HeaderName>IPv6 Address</HeaderName>
        </ColumnHeader>
      - <ColumnHeader>
          <ColumnNumber>6</ColumnNumber>
          <HeaderName>MAC Address</HeaderName>
        </ColumnHeader>
      - <ColumnHeader>
          <ColumnNumber>7</ColumnNumber>
          <HeaderName>NIC Description</HeaderName>
        </ColumnHeader>
      </ColumnHeaders>
    - <ReportDataRows>
      - <ArrayOfCellData>
```

Figure 9 Generic report

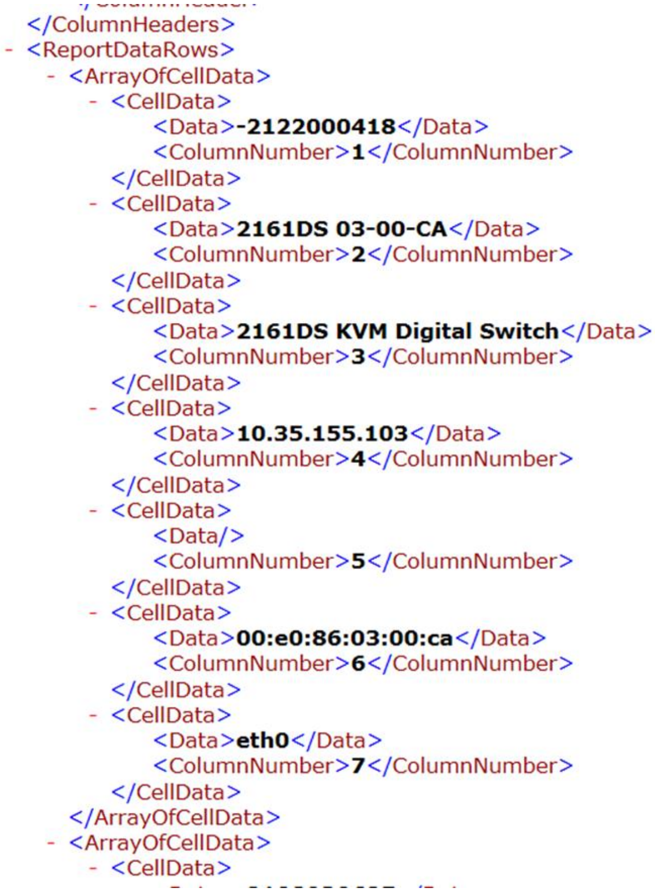
The title of the report is the same as the report in OpenManage Essentials. The TotalRowCount is the number of rows in the entire report.

If paginiation has been used as described in Data , then the RowOffset will be the number of rows skipped and the CurrentRowCount will be the number of rows included in this part of the report that has been returned.

The next part of the report are the column headers. The column number indicates the order of the columns. They match the names in the report, except for DeviceId. Device Id is included for convenience in retrieving data from other API calls.

The data rows are a list of rows which include the cell data and the matching column number. There are no row numbers to indicate the order, rather it is the order in which the report is read.

The following is an example of the data rows in the report:



```
          </ColumnHeader>
        </ColumnHeaders>
      - <ReportDataRows>
        - <ArrayOfCellData>
          - <CellData>
              <Data>-2122000418</Data>
              <ColumnNumber>1</ColumnNumber>
            </CellData>
          - <CellData>
              <Data>2161DS 03-00-CA</Data>
              <ColumnNumber>2</ColumnNumber>
            </CellData>
          - <CellData>
              <Data>2161DS KVM Digital Switch</Data>
              <ColumnNumber>3</ColumnNumber>
            </CellData>
          - <CellData>
              <Data>10.35.155.103</Data>
              <ColumnNumber>4</ColumnNumber>
            </CellData>
          - <CellData>
              <Data/>
              <ColumnNumber>5</ColumnNumber>
            </CellData>
          - <CellData>
              <Data>00:e0:86:03:00:ca</Data>
              <ColumnNumber>6</ColumnNumber>
            </CellData>
          - <CellData>
              <Data>eth0</Data>
              <ColumnNumber>7</ColumnNumber>
            </CellData>
          </ArrayOfCellData>
        - <ArrayOfCellData>
          - <CellData>
```

Figure 10 Data rows

When parsing the data, note that data types in this report format are all strings. You can determine the best type to which you want to deserialize the data.

