Dell Lifecycle Controller Web Services Interface Guide for Windows

A Dell Technical White Paper about the Best Practices to be followed for using the WS-Man—based Remote Services provided by iDRAC and Lifecycle Controller

Dell Engineering
September 2017
## Revisions

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>August 2013</td>
<td>Initial release</td>
</tr>
<tr>
<td>November 2016</td>
<td>Updates for iDRAC8 2.40.40.40 release (13th Generation PowerEdge Servers)</td>
</tr>
<tr>
<td>September 2017</td>
<td>Updates for iDRAC9 3.00.00.00 release (14th Generation PowerEdge Servers)</td>
</tr>
</tbody>
</table>

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.

Copyright © 2016 Dell Inc. or its subsidiaries. All Rights Reserved. Dell, EMC, and other trademarks are trademarks of Dell Inc. or its subsidiaries. Other trademarks may be the property of their respective owners.

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

1. Introduction .................................................................................................................. 13
2. References ...................................................................................................................... 14
3. Overview ......................................................................................................................... 15
   3.1 Format for WinRM CLI Examples in Document ......................................................... 15
   3.2 WS-Man Security and Time Parameters ................................................................... 16
      3.2.1 Encryption Certificate Security .......................................................................... 16
      3.2.2 Handling Invalid Responses from WS-Man Commands .................................... 16
      3.2.3 Improving WinRM Enumeration Performance .................................................. 17
      3.2.4 Limitations of WINRM ..................................................................................... 17
      3.2.5 Specifying Time using WSMAN .......................................................................... 17
      3.2.6 Specifying Integer values in WSMAN commands ............................................... 18
      3.2.7 Return Values ..................................................................................................... 18
      3.2.8 Glossary ............................................................................................................ 18
4. Discovery ........................................................................................................................ 19
   4.1 Discovering Web Service Capability ......................................................................... 19
   4.2 Discovering the Implemented Profiles ....................................................................... 19
   4.3 Discovering Implementation Namespace ..................................................................... 20
5. Managing iDRAC Local User Accounts ....................................................................... 22
   5.1 Description of iDRAC Attributes Versus Standard DMTF Model ............................ 22
   5.2 Account Inventory (Using iDRAC Attributes) ............................................................ 22
      5.2.1 Account and Capabilities (Using iDRAC Attributes) ......................................... 22
      5.2.2 Privilege and Capabilities (Using iDRAC Attributes) ......................................... 23
   5.3 Manage Account Settings (Using iDRAC Attributes) ................................................ 23
      5.3.1 Modify User Name (Using iDRAC Attributes) .................................................... 24
      5.3.2 Modify Password (Using iDRAC Attributes) ....................................................... 24
      5.3.3 Modify Account State (Using iDRAC Attributes) ............................................... 25
      5.3.4 Modify User Privilege (Using iDRAC Attributes) .............................................. 26
   5.4 Account Inventory (Using DMTF Model) .................................................................... 27
      5.4.1 Account and Capabilities (Using DMTF Model) .................................................. 27
      5.4.2 Privilege and Capabilities (Using DMTF Model) ................................................. 29
   5.5 Manage Account Settings (Using DMTF Model) ....................................................... 31
      5.5.1 Modify User Name (Using DMTF Model) ............................................................ 31
      5.5.2 Modify Password (Using DMTF Model) ............................................................... 34
      5.5.3 Modify Account State (Using DMTF Model) ....................................................... 34
      5.5.4 Modify User Privilege (Using DMTF Model) ...................................................... 35
6. Firmware Inventory ....................................................................................................... 36
   6.1 Software Inventory Profile Specification ................................................................. 36
   6.2 Remote Inventory Method Invocation — Get Software Inventory ............................ 36
7. Firmware Update ............................................................................................................ 38
7.1 Software Update Profile Specification ................................................................. 38
7.2 “Rollback” Firmware ............................................................................................... 38
7.2.1 Request “Rollback” Image .................................................................................. 38
7.2.2 Create Reboot Job .............................................................................................. 38
7.2.3 Schedule Update Jobs ......................................................................................... 38
7.2.4 Monitor Update Jobs .......................................................................................... 38
7.3 BIOS Firmware “Rollback” ...................................................................................... 39
7.4 NIC Firmware “Rollback” ....................................................................................... 40
7.5 Update from Network Source .................................................................................. 41
7.5.1 Request Update Download ............................................................................... 41
7.5.2 Monitor Download Status ................................................................................ 41
7.5.3 Reboot to Perform Update ................................................................................ 42
7.5.4 Wait for Job Completion ................................................................................... 42
7.5.5 Delete Job .......................................................................................................... 42
7.6 Update NICs from HTTP, CIFS Share, NFS Share, TFTP, or FTP ......................... 42
7.7 Update BIOS from HTTP, CIFS Share, NFS Share, TFTP, or FTP ....................... 44
7.8 CreateRebootJob() ............................................................................................... 45
7.9 Automatic Updates ............................................................................................... 46
7.9.1 Enable Automatic Update ................................................................................ 46
7.9.2 Create a Config Job .......................................................................................... 47
7.9.3 Set Update Schedule ......................................................................................... 47
7.9.4 Get the Update Schedule .................................................................................. 48
7.9.5 Clear the Update Schedule .............................................................................. 48
7.10 Device Update from Repository .......................................................................... 49
7.10.1 Install from Repository ................................................................................... 49
7.10.2 Get Repo-Based Update List .......................................................................... 50
8 Power State Management ......................................................................................... 52
8.1 Description of Base Server Versus Power State Management Methods .............. 52
8.2 Get Power State ..................................................................................................... 52
8.2.1 Base Server Method ....................................................................................... 52
8.2.2 Power State Management Method .................................................................... 53
8.3 Get Power Control Capabilities .......................................................................... 53
8.3.1 Base Server Method ....................................................................................... 53
8.3.2 Power State Management Method .................................................................... 55
8.4 Power Control ...................................................................................................... 56
8.4.1 Base Server Method ....................................................................................... 56
8.4.2 Power State Management Method .................................................................... 56
9 Hardware Inventory ................................................................................................. 58
9.1 Power Supply Inventory ....................................................................................... 58
9.2 Fan Inventory ........................................................................................................ 59
9.3 Memory Inventory ................................................................................................ 60
9.4 CPU Inventory  ................................................................. 61
9.5 iDRAC Card Inventory ....................................................... 62
9.6 PCI Device Inventory ....................................................... 63
9.7 Video Inventory ............................................................... 64
9.8 VFlash SD Card Inventory ................................................. 65
9.9 NIC Inventory and Configuration ........................................ 65
9.10 RAID Inventory and Configuration ....................................... 67
9.11 BIOS Inventory and Configuration ....................................... 69
9.12 System Inventory (Including CSIOR Attribute) ...................... 70
9.13 USB Device Inventory ....................................................... 72
9.14 System Quick Sync Hardware Inventory .............................. 72
9.15 PCIe SSD Inventory .......................................................... 72
9.16 PCIe SSD Extender Inventory ............................................. 72
9.17 PCIe SSD Backplane Inventory .......................................... 72
9.18 Host Network Interface Inventory ...................................... 72
9.19 Physical Computer System Inventory .................................. 72

10 Job Control Management .................................................... 73
10.1 Description of Job Management ......................................... 73
10.2 Remote Job Control Examples .......................................... 73
10.2.1 Setup Job Queue ........................................................ 73
10.2.2 Delete Job Queue ........................................................ 74
10.2.3 List Jobs in Job Store ................................................... 75

11 Operating System Deployment ............................................. 77
11.1 OS Deployment Profile Implementation Conformance ............. 77
11.2 Checking OS Deployment Service Availability ..................... 77
11.3 OS Deployment Method Invocation Examples ....................... 77
11.3.1 Get Driver Pack Information ........................................ 77
11.3.2 Unpack Selected Drivers and Attach to Host OS as USB Device 78
11.3.3 Detach Emulated USB Device Containing Drivers ............. 79
11.3.4 Unpack Selected Drivers and Copy to Network Share .......... 80
11.3.5 Check Job Status ....................................................... 80
11.3.6 Boot to Network ISO .................................................. 81
11.3.7 Detach Network ISO USB Device .................................. 82
11.3.8 Boot To PXE ............................................................ 82
11.3.9 Get Host MAC Address Information .............................. 83
11.3.10 Download ISO to VFlash .......................................... 83
11.3.11 Boot to ISO from VFlash .......................................... 84
11.3.12 Delete ISO from VFlash ........................................... 85
11.3.13 Detach ISO from VFlash .......................................... 85
11.3.14 Connect Network ISO Image ..................................... 86
11.3.15 Disconnect Network ISO Image ................................. 87
12 Lifecycle Controller Management Profile ................................................................. 92
  12.1 Collect System Inventory on Restart (CSIOR) ..................................................... 92
  12.2 Part Replacement Configuration and Management .............................................. 93
    12.2.1 Create Config Job ................................................................................... 93
    12.2.2 Get LC Config Job Status ......................................................................... 94
    12.2.3 List All LC Jobs ..................................................................................... 94
    12.2.4 Get CSIOR Component Configuration Recovery (CCR) Attribute ............... 95
    12.2.5 Get Part Firmware Update Attribute ....................................................... 96
  12.3 Re-Initiate Auto-Discovery Client ................................................................. 96
  12.4 Clear or Set Provisioning Server .................................................................. 97
  12.5 Check vFlash License Enablement .................................................................. 98
  12.6 Download Server Public Key ......................................................................... 99
  12.7 Download Client Certificates ................................................................... 100
  12.8 Delete Auto-Discovery Client Certificates .................................................. 101
  12.9 Set Public Certificates ................................................................................ 101
  12.10 Set iDRAC Certificate and Private Key ....................................................... 102
  12.11 Delete Auto-Discovery Server Public Key ................................................ 103
  12.12 Insert Comment in Lifecycle Controller Log ............................................. 103
  12.13 Export Lifecycle Controller Log ................................................................. 104
  12.14 Export Complete Lifecycle Log ................................................................. 105
  12.15 Export Hardware Inventory from Lifecycle Controller ................................ 106
  12.16 Export Factory Configuration .................................................................. 107
  12.17 System Decommission ............................................................................... 108
  12.18 Get Remote Services API Status .................................................................. 108
  12.19 Export System Configuration .................................................................. 109
  12.20 Import System Configuration .................................................................. 109
  12.21 Import System Configuration Preview ...................................................... 110
  12.22 Remote Diagnostics .................................................................................. 111
    12.22.1 Run Diagnostics .................................................................................. 111
    12.22.2 Export Diagnostics Results .................................................................. 112
    12.22.3 Verify the Diagnostics Job Status ....................................................... 112
  13 VFlash SD Card Management ............................................................................ 114
    13.1 Listing the SD Card Partitions .................................................................. 114
    13.2 Initialize the Virtual Flash Media ............................................................. 115
13.2.1 Get vFlash SD Card Inventory .................................................. 115
13.2.2 Initialize or Format Media .................................................. 115
13.2.3 Verify Initialization or Formatting ....................................... 116
13.3 Enable or Disable vFlash using vFlash State Change .................. 116
13.4 Create Partition ...................................................................... 117
13.5 Create Partition using Image .................................................. 118
13.6 Delete Partition ....................................................................... 120
13.7 Format Partition ....................................................................... 121
13.8 Modify Partition ....................................................................... 121
13.9 Attach Partition ....................................................................... 122
13.10 Detach Partition ..................................................................... 123
13.11 Export Data from Partition ..................................................... 124

14 Boot Control Configuration Management .................................. 126
14.1 Listing the Boot Inventory-ConfigSetting Class ......................... 126
14.2 Getting a Boot ConfigSetting Instance .................................... 127
14.3 Listing the Boot Inventory-SourceSetting Class ......................... 127
14.4 Changing the Boot Order by InstanceIDChangeBootOrderByID() .................................................. 128
14.5 Enable or Disable the Boot Source-ChangeBootSourceState() .................. 129

15 NIC or CNA Card Management .................................................. 130
15.1 Listing the NIC or CNA Inventory-Enumeration Class ................ 130
15.2 Listing the NIC or CNA Inventory-String Class .......................... 132
15.3 Listing the CNA Inventory-Integer Class ................................. 133
15.4 Listing the CNA Inventory-NICView Class .............................. 135
15.5 Listing the CNA Inventory-NICCapabilities Class .................... 136
15.6 Listing the CNA Inventory- NICStatistics Class ......................... 138
15.7 Applying the Pending Values for CNA-CreateTargetedConfigJob() .................................................. 139
15.8 Deleting the Pending Values for CNADeletePendingConfiguration() .................................................. 140
15.9 Getting the CNA Enumeration Instance ................................... 140
15.10 Setting the IscsiOffloadMode Attribute .................................. 141
15.11 Setting the MaxBandwidth Attribute ...................................... 142
15.12 Setting the VirtMacAddr Attribute ......................................... 143
15.13 Setting the LegacyBootProto Attribute .................................. 144
15.14 Setting CNA LAN Modes ....................................................... 145
15.15 Setting the iSCSI Boot Target ................................................ 146
15.16 Setting the FCoE Boot Target ................................................ 147

