Discovering Features in the iDRAC Remote Services API

A Dell technical white paper.

Chris A. Poblete
Khachatur Papanyan

Enterprise Product Group
Discovering Features in the iDRAC Remote Services API

Contents

Executive summary ..................................................................................................... 4
Introduction.............................................................................................................. 4
Before you start ......................................................................................................... 4
Discovering implemented profiles ................................................................................... 5
Major and minor profile versions ..................................................................................... 6
Discovering the licensed feature in a profile ................................................................. 8
Discovering the implementation namespace ....................................................................... 9
Discovering the device/component version ...................................................................... 10
How do you determine system generation? ................................................................. 11
Conclusion .............................................................................................................. 11

Tables

Table 1. Relationship between DTMF standard profiles and corresponding DCIM profiles. .......... 6
Executive summary

This document is for systems administrators or console application developers who are interested in automating the discovery of the vast features and capabilities offered by the Remote Services API in iDRAC, which constantly changes and grows between firmware generation and revisions. Consoles can interact with the service to determine features and capabilities.

The document primarily focuses on the Dell PowerEdge 12th generation of servers but the methodologies discussed apply to the 11th generation servers as well.

Introduction

Dell PowerEdge servers equipped with the Integrated Dell Remote Access Controller (iDRAC) offers an agentless, out-of-band, remote management through the Remote Services API. Remote Services is also called Web Services because the interface is WS-Management (WS-MAN), a SOAP-based protocol that provides CIM-style data access. Remote Services is not an end-user interface but rather an API by which inputs can be complex type and outputs may require complex processing and interpretation based on the API definition. API definitions are described in a profile specification.

Profile specifications define the CIM model and behavior for a particular management domain including associated classes, methods and properties. A DMTF profile defines the classes, properties, methods and values that are instantiated and manipulated to represent and manage a given domain. Implementations adhering to the specification lend to the interoperability nature of CIM. Profiles may be extended to include features and capabilities new to the domain. DCIM profiles are Dell extension profiles.

Understanding and navigating through the API may be overwhelming. This document focuses on the initial phase of this challenge, which is to discover features and capabilities of implemented API.

Before you start

This section contains references to bookmark as related to this document.

1. Lifecycle Controller Wiki with links to white papers, scripts and more: http://delltechcenter.com/LC
2. DMTF profile specification documents: http://dmtf.org/standards/profiles
Discovering Features in the iDRAC Remote Services API

Discovering implemented profiles

The Remote Service API management features are advertised through the implemented profile registration. An implemented profile advertises its existence by instantiating a CIM_RegisteredProfile instance in the “root/interop” namespace. The instance provides the name and version of the CIM or DCIM profile specification to which the implementation conforms.

Command Pseudocode:

Enumerate CIM_RegisteredProfile class on root/interop namespace.

Response Snippet:

<n1:DCIM_LCRegisteredProfile>
  <n1:AdvertiseTypeDescriptions>WS-Identify</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypeDescriptions>Interop Namespace</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypes>1</n1:AdvertiseTypes>
  <n1:InstanceID>DCIM:LCManagement:1.3.0</n1:InstanceID>
  <n1:OtherRegisteredOrganization>DCIM</n1:OtherRegisteredOrganization>
  <n1:RegisteredName>LC Management</n1:RegisteredName>
  <n1:RegisteredOrganization>1</n1:RegisteredOrganization>
  <n1:RegisteredVersion>1.3.0</n1:RegisteredVersion>
</n1:DCIM_LCRegisteredProfile>

The example above advertises that the Web Service offers management capability that conforms to the “LC Management” profile specification version 1.3.0. Use this information to look up the specification document to understand what it can do for you. Also, check the value of “RegisteredOrganization”. If the value is 2, which maps to DMTF, then reference a DMTF profile. If the value is 1, which maps to Other, then look at the value of “OtherRegisteredOrganization” and find the DCIM. In this case, reference a DCIM profile.

The profile specifications are found on the internet. If the implemented profile’s registered organization is “DCIM”, then the profile is a Dell extended profile. The Dell extended profiles, also referred as DCIM profiles, are found at the following URL:


The DMTF profile is found at the following URL:

http://www.dmtf.org/standards/profiles

In the Dell PowerEdge 12th generation of servers, the specifics of Dell implementation of the DMTF standard profiles are advertised through the similarly named Dell extended, DCIM, profiles. The following table shows the relationship between the DMTF standard profiles and their corresponding DCIM profiles:
Table 1. Relationship between DMTF standard profiles and corresponding DCIM profiles.