Note: The dates and time are all in UTC format and are consistent with other XML date and time formats.

2.17 Manage Discovery ranges

This provides an option to get list of discovery ranges and groups, new discovery group creation and add discovery ranges to the group, add discovery ranges to an existing group, add exclude ranges, and delete discovery range and group.

The following URI can be used to manage discovery ranges/groups:

URI: **<BASE_URI>/DiscoveryRanges**

Operations supported: GET, POST, PUT, and DELETE

Attributes (filtering): RangeID, and GroupID

The attributes associated with manage discovery groups and discovery ranges are described in the following table:

Attribute Name	Description
Address	IP address of the discovery range.
HostName	Name of the host.

Attribute Name	Description
Name	The name of the range.
RangeID	The unique identifier for the range.
SubnetMask	Subnet mask for the discovery range.
GroupName	The name of the discovery group.
GroupID	The unique identifier for the group.
Timeout	Timeout
Retries	Number of attempts for ICMP/WSMan configuration
WSManUserName	iDRAC user name.
WSManPassword	iDRAC password.
Port	WSMan port number
SecureModeEnabled	true/false
SkipNameCheck	true/false
TrustedSiteEnabled	true/false
CertificateFile	Path of the certificate file if TrustedSiteEnabled is false. Otherwise value for CertificateFile will be null.
DiscoveryAction	Discovery action to be performed on the added discovery range. (Enum value)

Note:

1. The certificate file should be located in the same server where OpenManage Essentials is running.
2. All the boolean values (true/false) should be in lower case only.

The attributes associated with the discovery action enumeration are defined in the following table.

Enum Value	Description
0	Do not perform discovery or inventory
1	Perform only discovery
2	Perform both discovery and inventory

2.17.1 Get list of Discovery Groups and Discovery Ranges

The following resource URI can be used to get list of discovery groups and discovery ranges in OpenManage Essentials:

<BASE_URI>/DiscoveryRanges

Operations supported: GET

A GET operation on an URI retrieves all the child groups and the ranges under child groups and root group (All Ranges).

Note: If there is any group without discovery range(s), then it will show only GroupID and GroupName.

2.17.2 Group creation and add Discovery Range(s)

The following resource URI can be used to create a group and add discovery rang(s) to this group in OpenManage Essentials:

<BASE_URI>/DiscoveryRanges

Operations supported: POST

A sample XML that would comprise the payload when initiating a POST operation to create a group and add discovery range(s) is as follows:

```
<RESTAddDiscoveryRange>
      <DiscoveryRanges>
            <GroupName>groupname</GroupName>
            <Ranges>
                  <DiscoveryRange>
                        <Address>IPRange</Address>
                        <Name>rangename</Name>
                        <NetMask>netmask</NetMask>
                  </DiscoveryRange>
                  <DiscoveryRange>
                        <Address>IPRange</Address>
                        <Name>rangename</Name>
                        <NetMask>netmask</NetMask>
                  </DiscoveryRange>
                  <DiscoveryRange>
                        <HostName>hostname</HostName>
                        <NetMask>netmask</NetMask>
                  </DiscoveryRange>
            </Ranges>
      </DiscoveryRanges>
      <DiscoveryConfiguration>
            <ICMPConfig>
                  <Timeout>timeout</Timeout>
                  <Retries>retries</Retries>
            </ICMPConfig>
            <WSManDiscoveryConfig>
                  <WSManUserName>iDRACUserName</WSManUserName>
                  <WSManPassword>iDRACPassword</WSManPassword>
                  <Timeout>timeout</Timeout>
                  <Retries>retries</Retries>
                  <Port>portnumber</Port>
                  <SecureModeEnabled>true/false</SecureModeEnabled>
                  <SkipNameCheck>true/false</SkipNameCheck>
                  <TrustedSiteEnabled>true/false</TrustedSiteEnabled>
                  <CertificateFile>certificatepath</CertificateFile>
            </WSManDiscoveryConfig>
      </DiscoveryConfiguration>
      <DiscoveryAction>discoveryaction</DiscoveryAction>
</RESTAddDiscoveryRange>
```

Note:
This feature is supported for servers only with WSMan protocol.