16 RAID Storage Management ....................................................... 149
16.1 Listing the RAID Inventory-Enumeration Class ......................... 149
16.2 Getting a RAID Enumeration Instance .................................... 150
16.3 Listing the RAID Inventory-Integer Class ............................... 150
16.4 Getting a RAID Integer Instance ............................................ 152
16.5 Listing the RAID Inventory-String Class .................................. 152
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>16.6</td>
<td>Getting a RAID String Instance</td>
</tr>
<tr>
<td>16.7</td>
<td>Listing the RAID Inventory-ControllerView Class</td>
</tr>
<tr>
<td>16.8</td>
<td>Getting a RAID ControllerView Instance</td>
</tr>
<tr>
<td>16.9</td>
<td>Listing the RAID Inventory-PhysicalDiskView Class</td>
</tr>
<tr>
<td>16.10</td>
<td>Listing the RAID VirtualDiskView Inventory</td>
</tr>
<tr>
<td>16.11</td>
<td>Listing the RAID EnclosureView Inventory</td>
</tr>
<tr>
<td>16.12</td>
<td>Listing the Controller Batteries</td>
</tr>
<tr>
<td>16.13</td>
<td>Listing the Enclosure EMM</td>
</tr>
<tr>
<td>16.14</td>
<td>Listing the Enclosure PSU</td>
</tr>
<tr>
<td>16.15</td>
<td>Reset Configuration-ResetConfig()</td>
</tr>
<tr>
<td>16.16</td>
<td>Clearing the Foreign Configuration-ClearForeignConfig()</td>
</tr>
<tr>
<td>16.17</td>
<td>Applying the Pending Values for RAID-CreateTargetedConfigJob()</td>
</tr>
<tr>
<td>16.18</td>
<td>Deleting the Pending Values for RAIDDeletePendingConfiguration()</td>
</tr>
<tr>
<td>16.19</td>
<td>Managing Hot Spare</td>
</tr>
<tr>
<td>16.19.1</td>
<td>Determining Potential Disks-GetDHSDisks()</td>
</tr>
<tr>
<td>16.19.2</td>
<td>Assigning the Hot Spare-AssignSpare()</td>
</tr>
<tr>
<td>16.19.3</td>
<td>Unassigning the Hot Spare-UnassignSpare()</td>
</tr>
<tr>
<td>16.20</td>
<td>Managing Keys for Self Encrypting Drives</td>
</tr>
<tr>
<td>16.20.1</td>
<td>Setting the Key-SetControllerKey()</td>
</tr>
<tr>
<td>16.20.2</td>
<td>Locking the Virtual Disk-LockVirtualDisk()</td>
</tr>
<tr>
<td>16.20.3</td>
<td>Locking the Controller with a Key-EnableControllerEncryption()</td>
</tr>
<tr>
<td>16.20.4</td>
<td>Rekeying the Controller-ReKey()</td>
</tr>
<tr>
<td>16.20.5</td>
<td>Removing the Key-RemoveControllerKey()</td>
</tr>
<tr>
<td>16.21</td>
<td>Managing Virtual Disk</td>
</tr>
<tr>
<td>16.21.1</td>
<td>Getting the Available RAID levels-GetRAIDLevels()</td>
</tr>
<tr>
<td>16.21.2</td>
<td>Getting the Available Disks-GetAvailableDisks()</td>
</tr>
<tr>
<td>16.21.3</td>
<td>Checking the Create VD Parameters Validity-CheckVDValues()</td>
</tr>
<tr>
<td>16.21.4</td>
<td>Creating a Single Virtual Disk-CreateVirtualDisk()</td>
</tr>
<tr>
<td>16.21.5</td>
<td>Creating a Sliced Virtual Disk-CreateVirtualDisk()</td>
</tr>
<tr>
<td>16.21.6</td>
<td>Creating a Cachecode Virtual Disk-CreateVirtualDisk()</td>
</tr>
<tr>
<td>16.21.7</td>
<td>Deleting a Virtual Disk-DeleteVirtualDisk()</td>
</tr>
<tr>
<td>16.21.8</td>
<td>Rename Virtual Disk – RenameVD()</td>
</tr>
<tr>
<td>16.21.9</td>
<td>Cancel background initialization -CancelBackgroundInitialization()</td>
</tr>
<tr>
<td>16.21.10</td>
<td>Rebuild physical disk -RebuildPhysicalDisk()</td>
</tr>
<tr>
<td>16.21.11</td>
<td>Cancel rebuild physical disk -CancelRebuildPhysicalDisk()</td>
</tr>
<tr>
<td>16.21.12</td>
<td>Online capacity expansion -OnlineCapacityExpansion()</td>
</tr>
<tr>
<td>16.21.13</td>
<td>RAID level migration -RAIDLevelMigration()</td>
</tr>
<tr>
<td>16.21.14</td>
<td>Set enclosure asset name -SetAssetName()</td>
</tr>
<tr>
<td>16.21.15</td>
<td>Set enclosure asset tag -SetAssetTag()</td>
</tr>
<tr>
<td>16.22</td>
<td>Setting Controller Attributes</td>
</tr>
<tr>
<td>16.22.1</td>
<td>Changing the Value of a RAID Controller Enumeration Attribute</td>
</tr>
</tbody>
</table>
19.4.2 Polling Job Completion ................................................................. 210
19.4.3 Set Attribute Verification ............................................................. 211

19.5 Listing the iDRAC Card Inventory-Integer Class .................................. 212
19.6 Listing the iDRAC Card Inventory-Integer Class using groupID ................ 213
19.7 Listing the iDRAC Card Inventory-String Class ..................................... 214
19.8 Listing the iDRAC Card Inventory-String Class using groupID ................ 215
19.9 Changing the iDRAC IP Change Notification ....................................... 217
19.9.1 Getting the Current iDRAC IPChange State .................................... 217
19.9.2 Setting the iDRAC IPChange Notification-SetAttribute() ..................... 217

19.10 Reset to Defaults keep Network and users ...................................... 218

20 Remote Service Status ........................................................................ 219
20.1 Getting Remote Service Status .......................................................... 219
20.2 Restarting Remote Service Status ....................................................... 220

21 System Information ............................................................................ 221
21.1 Listing the System Inventory-SystemView Class .................................... 221

22 Sensor Information ............................................................................. 223
22.1 Listing the Sensors Inventory-PSNumericSensor Class .......................... 223

23 Managing Fiber Channel (FC) Configuration ....................................... 225
23.1 Listing the FC Inventory-Attribute Class ............................................ 225
23.2 Listing the FC Inventory-Statistics Class ............................................ 226
23.3 Listing the FC Inventory-String Class .................................................. 227
23.4 Listing the FC Inventory-Integer Class ................................................ 227
23.5 Listing the FC Inventory-Enumeration Class ....................................... 228
23.6 Changing the FC Attributes-SetAttribute() ...................................... 228
23.7 Applying the Pending Values for FC-CreateTargetedConfigJob() .............. 229
23.8 Deleting the Pending Values for FC-DeletePendingConfiguration() ............ 230
23.9 Listing the FC Views ......................................................................... 231

24 Support for HTTP and HTTPS Network Shares in 14G ............................ 232
25 Support for file transfer without need of network share in 14G .................. 233
25.1 ImportData(): .................................................................................. 233
25.2 ExportData(): .................................................................................. 234
25.3 ClearTransferSession(): .................................................................. 235

26 GroupManager Configuration ................................................................ 237
26.1 Enabling GroupManger Feature ......................................................... 237
26.2 To View GroupName and GroupUUID Details ..................................... 237
26.3 To Join Group with existing master group .......................................... 239
26.4 To Remove Self from Group .............................................................. 239
26.5 To Delete the Master Group .............................................................. 240

27 System configure Lock down mode ...................................................... 241
27.1 Enabling the System Configure Lock down mode ................................ 241

28 Connection View ................................................................................. 242
28.1 Enabling the Connection View feature ................................................................. 242
28.2 Connection View Instances: .............................................................................. 242
28.3 Refresh the connection view instance .............................................................. 243

29 Secure Boot Configuration ..................................................................................... 244
29.1 Enabling Secure Boot configuration ................................................................. 244
29.2 View Secure boot Certificate .......................................................................... 245
29.3 Import boot Certificate .................................................................................... 245
29.4 Export boot Certificate ..................................................................................... 247
29.5 Delete boot Certificate ...................................................................................... 248
29.6 Reset boot Certificate ....................................................................................... 249

30 SupportAssist .......................................................................................................... 250
30.1 SupportAssistRegister ....................................................................................... 250
30.2 SupportAssistAcceptEULA .............................................................................. 251
30.3 SupportAssistGetEULASTatus ......................................................................... 251
30.4 SupportAssistSetAutoCollectSchedule ............................................................ 252
30.5 SupportAssistGetAutoCollectSchedule ............................................................ 252
30.6 SupportAssistClearAutoCollectSchedule ........................................................ 253
30.7 SupportAssistCollection .................................................................................... 253
30.8 SupportAssistExportLastCollection ................................................................. 254
30.9 SupportAssistUploadLastCollection ................................................................ 255
Executive Summary

Dell PowerEdge servers are equipped with the integrated Dell Remote Access Controller and the Lifecycle Controller solution for remote management—iDRAC6 on 11th generation servers, iDRAC7 on 12th generation servers, iDRAC8 on 13th Generation servers and iDRAC9 on 14th generation servers. These servers can be remotely managed by using the WS-Man services for configuration, update, deployment, and maintenance. This whitepaper describes the various WS-Man interfaces with examples about using different interfaces in a Windows environment.
1 Introduction

This document serves as a guideline for utilizing the functionality available from embedded Lifecycle Controller Remote Enablement Web Services interfaces. The purpose of this document is to provide information and examples for utilizing the Web services for Management (WS-Man) management protocol using Windows WinRM and open source WSMANCLI command line utilities. Examples and invocation information is provided for the following functionality.

- Inventory for BIOS, component firmware and embedded Software
- Update of BIOS, component firmware and embedded Software
- Job Control of update tasks
- Enhancement of Operating System Deployment using VFlash SD Card
- Enhancement of Discovery and Handshake from LifeCycle Controller 1.x
- RAID configuration management
- iDRAC Inventory and configuration features
- NIC configuration management
- Boot configuration management
- BIOS configuration management
- Backup & Restore of Server Profiles
- Automatic scheduled backup of server profiles
- Automatic scheduled firmware updates
- XML based Server Configuration
- WS-Eventing for event notifications

The following features have been added in the 13th Generation of Dell PowerEdge Servers.

- Monitor Out-of-band performance
- Enhanced Security using hash password
- View logs, monitoring and server information, and configure the network parameters of a server using a mobile device
- Delete server-related information
- View the managed System using Physical Computer System View
- Manage the Web Server Certificate
- Configure the USB Management Port
- View Embedded Tech Support Report to resolve business-critical issues
- Configure Storage Devices in Real time
- Support PCIe SSD Devices
- Support for 12Gbps SAS HBAs and PERC 9.1 Controllers

Update iDRAC and Lifecycle Controller firmware using a single firmware image

The target audience for this document is application and script writers who want to utilize the remote management capabilities using WS-Man protocol available from Dell Lifecycle Controller.
References

Lifecycle Controller Wiki www.delltechcenter.com/LC

Dell 12th generation PowerEdge server resources:
http://www.delltechcenter.com/12thGen

Dell CIM Profiles:
http://www.delltechcenter.com/page/DCIM.Library.Profile

Managed Object Format (MOF) files:
http://www.delltechcenter.com/page/DCIM.Library.MOF

WinRM Scripting API, MSDN:

Openwsman CLI:
http://www.openwsman.org/project/wsmanci

DMTF Common Information Model (CIM) Infrastructure Specification (DSP0004):
http://www.dmtf.org/standards/published_documents/DSP0004_2.5.0.pdf

List of PCI IDs:
http://pciids.sourceforge.net/pci.ids
Overview

The remote interface guidelines provided in this document are illustrated by command line interface (CLI) examples of the WS-Man protocol Web services APIs that expose the remote management capabilities of the Dell Lifecycle Controller. The command line examples are from the Microsoft® Windows® and Linux environments using WinRM and WSMANCLI respectively. The Lifecycle Controller remote management capabilities are organized by management domain and documented in Dell CIM Profile specifications. The remote enablement feature for Lifecycle Controller 2.0 provides the following capabilities:

- Remotely get inventory of the BIOS, component firmware, and embedded Software including version information of both the installed as well as available cached versions
- Remote update of BIOS, component firmware, Diagnostic content, DRAC content, driver pack, power supplies from remotely located Dell Update Packages or cached images located in the Lifecycle Controller
- Remotely schedule and track the status of update tasks (jobs)
- Remotely manage the Part Replacement feature by allowing retrieving and setting auto update and auto System inventory sync
- Enable re-initiation of Lifecycle Controller Auto-Discovery feature
- Enhancement of Operation System Deployment capabilities by supporting the downloading of an ISO image to a Dell VFlash SD Card and booting to the ISO image on the VFlash SD Card
- NIC configuration enables the ability to get and set NIC attributes that are configurable using NIC Option ROM or NIC UEFI HII.
- Remote RAID configuration allows users to remotely query and configure the Hardware Raid of the System
- Multiple HW Inventory views allows users to remote query the inventory of Hardware
- Backup & Restore of Server Profiles and automatic scheduling of backups
- Automatic scheduled firmware updates
- XML based server configuration
- WS-Eventing for event notifications

The following features have been added in the 13th Generation of Dell PowerEdge Servers.