<table>
<thead>
<tr>
<th>DCIM profile</th>
<th>Relationship</th>
<th>DMTF profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCIM Base Metrics Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Base Metrics Profile 1.0</td>
</tr>
<tr>
<td>DCIM Base Server and Physical Asset Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Base Server Profile 1.0</td>
</tr>
<tr>
<td>DCIM Base Server and Physical Asset Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Physical Asset Profile 1.0</td>
</tr>
<tr>
<td>DCIM BIOS and Boot Management Profile 1.2</td>
<td>Specializes*</td>
<td>DMTF BIOS Management Profile 1.0</td>
</tr>
<tr>
<td>DCIM BIOS and Boot Management Profile 1.2</td>
<td>Relates*</td>
<td>DMTF Boot Control Profile 1.0</td>
</tr>
<tr>
<td>DCIM Persistent Storage Profile 1.0</td>
<td>Relates*</td>
<td>DMTF Opaque Management Data Profile 1.0</td>
</tr>
<tr>
<td>DCIM Power State Management Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Power State Management Profile 1.0 and 2.0</td>
</tr>
<tr>
<td>DCIM Power Supply Profile 2.1</td>
<td>Specializes*</td>
<td>DMTF Power Supply Profile 1.1</td>
</tr>
<tr>
<td>DCIM Profile Registration Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Profile Registration Profile 1.0</td>
</tr>
<tr>
<td>DCIM Record Log Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Record Log Profile 1.0 and 2.0</td>
</tr>
<tr>
<td>DCIM Role Based Authorization Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Role Based Authorization Profile 1.0</td>
</tr>
<tr>
<td>DCIM Sensors Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Sensors Profile 1.0</td>
</tr>
<tr>
<td>DCIM Simple Identity Management Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Simple Identity Management Profile 1.0</td>
</tr>
<tr>
<td>DCIM Service Processor Profile 1.0</td>
<td>Specializes*</td>
<td>DMTF Service Processor Profile 1.0</td>
</tr>
<tr>
<td>DCIM Software Inventory Profile 1.1</td>
<td>Specializes*</td>
<td>DMTF Software Inventory Profile 1.1</td>
</tr>
<tr>
<td>DCIM Software Update Profile 1.0</td>
<td>Relates*</td>
<td>DMTF Software Update Profile 1.0</td>
</tr>
</tbody>
</table>

Note: * - Specializes relationship means that the DCIM profiles completely conforms to the requirements of the corresponding DMTF profile. Relates relationship means that the DCIM profile uses the same modeling as the corresponding DMTF profile but does not completely conform to it. In the related profiles, the deviation is usually confined to the DCIM profile not instantiating the association classes required by the DMTF profile.

**Major and minor profile versions**

The profile versions advertise to the clients not only the exact specification version that the particular profile implementation conforms to, but also how much has changed from the previous profile implementation.
The same named profile may have different versions advertised in different revisions of iDRAC firmware. For example, in earlier revision of the iDRAC firmware the profile may have version 1.3 (see the highlighted line from the below response snippet):

Response Snippet:

<n1:DCIM_LCRegisteredProfile>
  <n1:AdvertiseTypeDescriptions>WS-Identify</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypeDescriptions>Interop Namespace</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypes>1</n1:AdvertiseTypes>
  <n1:InstanceID>DCIM:LCManagement:1.3.0</n1:InstanceID>
  <n1:OtherRegisteredOrganization>DCIM</n1:OtherRegisteredOrganization>
  <n1:RegisteredName>LC Management</n1:RegisteredName>
  <n1:RegisteredOrganization>1</n1:RegisteredOrganization>
  <n1:RegisteredVersion>1.3.0</n1:RegisteredVersion>
</n1:DCIM_LCRegisteredProfile>

But in the later revision of the iDRAC firmware the version may be 1.5.0 for the same profile:

Response Snippet:

<n1:DCIM_LCRegisteredProfile>
  <n1:AdvertiseTypeDescriptions>WS-Identify</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypeDescriptions>Interop Namespace</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypes>1</n1:AdvertiseTypes>
  <n1:InstanceID>DCIM:LCManagement:1.3.0</n1:InstanceID>
  <n1:OtherRegisteredOrganization>DCIM</n1:OtherRegisteredOrganization>
  <n1:RegisteredName>LC Management</n1:RegisteredName>
  <n1:RegisteredOrganization>1</n1:RegisteredOrganization>
  <n1:RegisteredVersion>1.5.0</n1:RegisteredVersion>
</n1:DCIM_LCRegisteredProfile>

In this case, only the minor version of the profile has changed from 1.3.0 to 1.5.0. Such a change in the minor version of the profile indicates that the implementation is backwards compatible. Thus, the client or console that was developed to work with the earlier version of the iDRAC works to process the later version of the iDRAC regardless of the minor version upgrade.