If group name is not provided by the user, then discovery range(s) will be added directly under the root node (All Ranges).
Comma separated Host names are not allowed.

Default value for the subnet mask is 255.255.255.0.
Maximum supported ICMP timeout is 5000 ms and retries is 10.
Maximum supported WSMan time out is 360s and retries is 10.

2.17.3 Add Discovery Range(s) to an existing Group

The following resource URI can be used to add discovery rang(s) to an existing group in OpenManage Essentials:

<BASE_URI>/DiscoveryRanges/Groups/{GroupID}

Operations supported: PUT

A sample XML that would comprise the payload when initiating a PUT operation to add discovery range(s) to an existing group is as follows:

```
<RESTAddDiscoveryRangeToExistingGroup>
      <DiscoveryRanges>
            <DiscoveryRange>
                  <Address>IPrange</Address>
                  <Name>rangename</Name>
                  <NetMask>netmask</NetMask>
            </DiscoveryRange>
            <DiscoveryRange>
                  <Address>IPrange</Address>
                  <NetMask>netmask</NetMask>
            </DiscoveryRange>
            <DiscoveryRange>
                  <HostName>hostname</HostName>
                  <NetMask>netmask</NetMask>
            </DiscoveryRange>
      </DiscoveryRanges>
</RESTAddDiscoveryRangeToExistingGroup>
```

Note:
Comma separated Host names are not allowed.
Default value for the subnet mask is 255.255.255.0.

2.17.4 Add Exclude Range(s)

The following resource URI can be used to add exclude rang(s) in OpenManage Essentials:

<BASE_URI>/ExcludeRanges

Operations supported: PUT

A sample XML that would comprise the payload when initiating a PUT operation to add exclude range(s) is as follows:

```
<RESTAddExcludeRange>
      <Ranges>
            <Range>
                  <Address>IPrange</Address>
                  <Name>rangename</Name>
            </Range>
            <Range>
                  <Address>IPrange</Address>
            </Range>
      </Ranges>
</RESTAddExcludeRange>
```

2.17.5 Delete Discovery Range

The following resource URI can be used to delete discovery range in OpenManage Essentials:

<BASE_URI>/DiscoveryRanges/Ranges/{RangeID}

Operations supported: DELETE

2.17.6 Delete Discovery Group

The following resource URI can be used to delete discovery group in OpenManage Essentials:

<BASE_URI>/DiscoveryRanges/Groups/{GroupID}

Operations supported: DELETE

2.18 System Update Compliance

System Update compliance provides information about non-compliant devices and packages for firmware and drivers in a group.

Note: The REST APIs for gathering information about systems and packages which are noncompliant with respect to the latest System Updates, takes more time to execute in a scaled environment. To view the noncomplaint systems in a scaled environment, you can use REST API call on individual groups or devices.

2.18.1 Non-Compliant devices

Following are URI’s used to fetched non-compliant device. These URI also support paging of output data.

Note: This will be equivalent to list of non-compliant devices under All Devices.

1. This URI will return all the devices which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant

2. This URI will return the count of devices which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant/Count

3. This URI will return the first {top} number of devices which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant/$top={top}

4. This URI will return the first {top} number of devices after skipping {skip} number of devices from the first which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant/$skip={skip}&$top={top}

2.18.2 Non-Compliant devices for DeviceGroups

Following are URI's used to fetched non-compliant device from input device group id. These URI also support paging of output data.