- Monitor Out-of-band performance
- Enhanced Security using hash password
- View logs, monitoring and server information, and configure the network parameters of a server using a mobile device
- Delete server-related information
- View the managed System using Physical Computer System View
- Manage the Web Server Certificate
- Configure the USB Management Port
- View Embedded Tech Support Report to resolve business-critical issues
- Configure Storage Devices in Real time
- Support PCIe SSD Devices
- Support for 12Gbps SAS HBAs and PERC 9.1 Controllers
- Update iDRAC and Lifecycle Controller firmware using a single firmware image

3.1 Format for WinRM CLI Examples in Document

The examples of WinRM and WSMANCLI command line invocations in this document are formatted for readability and often span multiple lines in the document. In actual use, scripted or hand-typed invocations are contained on one line. The examples also use substitute values for the target iDRAC IP address, username (with ExecuteServerCommand privilege), password and other site specific information. Actual use of these examples would require using values for IP Address, username and password, etc. that are valid. These values are represented in the examples as follows:

Target iDRAC IP address = [IPADDRESS]
iDRAC Username = [USER]
iDRAC Password = [PASSWORD]

Additional substitute values are used in some of the examples and are described in the specific example. The following example is typical of the formatting used in this document:

**EXAMPLE:**
```
winrm e cimv2/root/dcim/DCIM_OSDeploymentService
-u: [USER] -p: [PASSWORD]
-r: https://[IPADDRESS]/wsman:443
-encoding: utf-8 -a:basic
```

### 3.2 WS-Man Security and Time Parameters

#### 3.2.1 Encryption Certificate Security

For the WinRM examples provided in this document, the strict checks of certificates such as matching of CNs (Common Names) and verification with the actual CA (Certificate Authority) of the certificate of the WS-Management protocol HTTPS encryption certificate is assumed to be already configured and enabled. To disable the strict certificate checking, add the following command line options to all WinRM examples:
```
skipCACheck and -skipCNCheck.
```

Additionally, the following error may result if the end point does not support this feature. Use the switch `skiprevocationcheck` to bypass this error.

WSManFault

**Message** = The server certificate on the destination computer (10.35.0.232:443) has the following errors:

The SSL certificate could not be checked for revocation. The server used to check for revocation might be unreachable.

Refer to the WinRM documentation [4] and related documentation for directions on setting up encryption certificates for WinRM and executing WinRM invocations using full security capabilities. Refer to the Lifecycle Controller User Guide [1] for directions on configuring different encryption certificates for the iDRAC Web server. Dell recommends that the full security and encryption capabilities of the WSManagement protocol is used for production level utilization of the Lifecycle Controller Web services interfaces.

**Note**: If the Operating System Date is changed to a date beyond the iDRAC SSL Certificate Expiry Date, WSMAN sessions will start failing because the certificates will be marked as expired. A new certificate will have to be generated using “racadm resetssicfg” command on the iDRAC.

#### 3.2.2 Handling Invalid Responses from WS-Man Commands

WSMAN commands can fail or time out due to various factors – network availability, the stress levels on the iDRAC, incorrect URI, incorrect user name and passwords, insufficient privileges, Software errors, insufficient timeout value set within the console application etc.

Upon receiving errors, the user is recommended to:

- Check the network connectivity between the management console and the iDRAC
- Check the WS-Man syntax to ensure there are no typos in the command line
- Check if there are other WS-Man commands sending from other Systems Wait for a few seconds and re-try the WSMAN command

WSMAN Timeouts typically manifest themselves with the following error messages:

- “The WS-Management service cannot complete the operation within the time specified in OperationTimeout"
• “WinRM cannot complete the operation. Verify that the specified computer name is valid, that the computer is accessible over the network, and that a firewall exception for the WinRM service is enabled and allows access from this computer”
• “The WS-Management service cannot process the request. The request contains one or more invalid SOAP headers”
• Upon receiving a time out error, the user is recommended to:
  • Check the network connectivity between the management console and the iDRAC
  • Attempt a retry after 5 seconds upon receiving a timeout error on sending a WSMAN command.

3.2.3 Improving WinRM Enumeration Performance
When an enumeration command is executed, the default WinRM configuration gets only 20 instances at a time and therefore slows down the System drastically. Changing the WinRM configuration to allow a greater number, such as 50, will reduce the time taken by the enumeration operations.

Execute the following command to get instances in groups of up to 50.

```
winrm set winrm/config @{MaxBatchItems="50"}
```

Additionally, increasing the allotted maximum envelope size and timeout can also increase performance.

```
winrm set winrm/config @{MaxEnvelopeSizekb="150"}
```

Other optional WinRM configuration commands are listed below for convenience. To get the current WinRM configuration settings, execute the following command.

```
winrm g winrm/config
```

By default, the client computer requires encrypted network traffic. To allow the client computer to request unencrypted traffic, execute the following command:

```
winrm s winrm/config/Client @{AllowUnencrypted="true"}
```

TrustedHosts is an array that specifies the list of remote computers that are trusted. Other computers in a workgroup or computers in a different domain should be added to this list.

**Note:** The computers in the TrustedHosts list are not authenticated.

Execute the following command to allow all computers to be included in TrustedHosts.

```
winrm s winrm/config/Client @{TrustedHosts="*"}
```

Basic authentication is a scheme in which the user name and password are sent in clear text to the server or proxy. This method is the least secure method of authentication. The default is True.

Execute the following command to set client computer to use Basic authentication.

```
winrm s winrm/config/Client/Auth @{Basic="true"}
```

3.2.4 Limitations of WINRM
WINRM tool sets a default response timeout of 60 seconds for all WSMAN commands it sends out. WINRM does not allow for varying this timeout interval.

Using “winrm set winrm/config @{MaxTimeoutms ="80000"}” does not actually change the timeout due to a bug in the WINRM tool. It is therefore recommended to not use WINRM for commands that may take more than 1 minute to complete execution. **EXAMPLE:** Enumerate of Lifecycle Logs typically takes 90 seconds or more to complete.

The use of libraries that create SOAP-XML packets is highly recommended since the user can configure the timeout duration using these libraries.

3.2.5 Specifying Time using WSMAN
The several methods that attach a virtual USB device to the target System accept a StartTime and Until parameter. The parameter data type is CIM date-time. If the StartTime parameter is null the action will not be started. If the Until parameter is null, the default value will be 17 hours. The date-time data type is defined in the CIM Infrastructure Specification 4 as:

```
ddddddhhmmss.mmmmmmm
```
Where:
- ddddddd is the number of days
- hh is the remaining number of hours
- mm is the remaining number of minutes
- ss is the remaining number of seconds
- mmmmmm is the remaining number of microseconds

The Lifecycle controller firmware update, and set attribute related methods that require a date time parameter, use the form YYYYMMDDhhmmss (Eg. 20090930112030). The user is expected to enter the date and time in this format for all Lifecycle Controller updates and set attribute tasks. TIME_NOW is a special value that represents “running the tasks immediately”.

Note: When attempting to change the RAC time by using ManageTime() WSMAN API, if the time being set is greater than the current RAC Time, the WSMAN operation returns a TIMEOUT even though the operation may be successful.

3.2.6 Specifying Integer values in WSMAN commands

When specifying Integer values using WSMAN, refrain from using empty string as input as it internally gets converted to 0.

EXAMPLE: ShareType is an Integer property used with WSMAN APIs such as ExportSystemConfiguration. Specifying ShareType="" will internally convert it to “0” which refers to NFS shares.

3.2.7 Return Values

Many of the methods in this document have the following possible return values. They are summarized here for convenience.

- 0 = Success
- 1 = Not Supported
- 2 = Failed
- 4096 = Job Created

3.2.8 Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOS</td>
<td>Basic Input / Output System</td>
</tr>
<tr>
<td>HW</td>
<td>Hardware</td>
</tr>
<tr>
<td>DRAC</td>
<td>Integrated DELL Remote Access Controller</td>
</tr>
<tr>
<td>IPL</td>
<td>Initial Program Load</td>
</tr>
<tr>
<td>DUP</td>
<td>Dell Update Package</td>
</tr>
<tr>
<td>MOF</td>
<td>Managed Object File</td>
</tr>
<tr>
<td>CIM</td>
<td>Common Information Model</td>
</tr>
<tr>
<td>NIC</td>
<td>Network Interface Controller</td>
</tr>
<tr>
<td>RAID</td>
<td>Redundant Array of Independent Disks</td>
</tr>
<tr>
<td>FQDD</td>
<td>Fully Qualified Device Description</td>
</tr>
<tr>
<td>UEFI</td>
<td>Unified Extensible Firmware Interface</td>
</tr>
<tr>
<td>AMEA</td>
<td>Advanced Management Enablement Adapter</td>
</tr>
<tr>
<td>HII</td>
<td>Human Interface Infrastructure</td>
</tr>
<tr>
<td>WS-Man</td>
<td>WS-Management is a specification of a SOAP-based protocol for the management of servers, devices, applications and more</td>
</tr>
</tbody>
</table>
4 Discovery

4.1 Discovering Web Service Capability

Determine if the target System supports the WinRM interface using the ‘identify’ command.

Profiles:
http://www.dmtf.org/sites/default/files/standards/documents/DSP0217_2.0.0.pdf

EXAMPLE:
winrm identify
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCAcheck
-encoding:utf-8 -a:basic

OUTPUT:
IdentifyResponse
	ProductVendor = Openwsman Project
	ProductVersion = 2.2.4

4.2 Discovering the Implemented Profiles

Implemented profiles are advertised using the class CIM_RegisteredProfile. Enumerate this class in the “root/interop” CIM namespace.

Profiles:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1033_1.0.0.pdf

EXAMPLE:
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_RegisteredProfile?__cimnamespace=root/interop
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCAcheck
-encoding:utf-8 -a:basic

OUTPUT:
DCIM_LCRegisteredProfile
	AdvertiseTypeDescriptions = WS-Identify, Interop Namespace
	AdvertiseTypes = 1, 1
	InstanceID = DCIM:Memory:1.0.0
	OtherRegisteredOrganization = DCIM
	RegisteredName = Memory
	RegisteredOrganization = 1 RegisteredVersion = 1.0.0 ...
	DCIM_RegisteredProfile
	AdvertiseTypeDescriptions = WS-Identify
	AdvertiseTypes = 1
	Caption = null
	Description = null
	ElementName = null
InstanceID = DCIM:CSRegisteredProfile:1
OtherRegisteredOrganization = null
RegisteredName = Base Server
RegisteredOrganization = 2
RegisteredVersion = 1.0.0

The above example shows that the DMTF Base Server profile version 1.0.0 is implemented.

4.3 Discovering Implementation Namespace

The implementation CIM namespace may be discovered from the interop (root/interop) CIM namespace using the class CIM_ElementConformsToProfile that associates an instance of CIM_RegisteredProfile class with an instance of CIM_ComputerSystem class.

Profiles: N/A

EXAMPLE:

```sh
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/*
   -dialect:association -associations
   -filter: {object=DCIM_ComputerSystem?CreationClassName=DCIM_ComputerSystem+
      Name=srv:System,__cimnamespace=root/dcim}
   -u:[USER] -p:[PASSWORD]
   -r:https://[IPADDRESS]/wsman -encoding:utf-8 -a:basic
   -SkipCNcheck -SkipCAcheck
```

OUTPUT:

```xml
DCIM_CSRoleLimitedToTarget
DefiningRole
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
   ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Role
SelectorSet
   Selector: CreationClassName = DCIM_Role,
      Name = DCIM:Role:9,__cimnamespace = root/dcim
TargetElement Address =
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
   ResourceURI =
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_ComputerSystem
SelectorSet
   Selector: CreationClassName = DCIM_ComputerSystem,
      Name = srv:System,__cimnamespace = root/dcim DCIM_CSRoleLimitedToTarget
DefiningRole
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
   ResourceURI =
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Role
SelectorSet
   Selector: CreationClassName = DCIM_Role,Name = DCIM: Role:10,
      __cimnamespace = root/dcim TargetElement
Address =
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
```
ResourceURI =
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_ComputerSystem
SelectorSet
Selector: CreationClassName = DCIM_ComputerSystem,
Name = srv:System, __cimnamespace = root/dcim
5 Managing iDRAC Local User Accounts

5.1 Description of iDRAC Attributes Versus Standard DMTF Model

The iDRAC user account management data model is represented by both DMTF and Dell Profiles. Both models are currently offered. The DMTF Profiles for Simple Identity Management and Role Based Authorization represent iDRAC user accounts and privileges. The DMTF data model is complex and typically requires multiple transactions to accomplish simple operations such as specifying a username and password or giving a user account admin privileges. For this reason, LC also offers a Dell data model for managing iDRAC user accounts that is based on an attribute model. The DCIM iDRAC Card Profile specifies the attributes for each user account name, password, and privilege. The iDRAC has 15 local user account that can be managed.

5.2 Account Inventory (Using iDRAC Attributes)

The list of user accounts may be retrieved by enumerating the DCIM_iDRACCard classes. The class provides the user account name and enabled state properties.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

5.2.1 Account and Capabilities (Using iDRAC Attributes)

Enumerating the DCIM_iDRACCardEnumeration class, Section 19.1, and parsing the output for the attribute AttributeDisplayName = User Admin Enable, will display all of the 16 possible user accounts and their respective status.

EXAMPLE:

-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 a:basic

OUTPUT:

DCIM_iDRACCardEnumeration
   AttributeDisplayName = User Admin Enable
   AttributeName = Enable
  CurrentValue = Disabled
   DefaultValue = Disabled
   Dependency = null
   DisplayOrder = 0
   FQDD = iDRAC.Embedded.1
   GroupDisplayName = Users
   GroupID = Users.1
   InstanceID = iDRAC.Embedded.1#Users.1#Enable
   IsReadOnly = true
   PossibleValues = Disabled, Enabled

DCIM_iDRACCardEnumeration
   AttributeDisplayName = User Admin Enable
   AttributeName = Enable
   CurrentValue = Enabled
5.2.2 Privilege and Capabilities (Using iDRAC Attributes)

Enumerating the DCIM_iDRACCardEnumeration class, Section 19.1, and parsing the output for the attribute `AttributeDisplayName = User Admin IPMI LAN(or Serial) Privilege`, will display all of the 16 possible user accounts and their respective status.