However, if the major version is changed, for example the newer version of the iDRAC firmware has a response snippet containing 2.0.0 version of the profile, then there is no guarantee that the client or console developed for the 1.5.0 or 1.3.0 versions of the profile would work with the new version of the iDRAC firmware.

For more details on the exact requirement for advertising profiles, see:

DMTF Profile Registration
  http://dmtf.org/standards/profiles

DCIM Profile Registration
Discovering the licensed feature in a profile

The DCIM Profile Registration extended the DMTF profile to include the required license information. Conformant domain profiles advertise a list of licenses required for that profile. A license identifies the feature it supports. The status of the license is reported so you can make a conscious decision whether or not to use the feature. License information in Profile Registration is added beginning with the second generation of Lifecycle Controller.

Command Pseudocode:

Enumerate CIM_RegisteredProfile class on root/interop namespace.

Response Snippet:

```
<n1:DCIM_LCRegisteredProfile>
  <n1:AdvertiseTypeDescriptions>WS-Identify</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypeDescriptions>Interop Namespace</n1:AdvertiseTypeDescriptions>
  <n1:AdvertiseTypes>1</n1:AdvertiseTypes>
  <n1:InstanceID>DCIM:LCManagement:1.1.0</n1:InstanceID>
  <n1:OtherRegisteredOrganization>DCIM</n1:OtherRegisteredOrganization>
  <n1:ProfileRequireLicense>Auto Discovery</n1:ProfileRequireLicense>
  <n1:ProfileRequireLicense>Part Replacement</n1:ProfileRequireLicense>
  <n1:ProfileRequireLicense>Remote Firmware Configuration</n1:ProfileRequireLicense>
  <n1:ProfileRequireLicense>Remote Inventory Export</n1:ProfileRequireLicense>
  <n1:ProfileRequireLicense>Server Profile Export and Import</n1:ProfileRequireLicense>
  <n1:ProfileRequireLicenseStatus>LICENSED</n1:ProfileRequireLicenseStatus>
  <n1:ProfileRequireLicenseStatus>LICENSED</n1:ProfileRequireLicenseStatus>
  <n1:ProfileRequireLicenseStatus>LICENSED</n1:ProfileRequireLicenseStatus>
  <n1:ProfileRequireLicenseStatus>LICENSED</n1:ProfileRequireLicenseStatus>
  <n1:ProfileRequireLicenseStatus>NOT LICENSED</n1:ProfileRequireLicenseStatus>
  <n1:RegisteredName>LC Management</n1:RegisteredName>
  <n1:RegisteredOrganization>1</n1:RegisteredOrganization>
  <n1:RegisteredVersion>1.6.0</n1:RegisteredVersion>
</n1:DCIM_LCRegisteredProfile>
```

The example above advertises an implementation of the LC Management profile. It reports that this profile has functionality relating to the “Auto Discovery” feature and that it is licensed. However, it reports that “Server Profile Export and Import” is not licensed, which means that any functionality this profile provides for that feature is not present or-functional. For example, performing an Enumeration or Get operation on a given class associated with the feature that is not licensed returns no information. Similarly, invoking a method on an unlicensed class associated with the feature results in an error.

Related Profiles:

- DCIM Profile Registration; DCIM License Management

See Also:

- WSMAN License and Privilege Specification
Discovering the implementation namespace

The interface and data model used in the Remote Services API are based on DMTF standards designed for interoperability. An application compliant to WS-MAN and CIM standards can perform manageability of a particular domain to any device that offers the service based on the same standards. For example, a server, laptop, printer or a network switch offering a compliant WS-MAN service can respond successfully to a reboot command from a compliant application.

For interoperability, do not hard code an implementation namespace in an application. The namespace is dynamically discovered using the CIM_ElementConformsToProfile class instances in root/interop namespace. This class is used to associate instances of Central and Scoping classes defined in profiles with the CIM_RegisteredProfile that identifies the particular profile specifications that are implemented.

Command Pseudocode:

Enumerate CIM_ElementConformsToProfile class on root/interop namespace.