1. This URI will return all the devices which are non-compliant with respect to the group. If the group is custom group and associated to custom catalog, the devices which are non-compliant with associated catalog will be shown. If the group is not associated to custom catalog, then devices which are non-compliant with respect to default catalog will be shown:
   <BASE_URI>/DeviceGroups/<group-id>/SystemUpdateCompliance/NonCompliant/Devices

2. This URI will return the count of devices which are non-compliant with respect to the group. If the group is custom group and associated to custom catalog, the devices which are non-compliant with associated catalog will be considered. If the group is not associated to custom catalog then devices which are non-compliant with respect to default catalog will be considered:
   <BASE_URI>/DeviceGroups/<group-id>/SystemUpdateCompliance/NonCompliant/Devices/count

2.18.3 Non-Compliant Package For Devices

Following are URIs used to fetched non-compliant pacakges for devices. These URI also support paging of output data.

1. This URI will return all the non compliant packages for all the devices which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant/Packages

2. This URI will return the count of non compliant packages for all the devices which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant/Packages/count

3. This URI will return the {top} number of non compliant packages for all the devices which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant/Packages/$top={top}

4. This URI will return the {top} number of non compliant packages after skipping {skip} number of pacakges for all the devices which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/SystemUpdateCompliance/NonCompliant/Packages/$skip={skip}&$top={top}

5. This URI will return all the non compliant packages for the device mentioned using device id which are non-compliant with respect to system update portal:
   <BASE_URI>/Devices/<device-id>/SystemUpdateCompliance/NonCompliant/Packages

2.18.4 Non-Compliant Package for DeviceGroups

Following are URIs used to fetched non-compliant pacakges for devices. These URI also support paging of output data:

1. This URI will return all the non-compliant packages for the devices which are non-compliant with respect to the group. If the group is custom group and associated to custom catalog, the devices which are non-compliant with associated catalog will be considered. If the group is not associated to custom catalog, then devices which are non-compliant with respect to default catalog will be considered:
   <BASE_URI>/DeviceGroups/<group-id>/SystemUpdateCompliance/NonCompliant/Packages

2. This URI will return the count of non-compliant packages for the devices which are non-compliant with respect to the group. If the group is custom group and associated to custom catalog, the devices which are non-compliant with associated catalog will be considered. If the group is not associated to custom catalog, then devices which are non-compliant with respect to default catalog will be considered:
   <BASE_URI>/DeviceGroups/<group-id>/SystemUpdateCompliance/NonCompliant/Packages/Count

3. This URI will return the first {top} number of non-compliant packages for the devices which are non-compliant with respect to the group. If the group is custom group and associated to custom catalog, the devices which are non-compliant with associated catalog will be considered. If the group is not associated to custom catalog, then devices which are non-compliant with respect to default catalog will be considered:
   <BASE_URI>/DeviceGroups/<group-id>/SystemUpdateCompliance/NonCompliant/Packages/$top={top}