**EXAMPLE:**

```plaintext
DCIM_iDRACCardEnumeration
AttributeDisplayName = User Admin IPMI LAN Privilege
AttributeName = IpmiLanPrivilege
CurrentValue = NoAccess
DefaultValue = NoAccess
Dependency = null
DisplayOrder = 0
FQDD = iDRAC.Embedded.1
GroupDisplayName = Users
GroupID = Users.1
InstanceID = iDRAC.Embedded.1#Users.1#IpmiLanPrivilege
IsReadOnly = true
PossibleValues = User, Operator, Administrator, NoAccess
```

```plaintext
DCIM_iDRACCardEnumeration
AttributeDisplayName = User Admin IPMI Serial Privilege
AttributeName = IpmiSerialPrivilege
CurrentValue = NoAccess
DefaultValue = NoAccess
Dependency = null
DisplayOrder = 0
FQDD = iDRAC.Embedded.1
GroupDisplayName = Users
GroupID = Users.1
InstanceID = iDRAC.Embedded.1#Users.1#IpmiSerialPrivilege
IsReadOnly = true
PossibleValues = User, Operator, Administrator, NoAccess
```

5.3 Manage Account Settings (Using iDRAC Attributes)

When the account setting capability allows, the user name of an account may be modified by invoking the `ApplyAttributes()` method on the `UserName` property. Confirmation of successful user name or password verification can be obtained by enumerating the DCIM_iDRACCardString class (Section 19.6).
5.3.1 Modify User Name (Using iDRAC Attributes)

**EXAMPLE:**

```bash
Winrm I ApplyAttributes
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:iDRACCardService
-u:[USER] -p:[PASSWORD]
-r:[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file: DracCard_UserName.xml
```

The input file, DracCard_UserName.xml, is shown below:

```xml
<p:ApplyAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService">
  <p:Target>iDRAC.Embedded.1</p:Target>
  <p:AttributeName>Users.4#UserName</p:AttributeName>
  <p:AttributeValue>HELLO</p:AttributeValue>
</p:ApplyAttributes_INPUT>
```

**OUTPUT:**

When this method is executed, a jobid or an error message is returned.

```xml
ApplyAttributes_OUTPUT
ReturnValue = 4096
Job
  EndpointReference
    Address = https://127.0.0.1:443/wsman
  ReferenceParameters
  SelectorSet
    Selector: __cimnamespace = root/dcim,InstanceID = JID_001296571842
```

5.3.2 Modify Password (Using iDRAC Attributes)

**EXAMPLE:**

```bash
winrm i ApplyAttributes
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:iDRACCardService
-u:[USER] -p:[PASSWORD]
-r:[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file: DracCard_Password.xml
```

The input file, DracCard_Password.xml, is shown below:
When this method is executed, a jobid or an error message is returned.

ApplyAttributes_OUTPUT

Job

EndpointReference

Address = https://127.0.0.1:443/wsman

ReferenceParameters


SelectorSet

Selector: __cimnamespace = root/dcim,InstanceID = JID_001296571842

ReturnValue = 4096

5.3.3 Modify Account State (Using iDRAC Attributes)

When the account setting capability allows, the user account may be enabled or disabled by invoking the method ApplyAttributes() method on the Enable property. Confirmation of the change can be obtained by enumerating the DCIM_iDRACCardString class (Section 19.6).

EXAMPLE:

winrm i ApplyAttributes
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:iDRACCardService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file: DracCard_AccountChange.xml

The input file, DracCard_AccountChange.xml, is shown below:

<ApplyAttributes_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService">
  <Target>iDRAC.Embedded.1</Target>
  <AttributeName>Users.4#Enable</AttributeName>
  <AttributeValue>Enabled</AttributeValue>
  <AttributeName>Users.4#Password</AttributeName>
  <AttributeValue>PASSWORDHERE</AttributeValue>
</ApplyAttributes_INPUT>
OUTPUT:
When this method is executed, a jobid or an error message is returned.

ApplyAttributes_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous

ReferenceParameters
ResourceURI =
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_LifecycleJob
SelectorSet
Selector: InstanceID = JID_001296744532, __cimnamespace = root/dcim
ReturnValue = 4096

The following error may result if the password has not initially been set to a value. The password may be
set an initail value at the same time as the account is enabled by adding the Users.4#Password attribute
name and corresponding attribute value, as shown above.

ApplyAttributes_OUTPUT
Message = The User Password is not configured so cannot Enable the User or
set values for IPMI Lan IPMISerial or User Admin Privilege
MessageArguments = NULL
MessageID = RAC023
ReturnValue = 2

5.3.4 Modify User Privilege (Using iDRAC Attributes)
When the account setting capability allows, the user privileges may be enabled or disabled by invoking
the method ApplyAttributes() method on the Enable property. Confirmation of the change can be obtained
by enumerating the DCIM_iDRACCardService class (Section 19.6).

EXAMPLE:
winrm i ApplyAttributes http://schemas.dmtf.org/wbem/wscim/1/cim-
schema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_Computer
System +CreationClassName=DCIM_iDRACCardService
+SystemName=DCIM: ComputerSystem
+Name=DCIM: iDRACCardService
-u:USER -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8
-a:basic
-file: DracCard_PrivilegeChange.xml

The input file, DracCard_PrivilegeChange.xml, is shown below:

<p:ApplyAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRAC
CardService”>
    <p:Target>IDRAC.Embedded.1</p:Target>
    <p:AttributeName>Users.4#IpmiLanPrivilege</p:AttributeName>
    <p:AttributeValue>Operator</p:AttributeValue>
</p:ApplyAttributes_INPUT>
When this method is executed, a jobid or an error message is returned.

```
ApplyAttributes_OUTPUT
   Job
       Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
       ReferenceParameters
       ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_LifecycleJob
       SelectorSet
           Selector: InstanceID = JID_001296745342, __cimnamespace = root/dcim
       ReturnValue = 4096
```

## 5.4 Account Inventory (Using DMTF Model)

The list of user accounts may be retrieved by enumerating the CIM_Account class. The class provides the user account name and EnabledState properties. The user account password is also included but it is a write-only property.

**Profiles:**

- [http://www.dmtf.org/sites/default/files/standards/documents/DSP1039_1.0.0.pdf](http://www.dmtf.org/sites/default/files/standards/documents/DSP1039_1.0.0.pdf)

### 5.4.1 Account and Capabilities (Using DMTF Model)

Example-A demonstrates standard output. Example-B demonstrates EPR mode output.

**EXAMPLE-A:**

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_Account
-u: [USER] -p: [PASSWORD]
-r: https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT-A:**

```
DCIM_Account
   CreationClassName = DCIM_Account
   ElementName = DCIM Account
   EnabledDefault = 2
   EnabledState = 3
   Name = iDRAC.Embedded.1#Users.1
   OrganizationName = DCIM
   RequestedState = 0
   SystemCreationClassName = DCIM_SPComputerSystem
   SystemName = Systemmc TransitioningToState = 12
   UserID = null
   UserPassword = null
```
DCIM_Account
  CreationClassName = DCIM_Account
  ElementName = DCIM Account
  EnabledDefault = 2
  EnabledState = 2
  Name = iDRAC.Embedded.1#Users.2
  OrganizationName = DCIM
  RequestedState = 0
  SystemCreationClassName = DCIM_SPComputerSystem
  SystemName = Systemmc TransitioningToState = 12
  UserID = root
  UserPassword

EXAMPLE-B:
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_Account
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-returntype:EPR

OUTPUT-B:
EndpointReference
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
    ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Account
    SelectorSet
      Selector: __cimnamespace = root/dcim,
      Name = iDRAC.Embedded.1#Users.1,
      CreationClassName = DCIM_Account, SystemName = Systemmc,
      SystemCreationClassName = DCIM_SPComputerSystem

EndpointReference
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
    ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Account
    SelectorSet
      Selector: __cimnamespace = root/dcim,
      Name = iDRAC.Embedded.1#Users.2,
      CreationClassName = DCIM_Account, SystemName = Systemmc,
      SystemCreationClassName = DCIM_SPComputerSystem

Account setting capability is defined in the class CIM_AccountManagementCapabilities associated with the CIM_Account class instance. The ability to enable and disable an account is defined in the capability class CIM_EnabledLogicalElementCapabilities associated with the CIM_Account class.

To determine account setting capabilities:

1. Get the CIM_Account class instance of interest using EnumerateEPR mode.
2. Enumerate the associators of the CIM_Account instance and search for CIM_AccountManagementService class instance using EnumerateEPR mode.
3. Enumerate the associators of the CIM_AccountManagementService instance and search for CIM_AccountManagementCapabilities class instance.
4. One exception is account index 0. The first account is static and could not be set.

OUTPUT-C:

DCIM_LocalUserAccountManagementCapabilities
   - ElementName = Local User Account Management Capabilities
   - ElementNameEditSupported = false
   - InstanceID = DCIM:LocalUserAccountManagementCapabilities:1
   - MaxElementNameLen = 0
   - OperationsSupported = 3
   - SupportedAuthenticationMethod = 0, 1, 2

DCIM_IPMICLPAccountManagementCapabilities
   - ElementName = IPMI/CLP Account Management Capabilities
   - ElementNameEditSupported = false
   - InstanceID = DCIM:IPMICLPAccountManagementCapabilities:1
   - MaxElementNameLen = 0
   - OperationsSupported = 3

To determine account state setting capabilities:

1. Get the CIM_Account class instance of interest using EnumerateEPR mode.
2. Enumerate the associators of the CIM_Account instance and search for CIM_EnabledLogicalElementCapabilities class instance.
3. The presence of "RequestedStatesSupported" determines which states could be set.
4. One exception is account index 0. The first account is static and could not be set.

OUTPUT-D:

DCIM_EnabledLogicalElementCapabilities
   - ElementName = Account Capabilities
   - ElementNameEditSupported = false
   - InstanceID = DCIM_EnabledLogicalElementCapabilities:1
   - MaxElementNameLen = 0
   - RequestedStatesSupported = 2, 3

5.4.2 Privilege and Capabilities (Using DMTF Model)

The account privilege assigned to a user is defined in the class CIM_Privilege associated with the CIM_Account class. The class contains a list of privileges granted to the user account.

Profiles:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1034_1.0.1.pdf
http://www.dmtf.org/sites/default/files/standards/documents/DSP1039_1.0.0.pdf

To get the instance of CIM_Privilege for an account:

1. Get the CIM_Account class instance of interest using EnumerateEPR mode.
2. Enumerate the associators of the CIM_Account instance and search for CIM_Identity class instance using EnumerateEPR mode.
3. Enumerate the associators of the CIM_Identity instance and search for CIM_Role class instance using EnumerateEPR mode.
4. Enumerate the associators of the CIM_Role instance and search for CIM_Privilege class instance. An alternative to the above method, you can retrieve the specific CIM_Privilege instance by enumerating the class directly with filter. This method is similar to the example used to retrieve CIM_Account.
EXAMPLE:

winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/DCIM_LocalRolePrivilege
-u: [USER] -p: [PASSWORD]
-r: https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding: utf-8 -a:basic

OUTPUT:

DCIM_LocalRolePrivilege
  Activities = null
  ActivityQualifiers = null
  ElementName = DCIM Local Privilege 1
  InstanceID = DCIM:Privilege:1
  PrivilegeGranted = true
  QualifierFormats = null
  RepresentsAuthorizationRights = false

DCIM_LocalRolePrivilege
  Activities = 7, 7, 7, 7, 7, 7, 7, 7
  ActivityQualifiers = Login to DRAC, Configure DRAC, Configure Users, Clear Logs, Test Alerts, Execute Server Control Commands, Access Console Redirection, Access Virtual Media, Execute Diagnostic Commands
  ElementName = DCIM Local Privilege 2
  InstanceID = DCIM:Privilege:2
  PrivilegeGranted = true
  QualifierFormats = 9, 9, 9, 9, 9, 9, 9, 9
  RepresentsAuthorizationRights = true

DCIM_LocalRolePrivilege
  Activities = null
  ActivityQualifiers = null
  ElementName = DCIM Local Privilege 3
  InstanceID = DCIM:Privilege:3
  PrivilegeGranted = true
  QualifierFormats = null
  RepresentsAuthorizationRights = false

Privilege setting capability is defined in the class CIM_RoleBasedManagementCapabilities associated with the CIM_Privilege class instance. This class contains the list of possible values used to assign privileges.

Look for the property ActivityQualifiersSupported.

To determine privilege setting capabilities:

1. Acquire the class instance of CIM_Privilege of interest.
2. Enumerate the associators of the CIM_Privilege instance and search for CIM_RoleBasedAuthorizationService class instance using EnumerateEPR mode.
3. Enumerate the associators of the CIM_RoleBasedAuthorizationService instance and search for CIM_RoleBasedManagementCapabilities class instance using EnumerateEPR mode.
31

5.5 Manage Account Settings (Using DMTF Model)

5.5.1 Modify User Name (Using DMTF Model)
When the account setting capability allows, the user name of an account may be modified by issuing a set operation on the UserID property of the CIM_Account class instance. The set operation requires an instance reference. The instance reference may be retrieved by adding EnumerateEPR mode to enumerate or get of the class.

Profiles:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1034_1.0.1.pdf
http://www.dmtf.org/sites/default/files/standards/documents/DSP1039_1.0.0.pdf

The steps below demonstrate how to set the user name and password for local accounts.
A) Enumerate CIM_Account with EPR to identify all possible instance information to be used in a subsequent put or set operations.
EXAMPLE-A:

```bash
winrm e 
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/CIM_Account?__cimnamespace=root/dcim 
-u:[USER] -p:[PASSWORD] 
-r:https://[IPADDRESS]/wsman:443 
-SkipCNcheck -SkipCACheck 
-encoding:utf-8 -a:basic 
-returntype:EPR
```

When this command is executed, a list of objects will be returned. Below is a snippet of the output.