Response Snippet:

```xml
<n1:ManagedElement>
  <wsa:Address>http://schemas.xmlsoap.org/soap/addressing/role/anonymous</wsa:Address>
  <wsa:ReferenceParameters>
    <wsman:SelectorSet>
      <wsman:Selector Name="Name">srv:system</wsman:Selector>
      <wsman:Selector Name="CreationClassName">DCIM_ComputerSystem</wsman:Selector>
      <wsman:Selector Name="__cimnamespace">root/dcim</wsman:Selector>
    </wsman:SelectorSet>
  </wsa:ReferenceParameters>
</n1:ManagedElement>
```

The ManagedElement property of the example CIM_ElementConformsToProfile instance is an EndPointReference of the Central and Scoping class, which contains the implementation namespace. The implementation namespace in the example is “root/dcim.”

Another method to achieve a similar response is to enumerate the Associated (also called Associator) classes of CIM_RegisteredProfile instance. This method returns a smaller set compared to the above method.

Related Profiles:

DMTF Profile Registration

http://dmtf.org/standards/profiles

DCIM Profile Registration

Discovering the device/component version

Component version and supported device firmware information are advertised by the Software Inventory profile specification. Besides reporting the version, it complements the Software Update profile specification. It reports Updatable property, which tells you whether it is updated or not. Use the properties ComponentID and DeviceID to match the corresponding update package for the device.

Command Pseudocode:

Enumerate CIM_SoftwareIdentity class on implementation namespace.

Response Snippet:

```
<n1:DCIM_SoftwareIdentity>
  <n1:BuildNumber>0</n1:BuildNumber>
  <n1:Classifications>10</n1:Classifications>
  <n1:ComponentID xsi:nil="true"/>
  <n1:ComponentType>FRMW</n1:ComponentType>
  <n1:DeviceID>0073</n1:DeviceID>
  <n1:ElementName>PERC H310 Mini</n1:ElementName>
  <n1:FQDD>RAID.Integrated.1-1</n1:FQDD>
  <n1:InstallationDate>2011-11-06T14:06:42Z</n1:InstallationDate>
  <n1:InstanceID>DCIM:INSTALLED#301_C_RAID.Integrated.1-1</n1:InstanceID>
  <n1:IsEntity>true</n1:IsEntity>
  <n1:MajorVersion>20</n1:MajorVersion>
  <n1:MinorVersion>10</n1:MinorVersion>
  <n1:RevisionNumber xsi:nil="true"/>
  <n1:RevisionString>1-0046</n1:RevisionString>
  <n1:Status>Installed</n1:Status>
  <n1:SubDeviceID>1F51</n1:SubDeviceID>
  <n1:SubVendorID>1028</n1:SubVendorID>
  <n1:Updateable>true</n1:Updateable>
  <n1:VendorID>1000</n1:VendorID>
  <n1:VersionString>20.10.1-0046</n1:VersionString>
  <n1:impactsTPMmeasurements>true</n1:impactsTPMmeasurements>
</n1:DCIM_SoftwareIdentity>
```

The example above shows the version information of the Dell PERC H310 controller present on the system. It reports that it can be updated. The value of InstanceID property is used by the Software Update profile to update this device.

An interesting property is Status. In the full response payload, you may notice other instances of the same device that has Status of “Available.” When this instance is present, it indicates that a previous version of the device firmware is available. It also indicates that the device firmware was updated in the past.

Related Profiles:

- DCIM Software Inventory
- DCIM Software Update

See Also:

- Remote Firmware Update white paper
Discovering Features in the iDRAC Remote Services API

How do you determine system generation?

Remote Services features and capabilities are constantly changing and growing to adapt to customer needs and improve user experience. Changes may be significant between firmware generations adding features and modifying existing features. See the following white paper to help you understand API changes between Dell PowerEdge 11th and 12th generation servers:

http://en.community.dell.com/techcenter/extras/m/white_papers/20066195.aspx

There is currently no interoperable method to determine system generation. For DCIM, you can determine system generation by enumerating the DCIM_SystemView class. In the instance, examine the value of LifecycleControllerVersion property. The value is in the form “x.y.z”. For Dell PowerEdge 12th generation server generation, the value of x is 2. For Dell PowerEdge 11th generation server generation, this property is added at a later revision. Therefore, if this property is not present in the instance, or if the value of x is 1, then it is a Dell PowerEdge 11th generation server.

Conclusion

With various methods to automate the discovery of the many rich features offered by the Remote Services API in iDRAC, you can develop an application or script that is interoperable across the generation and version of the firmware.