4. This URI will return the first {top} number of non-compliant packages after skipping {skip} count for the devices which are non-compliant with respect to the group. If the group is custom group and associated to custom catalog, the devices which are non-compliant with associated catalog will be considered. If the group is not associated to custom catalog, then devices which are non-compliant with respect to default catalog will be considered:
   <BASE_URI>/DeviceGroups/<group-id>/SystemUpdateCompliance/NonCompliant/Packages/$skip={skip}&$top={top}

2.18.5 Sample Output for Devices and Packages

Sample Output for List<Devices>

```
<GetAllNonCompliantDevicesResponse>
  <GetAllNonCompliantDevicesResult xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  <Device>
    <AssetTag />
    <DNSName />
    <DiscoveryTime>2017-11-23T00:20:12</DiscoveryTime>
    <ExpressServiceCode>38702992502</ExpressServiceCode>
    <GlobalStatus>16</GlobalStatus>
    <Id>3</Id>
    <InventoryTime>2017-11-23T00:19:38</InventoryTime>
    <IsIdrac>true</IsIdrac>
    <IsInband>false</IsInband>
    <LaunchURL>https://100.96.26.171:443</LaunchURL>
    <NICS>
      <NIC>
        <Description>Broadcom Gigabit Ethernet BCM5720 - 44:A8:42:30:8C:44</Description>
        <IPAddress />
        <MACAddress>44:A8:42:30:8C:44</MACAddress>
        <Pingable />
```

```
           <Vendor>Broadcom Corp</Vendor>
         </NIC>
         <NIC>
           <Description>iDRAC.Embedded.1</Description>
           <IPAddress>100.96.26.171</IPAddress>
           <MACAddress>44:A8:42:30:8C:47</MACAddress>
           <Pingable>2</Pingable>
           <Vendor />
         </NIC>
         <NIC>
            <Description>bond0</Description>
            <IPAddress>100.96.25.102</IPAddress>
            <MACAddress>4c:76:25:4f:1f:cd</MACAddress>
            <Pingable>1</Pingable>
            <Vendor />
         </NIC>
        </NICS>
        <Name>idrac</Name>
        <NodeId>HS2RG52</NodeId>
        <OSName>VMware ESXi 6.5.0 build-4504200</OSName>
        <OSRevision>6.5.0 GA (build-4504200) Kernel 6.5.0 (x86_64)</OSRevision>
        <PowerStatus>1</PowerStatus>
        <ServiceTag>HS2RG52</ServiceTag>
        <StatusTime>2017-12-07T06:34:21</StatusTime>
        <SystemId>1594</SystemId>
        <SystemModel>PowerEdge R530</SystemModel>
        <Type>4</Type>
       </Device>
    </GetAllNonCompliantDevicesResult>
  </GetAllNonCompliantDevicesResponse>
```

Sample output for List<Packages>

```
   <ArrayOfUpdatePackage xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
    <UpdatePackage>
      <CategoryID>FW</CategoryID>
      <ComponentTypeID>FRMW</ComponentTypeID>
      <ComponentTypeName>Firmware</ComponentTypeName>
      <DeviceDiscoveredTime>2017-11-23T00:20:08</DeviceDiscoveredTime>
      <DeviceId>123</DeviceId>
      <DeviceInventoryTime>2017-11-23T00:20:40</DeviceInventoryTime>
      <DeviceName>WIN-O9A8M4TTVGV</DeviceName>
      <DeviceServiceTag>451BHJ2</DeviceServiceTag>
      <DeviceSystemModelType>PowerEdge R430</DeviceSystemModelType>
      <DisplayVersion>2.25</DisplayVersion>
      <HWDescription>BP13G+ 0:1</HWDescription>
      <HashMD5>c7057ae5c6d45539e4f911fff5534aa6</HashMD5>
      <ImportanceID>3</ImportanceID>
      <InformationURL>http://www.dell.com/support/home/us/en/19/Drivers/DriversDetails?driverId=HRP
        1V</InformationURL>
      <InstallationServices>3</InstallationServices>
      <InstanceId>DCIM:INSTALLED#308_C_Enclosure.Internal.0-1:RAID.Integrated.1-1</InstanceId>
      <ManifestID>5d3bd680-549c-30ca-8656-989782b6de7e</ManifestID>
      <OOBCapability>1</OOBCapability>
      <OSArchitecture i:nil="true" />
      <OperatingSystem>Microsoft Windows 2016 Server, Standard x64 Edition</OperatingSystem>
      <PackageFile>Firmware_HRP1V_WN64_2.25_A00-00_01.EXE</PackageFile>
      <PackageID>00370063-0030-0035-3700-610065003500</PackageID>