OUTPUT-A:

```
EndpointReference
   Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
   ReferenceParameters
   ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Account
   SelectorSet
   Selector: __cimnamespace = root/dcim,Name = iDRAC.Embedded.1#Users.1,CreationClassName = DCIM_Account,
             SystemName = Systemmc,SystemCreationClassName = DCIM_SPComputerSystem

EndpointReference
   Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
   ReferenceParameters
   ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Account
   SelectorSet
   Selector: __cimnamespace = root/dcim,Name = iDRAC.Embedded.1#Users.2,CreationClassName = DCIM_Account,
             SystemName = Systemmc,SystemCreationClassName = DCIM_SPComputerSystem
```

B) Perform a 'get' on any instance from A) to ensure correctness of the URI.

EXAMPLE-B:

```bash
winrm g
"http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Account?__cimnamespace=root/dcim
+CreationClassName=DCIM_Account
+Name= iDRAC.Embedded.1#Users.16
+SystemCreationClassName=DCIM_SPComputerSystem
+SystemName=Systemmc"
-r:https://[IPADDRESS]
-u:[USER] -p:[PASSWORD]
-a:basic
-encoding:utf-8
-SkipCACheck
-SkipCNCheck
```
When this method is executed, the particular object will be returned. Below is the output.

**OUTPUT-B:**

```
DCIM_Account
  CreationClassName = DCIM_Account
  ElementName = DCIM Account
  EnabledDefault = 2
  EnabledState = 3
  Name = iDRAC.Embedded.1#Users.16
  OrganizationName = DCIM
  RequestedState = 0
  SystemCreationClassName = DCIM_SPComputerSystem
  SystemName = Systemmc
  TransitioningToState = 12
  UserID = null
  UserPassword = null
```

**C)** If **B)** is successful, set the new values for the specified instance.

**EXAMPLE-C:**

```
winrmlink
 "http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_Account?__cimnamespace=root/dcim
+CcreationClassName= DCIM_Account
+CName= iDRAC.Embedded.1#Users.16
+CSystemCreationClassName=DCIM_SPComputerSystem
+CSystemName=Systemmc"
-r:https://[IPADDRESS]
-u:[USER] -p:[PASSWORD]
-a:basic -encoding:utf-8
 @{UserID="testuser4";UserPassword="testuser4"}
-SkipCACheck
-SkipCNCheck
-SkipRevocationCheck
```

When this command is executed, the UserID will be displayed in the output. The UserPassword will be displayed as null when the account is disabled. After the account is enabled, it will be displayed as blank. The value of UserPassword will never be displayed.

**OUTPUT-C:**

```
DCIM_Account
  CreationClassName = DCIM_Account
  ElementName = DCIM Account
  EnabledDefault = 2
  EnabledState = 3
  Name = iDRAC.Embedded.1#Users.16
  OrganizationName = DCIM
  RequestedState = 0
  SystemCreationClassName = DCIM_SPComputerSystem
  SystemName = Systemmc
  TransitioningToState = 12
  UserID = testuser4
  UserPassword = null
```
UserID = testuser4
UserPassword = testuser4

If the account specified is new or not yet enabled, it will not be accessible. Login as root in the UI and verify the user name is set correctly and enable it.

Logout of the UI. Logging in with new user name and password and be successful.

Possible responses:
1. A fault is returned which suggests a possible error in the request payload.
2. An empty response which suggests an error occurred while processing the request.
3. An instance of the class is returned where the property value is unchanged.
4. An instance of the class is returned where the property value is modified. The set is successful.
5. The property value may be blank as intended by the implementation for security. To determine success, try logging in with the new password. Make sure the account is enabled.

5.5.2 Modify Password (Using DMTF Model)
When the account setting capability allows, the user password of an account may be modified by issuing a set operation on the UserPassword property of the CIM_Account class instance. The set operation requires an instance reference. The instance reference may be retrieved by adding EnumerateEPR mode to enumerate or get of the class.

Note: The profile defines this property as string array of type octet string. In this implementation, the password is a string of type clear text. The security concern is resolved by transmission of this information only through secure HTTPS communication.

Profiles:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1034_1.0.1.pdf
http://www.dmtf.org/sites/default/files/standards/documents/DSP1039_1.0.0.pdf

See Section 5.5.1 for an implementation example.

5.5.3 Modify Account State (Using DMTF Model)
When the account setting capability allows, the user account may be enabled or disabled by invoking the RequestStateChange() method of the CIM_Account class instance. The invoke operation requires an instance reference. The instance reference may be retrieved by adding EnumerateEPR mode to enumerate or get of the class.

Profiles:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1034_1.0.1.pdf
http://www.dmtf.org/sites/default/files/standards/documents/DSP1039_1.0.0.pdf

Replace “DCIM User 16” with the applicable user name and “2” with the desired request state.

Invoke RequestStateChange() with the following parameters and syntax:

EXAMPLE:
winrm invoke RequestStateChange
"http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/DCIM_Account?__cimnamespace=root/dcim
+CreationClassName=DCIM_Account
+Name= iDRAC.Embedded.1#Users.16
+SystemCreationClassName=DCIM_SPComputerSystem
+SystemName=Systemmm"
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:u8 a:basic
@{RequestedState="2"} -skiprevocationcheck
OUTPUT:
RequestStateChange_OUTPUT
ReturnValue = 0

Response status other than zero indicates failure and error message information may be provided.

5.5.4 Modify User Privilege (Using DMTF Model)

When the account setting capability allows, the user account privileges may be modified by issuing a set() operation on the ActivityQualifiers property of the CIM_Privilege class instance associated with the CIM_Account class instance. The set() operation requires an instance reference. The instance reference may be retrieved by adding EnumerateEPR mode to enumerate or get of the class.

The profile defines this property as string array containing all the privileges to be granted for the account. Setting the list of privileges is a complete over-write of the previous setting. This restriction is a limitation where the protocol does not define how to set a particular index in the list. The new list will replace the previous list in its entirety.

Profiles:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1034_1.0.1.pdf
http://www.dmtf.org/sites/default/files/standards/documents/DSP1039_1.0.0.pdf

Here is an example list of available privileges from an instance of the class CIM_RoleBasedManagementCapabilities:

```
DCIM_LocalRoleBasedManagementCapabilities
ActivitiesSupported = 7, 7, 7, 7, 7, 7, 7, 7, 7
ActivityQualifiersSupported = Login to DRAC, Configure DRAC, Configure Users,
Clear Logs, Execute Server Control Commands, Access Console Redirection, Access
Virtual Media, Test Alerts, Execute Diagnostic Commands
ElementName = Local Role Based Management Capabilities
InstanceID = DCIM:LocalRoleBasedManagementCapabilities
QualifierFormatsSupported = 9, 9, 9, 9, 9, 9, 9, 9, 9
SharedPrivilegeSupported = false
SupportedMethods = 8
```

The privilege property ActivityQualifiers is an array of type string. To set more than one privilege, you need to provide the same key name more than once. The tool does not allow duplicate keys to be entered through the command line. Instead, you need to perform two operations.

- Get an instance of the CIM_Privilege class of interest.
- Using the class instance, replace the property ActivityQualifiers with the new values. Use the new instance XML as input to the set operation.

To determine if the new password has been successfully set, try logging in with the new password. Ensure the account is enabled.
6  Firmware Inventory

6.1  Software Inventory Profile Specification

The Dell Common Information Model (CIM) class extensions for supporting remote firmware inventory are defined in the Dell OS Software Update\(^2\) and related MOFs\(^3\). The diagrams representing the classes that are implemented by the Lifecycle Controller firmware can be found in Dell Software Inventory Profile.

6.2  Remote Inventory Method Invocation — Get Software Inventory

The SoftwareIdentity class contains information for the BIOS and component firmware installed on the target System as well as available firmware images cached in the Lifecycle Controller. The enumeration of the SoftwareIdentity class returns a list of SoftwareIdentity objects with properties such as firmware type and version.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

EXAMPLE:

winrm e cimv2/root/dcim/DCIM_SoftwareIdentity
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8

When this method is executed, a list of Software Identity objects will be returned, including installed and available firmware. Below is a snippet of the output.

OUTPUT:

```
DCIM_SoftwareIdentity
   BuildNumber = 4846
   Classifications = 10
   ComponentID = 28897
   ComponentType = APAC
   DeviceID = null
   ElementName = Dell Lifecycle Controller 2, 1.0.0.4846, X79
   FQDD = USC.Embedded.1:LC.Embedded.1
   IdentityInfoType = OrgID:ComponentType:ComponentID
   IdentityInfoValue = DCIM:firmware:28897
   InstallationDate = 2012-01-15T22:22:32Z
   InstanceID = DCIM:INSTALLED#802__USC.Embedded.1:LC.Embedded.1
   IsEntity = true
   MajorVersion = 1
   MinorVersion = 0
   RevisionNumber = 0
   RevisionString = null
   Status = Installed
   SubDeviceID = null
   SubVendorID = null
   Updateable = true
   VendorID = null
```
VersionString = 1.0.0.4846
impactsTPMmeasurements = false

The key properties in the above output include the following:

- **InstanceID**: Normally identifies the firmware on a particular type of device. The substring right after DCIM: is the status of a payload or firmware on the System. This can be installed or available.
- **ComponentID**: Uniquely identifies a unique type of device such as BIOS, NIC, Storage and Lifecycle controller firmware.
- **InstallationDate**: The date when the payload was installed to the System. If the System time was not set when the firmware installation took place the install date will be 1970-01-01. Factory installed firmware will have the 1970-01-01 date.
- **VersionString**: Shows the version of the firmware represented.
7 Firmware Update

7.1 Software Update Profile Specification

The Dell Common Information Model (CIM) class extensions for supporting BIOS, component firmware, and embedded Software update are defined in the Dell Software Update Profile 2 and related MOF files. The diagrams representing the classes that are implemented by the Lifecycle Controller firmware can be found in Dell Software Update Profile as well.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

7.2 "Rollback" Firmware

The InstallFromSoftwareIdentity() method is used for installation of a previous version of a component firmware that is available on the Lifecycle Controller (i.e. "rollback" of component firmware). The general "Rollback" firmware steps are performed in several stages as described in the next sections. Meanwhile, the steps are demonstrated in examples in Section 7.3 and Section 7.4.

7.2.1 Request “Rollback” Image

The first stage is a request to initiate and download the rollback image from the Lifecycle Controller by invoking the InstallFromSoftwareIdentity() method.

7.2.2 Create Reboot Job

The second stage is to create a reboot job as shown in Section 7.8.

7.2.3 Schedule Update Jobs

The third stage is to invoke the SetupJobQueue() method as shown in Section 10.2.1. Use the jobID(JID) from InstallFromSoftwareIdentity() and rebootID(RID) from the reboot job. The reboot may take several minutes as the UEFI performs the desired operation.

7.2.4 Monitor Update Jobs

The output of getting the job status during various steps, Section 10.2.3, is shown below. Initial job status after invoking InstallFromSoftwareIdentity

DCIM_LifecycleJob
InstanceID = JID_001276741956
JobStartTime = TIME_NA
JobStatus = Downloaded
JobUntilTime = TIME_NA
Message = Package successfully downloaded.
MessageArguments = null
MessageID = RED002
Name = Rollback:DCIM:AVAILABLE:NONPCI:159:2.1.4

Job status after invoking SetupJobQueue

DCIM_LifecycleJob
InstanceID = JID_001276741956
JobStartTime = 00000101000000 JobStatus = Scheduled
JobUntilTime = 20100730121500
Message = Task successfully scheduled
MessageArguments = null
MessageID = JCP001
Name = Rollback:DCIM:AVAILABLE:NONPCI:159:2.1.4

Job status following reboot / install of operation

DCIM_LifecycleJob
InstanceID = JID_001276741956
JobStartTime = 00000101000000
JobStatus = Completed
JobUntilTime = 20100730121500
Message = Job finished successfully
MessageArguments = null
MessageID = USC1
Name = Rollback:DCIM:AVAILABLE:NONPCI:159:2.1.4

7.3 BIOS Firmware “Rollback”

The InstallFromSoftwareIdentity() method is used for installation of a previous version of a component firmware that is available on the Lifecycle Controller (i.e. “rollback” of component firmware).

All steps to complete a rollback successfully are listed below.

Invoke InstallFromSoftwareIdentity() with the following parameters and syntax:

[InstanceID]: This is the instanceID of the SoftwareIdentity that is to be used to rollback the firmware to a previous version. The InstanceID can have value such as:
DCIM:AVAILABLE:NONPCI:159:2.1.4
- It is available firmware on a NONPCI device.
- This refers BIOS version 2.1.4

EXAMPLE:
winrm i InstallFromSoftwareIdentity
cimv2/root/dcim/DCIM_SoftwareInstallationService?CreationClassName=DCIM_SoftwareInstallationService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=IDRAC:ID
+Name=SoftwareUpdate -file:RollInputBIOS.xml
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8

The rollback input file, RollInputBIOS.xml, is shown below:

<p:InstallFromSoftwareIdentity_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_SoftwareInstallationService">
xmlns:w="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd">
<a:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing</a:Address>
</p:Target>
<a:ReferenceParameters>
7.4 NIC Firmware “Rollback”

The InstallFromSoftwareIdentity() method is used for installation of a previous version of a component firmware that is available on the Lifecycle Controller (i.e. “rollback” of component firmware).