      <PackageLocation>C:\Users\Administrator\Desktop\FOLDER04204235M/16/Firmware_HRP1V_WN64_2.25_A
        00-00_01.EXE</PackageLocation>
      <PackageServices>3</PackageServices>
```

```
        <PackageType>LW64</PackageType>
        <RebootRequired>true</RebootRequired>
        <SixyBitSupport>1</SixyBitSupport>
        <SoftwareVersion>2.23</SoftwareVersion>
        <Upgrade>true</Upgrade>
      </UpdatePackage>
      <UpdatePackage>
        <CategoryID>NI</CategoryID>
        <ComponentTypeID>FRMW</ComponentTypeID>
        <ComponentTypeName>Firmware</ComponentTypeName>
        <DeviceDiscoveredTime>2017-11-23T00:20:08</DeviceDiscoveredTime>
        <DeviceId>123</DeviceId>
        <DeviceInventoryTime>2017-11-23T00:20:40</DeviceInventoryTime>
        <DeviceName>WIN-O9A8M4TTVGV</DeviceName>
        <DeviceServiceTag>451BHJ2</DeviceServiceTag>
        <DeviceSystemModelType>PowerEdge R430</DeviceSystemModelType>
        <DisplayVersion>20.6.18</DisplayVersion>
        <HWDescription>Broadcom Gigabit Ethernet BCM5720 - 18:66:DA:9E:33:ED</HWDescription>
        <HashMD5>96e2158b7cf91cc8af99318e68fd3fcc</HashMD5>
        <ImportanceID>3</ImportanceID>
        <InformationURL>http://www.dell.com/support/home/us/en/19/Drivers/DriversDetails?driverId=1T8HX</InformationURL>
        <InstallationServices>3</InstallationServices>
        <InstanceId>DCIM:INSTALLED#701__NIC.Embedded.2-1-1</InstanceId>
        <ManifestID>5d3bd680-549c-30ca-8656-989782b6de7e</ManifestID>
        <OOBCapability>1</OOBCapability>
        <OSArchitecture i:nil="true" />
        <OperatingSystem>Microsoft Windows 2016 Server, Standard x64 Edition</OperatingSystem>
        <PackageFile>Network_Firmware_1T8HX_WN64_20.6.18.EXE</PackageFile>
        <PackageID>00360039-0065-0032-3100-350038006200</PackageID>
        <PackageLocation>C:\Users\Administrator\Desktop\FOLDER04444506M/1/Network_Firmware_1T8HX_WN64_20.6.18.EXE</PackageLocation>
        <PackageServices>3</PackageServices>
        <PackageType>LW64</PackageType>
        <RebootRequired>true</RebootRequired>
        <SixyBitSupport>1</SixyBitSupport>
        <SoftwareVersion>20.2.17</SoftwareVersion>
        <Upgrade>true</Upgrade>
      </UpdatePackage>
    </ArrayOfUpdatePackage>
```

Entry Name	Possible values

CatagoryID	SM = System Management
	DD = Drivers for OS Deployment
	NI = Network
	SA = Serial ATA
	AS = SAS
	SF = SAS RAID
	SE = SAS Non RAID
	BI = BIOS
	Chipset = Chipset
	FC = Fibre Channel
	PC = PCIe
	ES = ESM
	TH = Tape Drives
	AU = Audio
	VI = Video
	DI = Diagnostics
	SV = Lifecycle Controller
	FW = Firmware
	SG = Storage Controller

	CE = Chassis System Management
	IN = Input
	AP = Application
	iDRAC with Lifecycle Controller = iDRAC with Lifecycle Controller
	Express Flash PCIe SSD = Express Flash PCIe SSD

ImportanceID	1= Recommended, 2= Urgent, 3 = Optional
Installation Services	1 = Out of Band, 2 = In Band, 3 = Both, 4 = CMC
Package Services	1 = Out of Band, 2 = In Band, 3 = Both, 4 = CMC
OOBCapability	1 = true, 0 = false
SixyBitSupport	1 = true, 0 = false

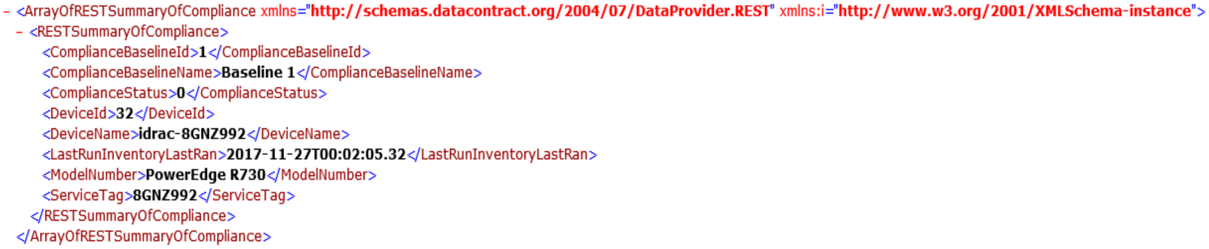
2.19 Configuration Compliance

The configuration of a server or chassis in a production environment must be properly maintained to ensure availability of the server.These server configuration settings tend to be drifted over time because of various reasons.The OME UI provides functionality to create configuration baseline and associate to multiple devices to have common settings if any drift in configuration then that device becomes non-compliant and user can take appropriate action to make it compliant. For more information about Configuration Baselines in OME, see the *Managing Server Configuration Baseline* white paper available on the support site.