Invoke InstallFromSoftwareIdentity with the following parameters and syntax:

[InstanceID]: This is the instanceID of the SoftwareIdentity that is to be used to rollback the firmware to a previous version. The InstanceID can have value such as:


- It refers to a previous firmware on a PCI device.
- ID (Vendor ID) = 14E4
- DID (Device ID) = 1639
- SSID (SubSystem ID) = 0237
- SVID (Subvendor ID) = 1028
- This refers to a Broadcom NetXtreme II BCM5709 network adaptor 7.

EXAMPLE:

```
winrm i InstallFromSoftwareIdentity
cimv2/root/dcim/DCIM_SoftwareInstallationService?CreationClassName=DCIM_SoftwareInstallationService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=IDRAC:ID
+Name=SoftwareUpdate -file:RollInputNIC.xml
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
```

The rollback input file, RollInputNIC.xml, is shown below:

```xml
<installFromSoftwareIdentity_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_SoftwareInstallationService">
    <selectorSet>
        <selector Name="InstanceID">[InstanceID]</selector>
    </selectorSet>
</installFromSoftwareIdentity_INPUT>
```

When this method is executed, a jobid or an error message is returned.

```
InstallFromSoftwareIdentity_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    <ReferenceParameters>
        <ResourceURI>
            http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_SoftUpdateConcreteJob
        </ResourceURI>
        <SelectorSet>
            <Selector> InstanceID = JID_001276741956, __cimnamespace = root/dcim
        </SelectorSet>
        ReturnValue = null
```

```xml
The rollback input file, RollInputNIC.xml, is shown below:

```xml
<installFromSoftwareIdentity_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_SoftwareInstallationService">
    <selectorSet>
        <selector Name="InstanceID">[InstanceID]</selector>
    </selectorSet>
</installFromSoftwareIdentity_INPUT>
```
  <a:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</a:Address>
  <a:ReferenceParameters>
    <w:SelectorSet>
      <w:Selector Name="InstanceID">[InstanceID]</w:Selector>
    </w:SelectorSet>
  </a:ReferenceParameters>
</p:Target>
</p:InstallFromSoftwareIdentity_INPUT>

**OUTPUT:**
When this method is executed, a jobid or an error message is returned.

**InstallFromSoftwareIdentity_OUTPUT**

Job
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
    ResourceURI = 
      http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_SoftUpdateConcreteJob
    SelectorSet
      Selector: InstanceID = JID_001265811668, __cimnamespace = root/dcim
  ReturnValue = null

Entering an invalid instanceID may yield the following error message:

**InstallFromSoftwareIdentity_OUTPUT**

  Message = Invalid InstanceID
  MessageID = SUP024
  ReturnValue = null

7.5 **Update from Network Source**

A Firmware update can be achieved by invoking the InstallFromURI() method in the class DCIM_SoftwareInstallationService. Firmware update is performed in several stages as described in the next sections. The steps are demonstrated in examples in Section 7.6 and Section 7.7.

Note: When using WSMAN command to initiate update jobs, make sure to wait for two seconds before submitting a second job in order to avoiding racing conditions.

7.5.1 **Request Update Download**
The first stage is a request to initiate and download the update image from a source defined by the user by invoking the InstallFromURI() method.

7.5.2 **Monitor Download Status**
Downloading the update package may take several minutes. The second stage is to monitor the download. The download status may be monitored by enumerating or getting the instance of the corresponding job.
7.5.3 **Reboot to Perform Update**
Once downloaded, the request needs to be scheduled. The third stage is to schedule the update. To schedule the update, use the SetupJobQueue() method of the class DCIM_JobService in Section 10.2.1.

7.5.4 **Wait for Job Completion**
The fourth stage is to wait for the job to be completed, which may take several minutes. The job status can be monitored as shown in Section 10.2.3.

7.5.5 **Delete Job**
The fifth and final stage is to delete the completed job from the job store. Deleting the job queue is shown in Section 10.2.2.

7.6 **Update NICs from HTTP, CIFS Share, NFS Share, TFTP, or FTP**
The InstallFromURI() method takes the following input and downloads the Dell Update Package to the Lifecycle Controller in the target System. The method returns a jobid for an instance of DCIM_SoftwareUpdateJob that can be scheduled to execute or queried for status at a later time. The following is the example of the method for updating a NIC firmware.

Invoke InstallFromURI() with the following parameters and syntax:

1. **[URI-IP-ADDRESS]**: This is the IP address of the location for Dell Update Package. The Dell Update Package will need to be the Windows type update package. The file share can be HTTP, CIFS, NFS, TFTP, or FTP type as shown below:
   - **HTTP Format:**
     `http://[IP ADDRESS]/[PATH TO FILE.exe]`
   - **CIFS Format:**
     `cifs://WORKGROUP_NAME\[USERNAME]@[PASSWORD]@[URI-IP-ADDRESS]/[FILE.exe];mountpoint=[DIRECTORYNAME]`
   - **TFTP or FTP Format:**
     `tftp://[IP ADDRESS]/[PATH TO FILE.exe]  ftp://[IP ADDRESS]/[PATH TO FILE.exe]`

2. **[InstanceId]**: The instanceID is the SoftwareIdentify instanceID that represents the firmware that is to be updated. This instanceID can be retrieved as described in Section 6.2. For example, the instanceID can be:

   - It is installed firmware on a PCI device.
     - **VID (Vendor ID) = 14E4**
     - **DID (Device ID) = 1636**
     - **SSID (SubSystem ID) = 0237**
     - **SVID (Subvendor ID) = 1028**
     - This refers to a Broadcom NetXtreme II BCM5709 network adaptor 7.

**EXAMPLE:**

```
winrm invoke InstallFromURI
cimv2/root/dcim/DCIM_SoftwareInstallationService?CreationClassName=DCIM_SoftwareInstallationService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=IDRAC:ID
+Name=SoftwareUpdate
-file:UpdateInputNIC.xml
-u:[UserName] -p:[Password]
-r:https://[IPADDRESS]/wsman:443
```
The above command takes in an input file named UpdateInputNic.xml to supply input parameters required for the InstallFromURI() method.

The syntax for UpdateInputNic.xml is:

```xml
<p:InstallFromURI_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_SoftwareInstallationService">
  <p:URI>http://[URI-IP-ADDRESS]/[PATH-TO-EXE]/[FILE.exe]</p:URI>
    <a:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</a:Address>
    <a:ReferenceParameters>
      <w:SelectorSet>
        <w:Selector Name="InstanceID">[INSTANCEID]</w:Selector>
      </w:SelectorSet>
    </a:ReferenceParameters>
  </p:Target>
</p:InstallFromURI_INPUT>
```

In the above sample, the [URI-IP-ADDRESS] must be replaced with the actual value of the IP address of the server that stores update content, [PATH-TO-EXE] must be replaced with the applicable path to the executable, [FILE.exe] must be replaced with the executable name, and [INSTANCEID] should be replaced with the actual InstanceID of the device to be updated.

**OUTPUT:**

When this method is executed, a jobid or an error message is returned. This jobid can then be used for subsequent processing with job control provider in Section 10.

**InstallFromURI_OUTPUT**

Job

Address = http://schemas.xmlsoap.org/ws

ReferenceParameters

  ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_SoftUpdateConcreteJob
  SelectorSet
    Selector: InstanceID = JID_001265810325, __cimnamespace = root/dcim

ReturnValue = null

Missing XML parameters may yield the following error message:

**InstallFromURI_OUTPUT**

Message = Insufficient Method Parameters

MessageID = SUP001

ReturnValue = null
7.7 Update BIOS from HTTP, CIFS Share, NFS Share, TFTP, or FTP

The InstallFromURI() method takes the following input and downloads the Dell Update Package to the Lifecycle Controller in the target System. The method returns a jobid for an instance of DCIM_SoftwareUpdateJob that can be scheduled to execute or queried for status at a later time. The following is an example of the method for updating a BIOS firmware.

Invoke InstallFromURI() with the following parameters and syntax:

[URI-IP-ADDRESS]: This is the IP address of the location for Dell Update Package. The Dell Update Package will need to be the Windows type update package. The file share can be HTTP, CIFS, NFS, TFTP, or FTP type as shown below:

**HTTP Format:**

```
http://[IP ADDRESS]/[PATH TO FILE.exe]
```

**CIFS Format:**

```
cifs://[USERNAME]:[PASSWORD]@[URI-IP-ADDRESS]/ [PATH TO FILE.exe]; mountpoint=/ [DIRECTORYNAME]
```

**TFTP or FTP Format:**

```
tftp://[IP ADDRESS]/[PATH TO FILE.exe] ftp://[IP ADDRESS]/[PATH TO FILE.exe]
```

**[InstanceId]:** The instanceID is the SoftwareIdentify instanceID that represents the firmware that is to be updated. This instanceID can be retrieved as described in Section 6.2. For example, the instanceID can be:

```
DCIM:AVAILABLE:NONPCI:159:2.1.4
```

- It is available firmware on a NONPCI device.
- This refers BIOS version 2.1.4

**EXAMPLE:**

```
winrm invoke InstallFromURI
cimv2/root/dcim/DCIM_SoftwareInstallationService?CreationClassName=DCIM_SoftwareInstallationService
+SystemCreationClassName=DCIM_ComputerSystem
+Name=SoftwareUpdate -file:UpdateInputBIOS.xml
-username: [UserName] -password: [Password]
-ignoreCNCheck
-auth:basic
-encoding:utf-8
```

The above command takes in an input file named UpdateInputBIOS.xml to supply input parameters required for the InstallFromURI() method.

The syntax for UpdateInputBIOS.xml is:

```
<p:InstallFromURI_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_SoftwareInstallationService">
  <p:URI>http://[URI-IP-ADDRESS]/[PATH-TO-EXE]/[FILE.exe]</p:URI>
    <a:Address>http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</a:Address>
    <a:ReferenceParameters>
    </a:ReferenceParameters>
</p:Target>
```

Dell Lifecycle Controller Web Services Interface Guide for Windows
In the above sample, the [URI-IP-ADDRESS] must be replaced with the actual value of the IP address of the server that stores update content, [PATH-TO-EXE] must be replaced with the applicable path to the executable, [FILE.exe] must be replaced with the executable name, and [INSTANCEID] should be replaced with the actual InstanceID of the device to be updated.

**OUTPUT:**

When this method is executed, a jobid or an error message is returned. This jobid can then be used for subsequent processing with job control provider in section 10.

**InstallFromURI_OUTPUT**

Job Address = http://schemas.xmlsoap.org/ws

ReferenceParameters

ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_SoftUpdateConcreteJob

SelectorSet

Selector: InstanceID = JID_001276741475, __cimnamespace = root/dcim

ReturnValue = null

### 7.8 CreateRebootJob()

The CreateRebootJob() method creates a reboot job that can be scheduled to reboot immediately or at a later time. When the reboot job is scheduled and then executed, via SetupJobQueue() (Section 10.2.1), the reboot will take several minutes depending on the System setup, including whether collecting System inventory (CSIOR) is enabled.

Invoke CreateRebootJob with the following parameters and syntax:

- **RebootJobType:** There are three options for rebooting the System.
  1 = PowerCycle
  2 = Graceful Reboot without forced shutdown
  3 = Graceful reboot with forced shutdown

**EXAMPLE:**

```
winrm invoke CreateRebootJob
//cimv2/root/dcim/DCIM_SoftwareInstallationService?CreationClassName=DCIM_SoftwareInstallationService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=IDRAC:ID
+Name=SoftwareUpdate -file:reboot.xml
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNCheck
-auth:basic
-encoding:utf-8
```
The syntax for reboot.xml is:

```xml
<p:CreateRebootJob_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_SoftwareInstallationService">
  <p:RebootJobType>2</p:RebootJobType>
</p:CreateRebootJob_INPUT>
```

**OUTPUT:**

This method will return a reboot jobid that can be set to reboot the System immediately or at a later time.

```xml
CreateRebootJob_OUTPUT
RebootJobID
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
  ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_SoftUpdateConcreteJob
  SelectorSet
    Selector: InstanceID = RID_001265648530, __cimnamespace = root/dcim
ReturnValue = null
```

The jobid in the above output is the instanceID:

```
Jobid = InstanceID = RID_001265648530
```

### 7.9 Automatic Updates

Automatic Updates feature allows for periodic firmware updates at regular intervals as configured by the user.

#### 7.9.1 Enable Automatic Update

This method enables/disables the “Automatic Update Feature” attribute.

**EXAMPLE:**

```bash
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://<<ip>>/wsman
-SkipCNcheck -SkipCACheck encoding:utf-8
-a:basic @{AttributeName="Automatic Update Feature";AttributeValue="Enabled"}
```

**OUTPUT:**

```
Message = The command was successful
MessageID = LC001
RebootRequired = No
ReturnValue = 0
SetResult = Set PendingValue
```
7.9.2 Create a Config Job
CreateConfigJob sets the pending value set by SetAttribute() method.

EXAMPLE:
```
winrm i CreateConfigJob
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://<<ip>>/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:
CreateConfigJob_OUTPUT
```
Job
EndpointReference
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet
        Selector: InstanceID = JID_859945354433, cimnamespace = root/dcim

ReturnValue = 4096
```

Verify the value of “Automatic Update Feature” attribute from DCIM_iDRACCardEnumeration. It has to be “Enabled” to set Automatic update schedule.

7.9.3 Set Update Schedule
SetUpdateSchedule() method sets the schedule for the automatic updates and the source repository from where the updates are to be applied from.