REST API provides functionality to get non-compliant devices. The following is a output of non-compliant devices and configuration details associated with a specific device/group identified by <DEVICE_ID>/<GROUP_ID> or for all devices/groups from below URI table:

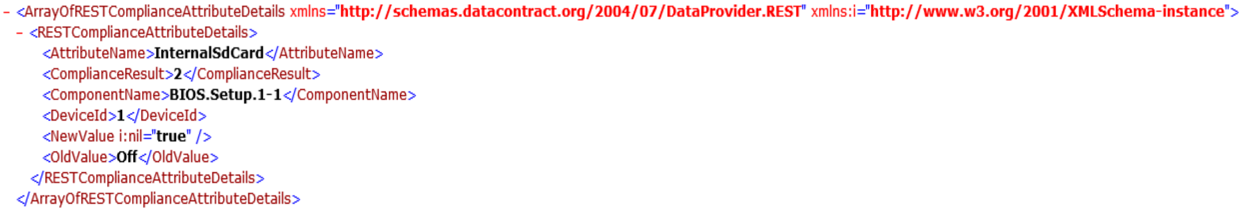
Operations Supported: GET

Sample output for non-compliant devices under device group.



```
- <ArrayOfRESTSummaryOfCompliance xmlns="http://schemas.datacontract.org/2004/07/DataProvider.REST" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  - <RESTSummaryOfCompliance>
      <ComplianceBaselineId>1</ComplianceBaselineId>
      <ComplianceBaselineName>Baseline 1</ComplianceBaselineName>
      <ComplianceStatus>0</ComplianceStatus>
      <DeviceId>32</DeviceId>
      <DeviceName>idrac-8GNZ992</DeviceName>
      <LastRunInventoryLastRan>2017-11-27T00:02:05.32</LastRunInventoryLastRan>
      <ModelNumber>PowerEdge R730</ModelNumber>
      <ServiceTag>8GNZ992</ServiceTag>
    </RESTSummaryOfCompliance>
  </ArrayOfRESTSummaryOfCompliance>
```

Sample output for non-compliant devices attribute details.



```
- <ArrayOfRESTComplianceAttributeDetails xmlns="http://schemas.datacontract.org/2004/07/DataProvider.REST" xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
  - <RESTComplianceAttributeDetails>
      <AttributeName>InternalSdCard</AttributeName>
      <ComplianceResult>2</ComplianceResult>
      <ComponentName>BIOS.Setup.1-1</ComponentName>
      <DeviceId>1</DeviceId>
      <NewValue i:nil="true" />
      <OldValue>Off</OldValue>
    </RESTComplianceAttributeDetails>
  </ArrayOfRESTComplianceAttributeDetails>
```

Attributes	Description
AttributeName	Name of configuration attribute.
ComplianceResult	Compliance status of the attribute.
ComponentName	Component name of the attribute.
DeviceId	Device Id
NewValue	New(Inventory) value of the attribute.

OldValue	Old(Template) value of the attribute.

Enum value for ComplianceResult Attribute.

Enum Value	Description
0	Unknown – The status is unknown.
1	Added – The status is Added.
2	Missing – The status is missing.
3	Different – The status is different.

2.19.1 Configuration Compliance For Devices

Following are the URIs used to fetch non-compliant configuration details for all Devices or for specific Device based on input device ID. These URIs also support paging of output data:

1. Get all non-compliant configuration details for all devices:
   <BASE_URI>/Devices/ConfigurationCompliance/NonCompliant/NonComplianceDetails

2. Get count of non-compliant configuration details for all devices:
   <BASE_URI>/Devices/ConfigurationCompliance/NonCompliant/NonComplianceDetails/Count

3. Get top {top} non-compliant configuration details:
   <BASE_URI>/Devices/ConfigurationCompliance/NonCompliant/NonComplianceDetails/$top={top}

4. Get top {top} non-compliant configuration details by skipping {skip}:
   <BASE_URI>/Devices/ConfigurationCompliance/NonCompliant/NonComplianceDetails/$skip={skip}&$top={top}

5. Get all non-compliant configuration details for device id {id}:
   <BASE_URI>/Devices/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails

6. Get count of non-compliant configuration details for device id {id}:
   <BASE_URI>/Devices/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails/Count

7. Get top {top} non-compliant configuration details for device id {id}:
   <BASE_URI>/Devices/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails/$top={top}

8. Get top {top} non-compliant configuration details by skipping {skip} for device id {id}:
   <BASE_URI>/Devices/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails/$skip={skip}&$top={top}

2.19.2 Configuration Compliance for DeviceGroups

Following are URI's used to fetched non-compliant configuration devices and details for all DeviceGroups or for specific DeviceGroups based on input group id. These URI also support paging of output data.

1. Get count of non-compliant devices for all groups:
   <BASE_URI>/DeviceGroups/ConfigurationCompliance/NonCompliant/Count

2. Get top {top} non-compliant devices for all groups:
   <BASE_URI>/DeviceGroups/ConfigurationCompliance/NonCompliant/$top={top}

3. Get top {top} non-compliant devices for group id {id}:
   <BASE_URI>/DeviceGroups/{id}/ConfigurationCompliance/NonCompliant/$top={top}

4. Get top {top} non-compliant devices by skipping {skip} for group id {id}:
   <BASE_URI>/DeviceGroups/{id}/ConfigurationCompliance/NonCompliant/$skip={skip}&$top={top}

5. Get count of non-compliant configuration details for group id {id}:
   <BASE_URI>/DeviceGroups/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails/Count

6. Get all non-compliant configuration details for group id {id}:
   <BASE_URI>/DeviceGroups/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails

7. Get top {top} non-compliant configuration details for group id {id}:
   <BASE_URI>/DeviceGroups/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails/$top={top}

8. Get top {top} non-compliant configuration details by skipping {skip} for group id {id}:
   <BASE_URI>/DeviceGroups/{id}/ConfigurationCompliance/NonCompliant/NonComplianceDetails/$skip={skip}&$top={top}

A Technical support and resources

- Dell.com/support is focused on meeting customer needs with proven services and support.
- To watch quick and short videos about handling the PowerEdge server components, visit the QRL video website.

A.1 Related resources

A.1.1 Contacting Dell EMC

Dell provides several online and telephone-based support and service options. Availability varies by country and product, and some services may not be available in your area. To contact Dell for sales, technical support, or customer service issues:

1. Visit www.dell.com/support.
   a. Select your support category.
   b. Verify your country or region in the **Choose a Country/Region** drop-down menu at the top of page.
   c. Select the appropriate service or support link based on your need.

For information about documentation support:

1. Go to dell.com/support/manuals.
   a. In the **Tell us about your Dell system** section, under No, select **Choose from a list of all Dell products** and click **Continue**.
   b. In the **Select your product type** section, click **Software, Monitors, Electronics & Peripherals**.
   c. In the **Choose your Dell Software, Monitors, Electronics & Peripherals** section, click **Software**.
   d. In the **Choose your Dell Software** section, click the required link from the following:
   - Client System Management
   - Enterprise System Management
   - Remote Enterprise
   - System Management–Serviceability Tools

   e. To view the document, click the required product version.