EXAMPLE:
```
winrm i SetUpdateSchedule
cimv2/root/dcim/DCIM_SoftwareInstallationService?SystemCreationClassName=DCIM_SoftwareInstallationService
+SystemName=IDRAC:ID
+CreationClassName=DCIM_SoftwareInstallationService
+Name=SoftwareUpdate -u:root -p:calvin
-r:https://<<ip>>/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8
-a:basic -file:SetSchedule.xml
```

SetSchedule.xml
```
    <p:IPAddress>IPADDR</p:IPAddress>
    <p:ShareName>Sharename/Repository</p:SharePath>
    <p:ShareType>0</p:ShareType>
```
7.9.4 **Get the Update Schedule**

GetUpdateSchedule() lists the parameter set by SetUpdateSchedule()

**EXAMPLE:**

```bash
winrm i GetUpdateSchedule
cimv2/root/dcim/DCIM_SoftwareInstallationService?SystemCreationClassName=DCIM_SoftwareInstallationService
+SystemName=IDRAC:ID
+CreationClassName=DCIM_SoftwareInstallationService
+Name=SoftwareUpdate
-u:root -p:calvin
-r:https://<<ip>>/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```bash
GetUpdateSchedule_OUTPUT
  ApplyReboot = 0
  CatalogName = Sample.xml
  DayofMonth = *
  IPAddress = 10.94.192.100
  Repeat = 5
  ReturnValue = 4096
  ShareName = Somepath
  ShareType = nfs
  Time = 16:00
```

7.9.5 **Clear the Update Schedule**

Clears the schedule for the automatic updates, that has been set by SetUpdateSchedule() method.

**EXAMPLE:**

```bash
winrm i ClearUpdateSchedule
cimv2/root/dcim/DCIM_SoftwareInstallationService?SystemCreationClassName=DCIM_SoftwareInstallationService
+SystemName=IDRAC:ID
+CreationClassName=DCIM_SoftwareInstallationService
+Name=SoftwareUpdate
-u:root -p:calvin
-r:https://[IPADDR]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```
7.10 Device Update from Repository

This feature allows multiple firmware updates by specifying a network repository which contains a catalog of available updates. All applicable updates contained in the repository are applied to the System.

Following two methods introduced as a part of DCIM_SoftwareUpdate profile:

- InstallFromRepository: Initiate a job for device updates and creates a comparison report.
- GetRepoBasedUpdateList: Get the comparison report generated with InstallFromRepository

7.10.1 Install from Repository

The InstallFromRepository method applies the updates.

Options available for user to update devices from repositories using WS-Man.

<table>
<thead>
<tr>
<th>ApplyUpdate</th>
<th>ReboootNeeded</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>Only comparison report is generated</td>
</tr>
<tr>
<td>1</td>
<td>TRUE</td>
<td>All updates are applied immediately. Note: If the update requires a restart, the System is automatically restarted immediately.</td>
</tr>
<tr>
<td>1</td>
<td>FALSE</td>
<td>Currently, only updates that do not require a System restart will be applied. For example, Driver Pack DUPs. For those that require a restart, a separate reboot job will have to be created by the user with CreateRebootJob. These updates will go to a scheduled state and are run after a restart.</td>
</tr>
</tbody>
</table>

EXAMPLE:

```
winrm i InstallFromRepository
cimv2/root/dcim/DCIM_SoftwareInstallationService?CreationClassName=DCIM_SoftwareInstallationService+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=IDRAC:ID
+Name=SoftwareUpdate
-u:root -p:calvin
-r:https://[iDRAc ip]/wsman:
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic @{ipAddress="[Share ip address] ";ShareName="[] ";ShareType="[] ";UserName="[] ";Password="
";RebootNeeded="TRUE";CatalogFile="Catalog.xml";ApplyUpdate="1"}
```

OUTPUT:

```
InstallFromRepository_OUTPUT
Job
  EndpointReference Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
    ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_LifecycleJob
    SelectorSet
      Selector: InstanceID = JID_776094296053, __cimnamespace = root/dcim
  ReturnValue = 4096
```
7.10.2 Get Repo-Based Update List

A comparison XML between the inventory present on the System and the updates present on the repository can be obtained using the GetRepoBasedUpdateList() method.

**EXAMPLE:**

```bash
winrm i GetRepoBasedUpdateList
```

**OUTPUT:**

```xml
GetRepoBasedUpdateList_OUTPUT
PackageList = <?xml version="1.0"?>
<CIM xmlns:fo="http://www.w3.org/1999/XSL/Format" CIMVERSION="2.0" DTDVERSION="2.0">
  <MESSAGE ID="4711" PROTOCOLVERSION="1.0">
    <SIMPLEREQ>
      <VALUE.NAMEDINSTANCE>
        <INSTANCENAME CLASSNAME="DCIM_RepoUpdateSWID">
          <PROPERTY NAME="Criticality" TYPE="string">
            <VALUE>1</VALUE>
          </PROPERTY>
        </VALUE.NAMEDINSTANCE>
        <PROPERTY NAME="DisplayName" TYPE="string">
          <VALUE>Intel(R) Ethernet 10G 4P X540/I350 rNDC - BC:30:5B:ED:50:38</VALUE>
        </PROPERTY>
        <PROPERTY NAME="BaseLocation" TYPE="string">
          <VALUE/>
        </PROPERTY>
        <PROPERTY NAME="PackagePath" TYPE="string">
          <VALUE>Network_Firmware_KTT4W_WN64_14.5.5_X03.EXE</VALUE>
        </PROPERTY>
        <PROPERTY NAME="PackageName" TYPE="string">
          <VALUE>Network_Firmware_KTT4W_WN64_14.5.5_X03.EXE</VALUE>
        </PROPERTY>
        <PROPERTY NAME="PackageVersion" TYPE="string">
          <VALUE>14.5.5</VALUE>
        </PROPERTY>
        <PROPERTY NAME="RebootType" TYPE="string">
          <VALUE>HOST</VALUE>
        </PROPERTY>
        <PROPERTY NAME="JobID" TYPE="string">
          <VALUE/>
        </PROPERTY>
      </VALUE.NAMEDINSTANCE>
    </SIMPLEREQ>
  </MESSAGE>
</CIM>
```
8  Power State Management

8.1 Description of Base Server Versus Power State Management Methods

The remote control of a server power state (On, Off) and methodology for cycling power is available through data models specified in both the DMTF Base Server Profile and the DMTF Power State Management Profile. The Base Server Profile offers the RequestStateChange() method on the instance of the CIM_ComputerSystem class representing the server platform. The Power State Management Profile offers the RequestPowerStateChange() method available on the instance of the PowerStateManagementService associated with the instance of CIM_ComputerSystem representing the server platform. Base Server Profile:

http://www.dmtf.org/sites/default/files/standards/documents/DSP1004_1.0.1.pdf

Power State Management Profile:

http://www.dmtf.org/sites/default/files/standards/documents/DSP1027_2.0.0.pdf

8.2 Get Power State

8.2.1 Base Server Method

The power state of the System is reported by the EnabledState property of the DCIM_ComputerSystem class.

EXAMPLE:


OUTPUT:

DCIM_ComputerSystem

  CreationClassName = DCIM_ComputerSystem
  Dedicated = 0
  ElementName
  EnabledState = 2
  HealthState = 25
  IdentifyingDescriptions = CIM:GUID, CIM:Tag, DCIM:ServiceTag
  Name = srv:System
  OperationalStatus = 6
  OtherIdentifyingInfo = 4c4c4544-0036-3510-8034-b7c04f333231,
  mainSystemchassis,
  7654321
  PrimaryStatus = 3
  RequestedState = 0
8.2.2 Power State Management Method

The power state of the System is also reported by the PowerState property of the DCIM_CSAssociatedPowerManagementService class.

Power State Management Profile:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1027_2.0.0.pdf

EXAMPLE:

winrm e
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

PowerState:
2 (On): System is fully on
13 (Off): System is powered off

DCIM_CSAssociatedPowerManagementService
PowerOnTime = null
PowerState = 2
RequestedPowerState = 0
ServiceProvided

EndpointReference
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
  SelectorSet
  Selector: SystemCreationClassName = DCIM_SPComputerSystem,
  CreationClassName = DCIM_CSPowerManagementService, SystemName = Systemmc, Name = pwrmgtsvc:1, __cimnamespace = root/dcim
UserOfService

EndpointReference
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
  ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_ComputerSystem
  SelectorSet
  Selector: Name = srv:System,
  CreationClassName = DCIM_ComputerSystem, __cimnamespace = root/dcim

8.3 Get Power Control Capabilites

8.3.1 Base Server Method

The power control capabilities are reported by the RequestedStatesSupported property of the CIM_EnabledLogicalElementCapabilities class associated with the main System CIM_ComputerSystem class.
In “Part A” enumerate the CIM_ElementCapabilities class and search for the DCIM_CSElementCapabilities reference. Use the resulting InstanceID in “Part B” to obtain the RequestedStatesSupported property.

EXAMPLE (Part A):

```plaintext
```

OUTPUT (Part A):

```
DCIM_CSElementCapabilities
Capabilities
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
    ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_CSEnabledLogicalElementCapabilities
SelectorSet
    Selector: InstanceID = DCIM:ComputerCap:1, __cimnamespace = root/dcim
    Characteristics = null
ManagedElement
    Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
    ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_ComputerSystem
SelectorSet
    Selector: Name = srv:System, CreationClassName =DCIM_ComputerSystem, __cimnamespace = root/dcim
```

EXAMPLE (Part B):

```plaintext
```

OUTPUT (Part B):

```
RequestedStatesSupported:
   A: Enabled
   B: Disabled
   11: Reset
```
8.3.2 Power State Management Method

The power control capabilities are also reported by the PowerStatesSupported property of the CIM_PowerManagementCapabilities (PMC) class associated with the CIM_PowerManagementService (PMS) class. Getting the instance of PMC is a two step process. First, enumerate the instance of PMS with EPR. Second, enumerate the associated PMC class. When there is only one instance of PMC class as in the case of iDRAC, the first step may be skipped and the PMC class may be enumerated directly.

Power State Management Profile:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1027_2.0.0.pdf

EXAMPLE (iDRAC case):

```
```

OUTPUT:

When the PowerStatesSupported property contains the value in the “PowerStatesSupported Value” column, the PowerChangeCapabilities property shall contain the value specified in the “PowerChangeCapabilities Value” column.

<table>
<thead>
<tr>
<th>PowerStatesSupported Value</th>
<th>PowerChangeCapabilities Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Power On)</td>
<td></td>
</tr>
<tr>
<td>(Sleep - Light)</td>
<td>(Power State Settable)</td>
</tr>
<tr>
<td>(Sleep - Deep)</td>
<td>(Power Cycling Supported)</td>
</tr>
<tr>
<td>(Power Cycle (Off Soft))</td>
<td></td>
</tr>
<tr>
<td>(Power Off - Hard)</td>
<td></td>
</tr>
<tr>
<td>(Hibernate)</td>
<td></td>
</tr>
<tr>
<td>(Power Off - Soft)</td>
<td>(Off Hard Power Cycling Supported)</td>
</tr>
<tr>
<td>(Power Cycle (Off Hard))</td>
<td>(HW Reset Supported)</td>
</tr>
<tr>
<td>(Master Bus Reset)</td>
<td></td>
</tr>
<tr>
<td>(Diagnostic Interrupt (NMI))</td>
<td>(HW Reset Supported)</td>
</tr>
<tr>
<td>(Power Off - Soft Graceful)</td>
<td>(Graceful Shutdown Supported)</td>
</tr>
<tr>
<td>(Power Off - Hard Graceful)</td>
<td></td>
</tr>
<tr>
<td>(Master Bus Reset Graceful)</td>
<td>(HW Reset Supported) and (Graceful Shutdown Supported)</td>
</tr>
<tr>
<td>(Power Cycle (Off - Soft Graceful))</td>
<td>(Power Cycling Supported) and (Graceful Shutdown Supported)</td>
</tr>
<tr>
<td>(Power Cycle (Off - Hard Graceful))</td>
<td>(Off Hard Power Cycling Supported) and (Graceful Shutdown Supported)</td>
</tr>
</tbody>
</table>
DCIM_CSPowerManagementCapabilities
Caption = null
Description = null
ElementName = Power Management Capabilities
InstanceID = DCIM:pwrmgtcap1
OtherPowerCapabilitiesDescriptions = null
OtherPowerChangeCapabilities = null
PowerCapabilities = null
PowerChangeCapabilities = 3, 4, 8
PowerStatesSupported = 2, 5, 8, 11, 12

8.4 Power Control

8.4.1 Base Server Method
Changing the power state, such as cycling the power, is performed by invoking the RequestStateChange() method of the CIM_ComputerSystem class instance. For iDRAC, there is one instance for the main System and another for iDRAC. Use the main System instance. The method requires you to specify the RequestedState argument. Refer to Section 8.3 to get the possible values for this argument.

Base Server Profile:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1004_1.0.1.pdf

EXAMPLE:
winrm invoke RequestStateChange
"http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_ComputerSystem?CreationClassName=DCIM_ComputerSystem
+Name=srv:System"
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic @{RequestedState="2"}
-output

OUTPUT:
RequestStateChange_OUTPUT
ReturnValue = 0

Return values of zero indicate success, while others indicate failure and may include a corresponding error message.

8.4.2 Power State Management Method
Changing the power state is performed by invoking the RequestPowerStateChange() method of the DCIM_PowerManagementService (PMS) class instance. It is a three step process shown below:
1. Enumerate the DCIM_PowerManagementService with EPR
2. Enumerate the DCIM_ComputerSystem class and search for the Host instance
3. Use the EPR on steps 1) and 2) to invoke RequestPowerStateChange()

Power State Management Profile:
http://www.dmtf.org/sites/default/files/standards/documents/DSP1027_2.0.0.pdf
EXAMPLE:

`winrm invoke RequestPowerStateChange http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_CSPowerManagementService__cimnamespace=root/dcim+SystemCreationClassName=DCIM_SPComputerSystem
+SystemName=Systemmc
+CreationClassName=DCIM_CSPowerManagementService
+Name=pwrmgtsvc:1
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8
-auth:basic @(PowerState="5")`
9 Hardware Inventory

The Dell Common Information Model (CIM) class extensions for supporting remote hardware inventories are defined in the various Dell profiles and related MOFs. The Hardware Inventory allows users to remote query the inventory of hardware.

Each of the hardware inventory classes return the attribute LastSystemInventoryTime, which is when the last time 'collect System inventory on restart' or CSIOR was run. See Section 12.1 for more details on CSIOR. It is an important attribute as it shows how recently the inventory was updated.

9.1 Power Supply Inventory

This section describes the implementation for the DCIM_PowerSupplyView class. The Dell Power Supply Profile describes platform’s power supply information. Each platform power supply is represented by an instance of DCIM_PowerSupplyView class.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate DCIM_PowerSupplyView with the following parameters and syntax:

EXAMPLE:

```
winrm e cimv2/root/dcim/DCIM_PowerSupplyView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
-SkipCACheck
-SkipCNCheck
```

OUTPUT:

```
DCIM_PowerSupplyView
   DetailedState = Presence Detected
   FQDD = PSU.Slot.1
   FirmwareVersion = 00.01.31
   InputVoltage = 120
   InstanceID = PSU.Slot.1
   LastSystemInventoryTime = 20100331101859
   LastUpdateTime = 20100401130928
   Manufacturer = Dell
   Model = PWR SPLY,502W,RDNT
   PartNumber = 0MU791A00
   PrimaryStatus = 1
   RedundancyStatus = 2
   SerialNumber = CN732459700411
   TotalOutputPower = 502
   Type = 0
```

```
DCIM_PowerSupplyView
   DetailedState = Presence Detected
   FQDD = PSU.Slot.2
   FirmwareVersion = 00.01.31
   InputVoltage = 118
   InstanceID = PSU.Slot.2
```

9.2 Fan Inventory

This section describes the requirements and guidelines for implementing Dell Fan Profile. The Dell Fan Profile describes platform’s fans including the fan speed sensor information. Each platform fan is represented by an instance of DCIM_FanView class.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate DCIM_FanView with the following parameters and syntax:

EXAMPLE:

```bash
```

OUTPUT:

```plaintext
DCIM_FanView
  ActiveCooling = true
  BaseUnits = 19
  CurrentReading = 4200
  FQDD = Fan.Embedded.1A
  InstanceID = Fan.Embedded.1A
  LastSystemInventoryTime = 20100331101859
  LastUpdateTime = 20100408115623
  PrimaryStatus = 1
  RateUnits = 4
  RedundancyStatus = 2
  UnitModifier = 0
  VariableSpeed = true
```

DCIM_FanView
  ActiveCooling = true
  BaseUnits = 19
  CurrentReading = 4440
  FQDD = Fan.Embedded.2A
  InstanceID = Fan.Embedded.2A
9.3 Memory Inventory

This section describes the implementation for the DCIM_MemoryView class. The Dell Memory Profile describes platform’s physical memory. Each DIMM’s information is represented by an instance of DCIM_MemoryView class.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate DCIM_MemoryView with the following parameters and syntax:

EXAMPLE:

```
winrm e cimv2/root/dcim/DCIM_MemoryView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
-SkipCACHheck

OUTPUT:
DCIM_MemoryView
  BankLabel = B
  CurrentOperatingSpeed = 1067
  FQDD = DIMM.Socket.B1
  InstanceID = DIMM.Socket.B1
  LastSystemInventoryTime = 20100331101859
  LastUpdateTime = 20100325134947
  ManufactureDate = Mon Jun 29 12:00:00 2009 UTC
  Manufacturer = Samsung
  MemoryType = 24
  Model = DDR3 DIMM
  PartNumber = M391B2873DZ1-CH9
  PrimaryStatus = 1
  Rank = 1
  SerialNumber = 85C6DF30
  Size = 1024
  Speed = 1333

DCIM_MemoryView
  BankLabel = A
  CurrentOperatingSpeed = 1067
  FQDD = DIMM.Socket.A3
```
9.4 CPU Inventory

This section describes the implementation for the DCIM_CPUView class. The Dell CPU Profile describes platform’s CPUs. Each CPU’s information is represented by an instance of DCIM_CPUView class.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate DCIM_CPUView with the following parameters and syntax:

EXAMPLE:

winrm e cimv2/root/dcim/DCIM_CPUView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
-SkipCACheck
-SkipCNCheck

OUTPUT:

DCIM_CPUView

CPUFamily = B3
CPUStatus = 1
Cache1Associativity = 7
Cache1ErrorMethodology = 5
Cache1Level = 0
Cache1PrimaryStatus = 1
Cache1SRAMType = 2
Cache1Size = 256
Cache1Type = 4
Cache1WritePolicy = 0
Cache2Associativity = 7
Cache2ErrorMethodology = 5
Cache2Level = 1
Cache2PrimaryStatus = 1
Cache2SRAMType = 2
Cache2Size = 2048
Cache2Type = 5
Cache2WritePolicy = 0
Cache3Associativity = 14
Cache3ErrorMethodology = 5
Cache3Level = 2
Cache3PrimaryStatus = 1
Cache3SRAMType = 2
Cache3Size = 20480
Cache3Type = 5
Cache3WritePolicy = 0
Characteristics = 4
CurrentClockSpeed = 2266
ExternalBusClockSpeed = 5860
FQDD = CPU.Socket.2
InstanceID = CPU.Socket.2
LastSystemInventoryTime = 20100331101859
LastUpdateTime = 20100325134947
Manufacturer = Intel
MaxClockSpeed = 3600
Model = Intel(R) Xeon(R) CPU E5520 @ 2.27GHz
NumberOfEnabledCores = 4
NumberOfEnabledThreads = 8
NumberOfProcessorCores = 4
PrimaryStatus = 1
Voltage = 1.20
DCIM_CPUView
    CPUFamily = B3
    CPUSstatus = 1
    Cache1Associativity = 7
    Cache1ErrorMethodology = 5

9.5 iDRAC Card Inventory

This section describes the implementation for the DCIM_iDRACCARDView class. The Dell iDrac Profile describes the platform’s iDrac remote access card. Each remote access card’s information is represented by an instance of DCIM_iDRACCARDView class.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate DCIM_iDRACCARDView with the following parameters and syntax:

EXAMPLE:

```
winrm e cimv2/root/dcim/DCIM_iDRACCARDView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
```
9.6 **PCI Device Inventory**

This section describes the implementation for the DCIM_PCIEDeviceView class. The Dell PCI Profile describes platform’s PCI devices. Each PCI device’s information is represented by an instance of DCIM_PCIEDeviceView class.

**Profile and Associated MOFs:**
http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate DCIM_PCIEDeviceView with the following parameters and syntax:

**EXAMPLE:**
```
winrm e cimv2/root/dcim/DCIM_PCIEDeviceView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic -encoding:utf-8
-SkipCACheck
-SkipCNCheck
```

**OUTPUT:**
```
DCIM_PCIEDeviceView
BusNumber = 0
DataBusWidth = 0002
Description = 82801I (ICH9 Family) USB UHCI Controller #4
DeviceNumber = 26
FQDD = USBUHCI.Embedded.4-1
FunctionNumber = 0
InstanceID = USBUHCI.Embedded.4-1
LastSystemInventoryTime = 20100331101859
LastUpdateTime = 20100325134947
Manufacturer = Intel Corporation
PCIDeviceID = 2937
```
DCIM_PCIDeviceView

BusNumber = 0
DataBusWidth = 0002
Description = 5520/5500/X58 I/O Hub PCI Express Root Port 3
DeviceNumber = 3
FQDD = P2PBridge.Embedded.4-1
FunctionNumber = 0
InstanceID = P2PBridge.Embedded.4-1
LastSystemInventoryTime = 20100331101859
LastUpdateTime = 20100325134947
Manufacturer = Intel Corporation
PCIDeviceID = 340A
PCISubDeviceID = 0000
PCISubVendorID = 0000
PCIVendorID = 8086
SlotLength = 0002
SlotType = 0002

9.7 Video Inventory

This section describes the implementation for the DCIM_VideoView class. The Dell Video Profile describes platform’s videos. Each video controller’s information is represented by an instance of DCIM_VideoView class.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate DCIM_VideoView with the following parameters and syntax:

EXAMPLE:

winrm e cimv2/root/dcim/DCIM_VideoView
-u: [USER] -p: [PASSWORD]
-r: https://[IPADDRESS]/wsman:443
-auth: basic
-encoding: utf-8
-SkipCACheck
-SkipCNCheck

OUTPUT:

DCIM_VideoView

BusNumber = 6
DataBusWidth = 0002
Description = PowerEdge R610 MGA G200eW WPCM450
DeviceNumber = 3
FQDD = Video.Embedded.1-1
FunctionNumber = 0
InstanceID = Video.Embedded.1-1
LastSystemInventoryTime = 20100331101859
LastUpdateTime = 20100325134947
Manufacturer = Matrox Graphics, Inc.
PCIDeviceID = 0532
PCISubDeviceID = 0236
PCISubVendorID = 1028
PCIVendorID = 102B
SlotLength = 0002
SlotType = 0002

9.8 VFlash SD Card Inventory

Each SD card partition is represented by an instance of DCIM_VFlashView that is used to represent the physical attributes of the virtual flash media, such as total size, available size, category etc. on which the partitions will reside. See Section 13 for more information.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate the DCIM_VFlashView with the following parameters and syntax:

EXAMPLE:

winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_VFlashView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_VFlashView
   AvailableSize = 970
   Capacity = 976
   ComponentName = vFlash SD Card
   FQDD = Disk.vFlashCard.1
   HealthStatus = OK
   InitializedState = Initialized
   InstanceID = Disk.vFlashCard.1
   LastSystemInventoryTime = 20100408123517
   LastUpdateTime = 20100408123517
   Licensed = true
   VFlashEnabledState = true
   WriteProtected = false

9.9 NIC Inventory and Configuration

The NIC Profile describes NIC controller’s representation and configuration. The profile also describes the relationship of the NIC classes to the DMTF/Dell profile version information. See Section 15 for more information, including inventories for NICString, NICInteger, and NICEnumeration.
Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate NICView with the following parameters and syntax:

EXAMPLE:

```
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
DCIM_NICView
  AutoNegotiation = 0
  BusNumber = 1
  ControllerBIOSVersion = 1.3
  CurrentMACAddress = 0:21:9B:92:70:57
  DataBusWidth = 0002
  DeviceNumber = 0
  EFIVersion = null
  FCoEOffloadMode = 3
  FCoEWWNN = null
  FQDD = NIC.Embedded.1-1
  FamilyVersion = null
  FunctionNumber = 0
  InstanceID = NIC.Embedded.1-1
  LastSystemInventoryTime = 20100413135024
  LastUpdateTime = 20100413134727
  LinkDuplex = 0
  LinkSpeed = 0
  MaxBandwidth = 0
  MediaType = 4
  MinBandwidth = 0
  NicMode = 3
  PCIDeviceID = 1639
  PCISubDeviceID = 236
  PCISubVendorID = 1028
  PCIVendorID = 14E4
  PermanentFCOEMACAddress
  PermanentMACAddress = 00:21:9B:92:70:57
  PermanentISCSIMACAddress = 00:21:9B:92:70:58
  ProductName = Broadcom NetXtreme Gigabit
  Ethernet - 00:21:9B:92:70:57
  ReceiveFlowControl = 0
  SlotLength = 0002
  SlotType = 0002
```
9.10 RAID Inventory and Configuration

The RAID profile extends the management capabilities of referencing profiles by adding the capability to represent the configuration of RAID storage. The RAID storage is modeled as collections of attributes where there are collections for the storage adaptors, physical disks, logical disks, end enclosures and
parent-child relationships between the collections. Additionally, there is a configuration service that contains all the methods used to configure the RAID storage. See Section 16 for more information, including inventories for PhysicalDiskView, VirtualDiskView, and EnclosureView.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate ControllerView with the following parameters and syntax:

EXAMPLE:

```bash
Winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_ControllerView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
DCIM_ControllerView
  Bus = 3
  CacheSizeInMB = 1024
  CachecadeCapability = 1
  ControllerFirmwareVersion = 20.10.1-0049
  Device = 0
  DeviceCardDataBusWidth = 1
  DeviceCardManufacturer = DELL
  DeviceCardSlotLength = 3
  DeviceCardSlotType = PCI Express x8
  DriverVersion = null
  EncryptionCapability = 0
  EncryptionMode = 0
  FQDD = RAID.Integrated.1-1
  Function = 0
  InstanceID = RAID.Integrated.1-1
  LastSystemInventoryTime = 20100331101859
  LastUpdateTime = 20100330124133
  PCIDeviceID = 73
  PCISlot = 1
  PCISubDeviceID = 1F51
  PCISubVendorID = 1028
  PCIVendorID = 1000
  PatrolReadState = 1
  PrimaryStatus = 0
  ProductName = PERC H310 Mini
  RollupStatus = 0
  SASAddress = 50026B902A8B6E00
  SecurityStatus = 0
  SlicedVDCapability = 1
```
9.11 BIOS Inventory and Configuration

The BIOS Management Profile extends the management capabilities of referencing profiles by adding the capability to represent and configure BIOS attributes, such as a Network Controller or IDE Controller. The individual BIOS attribute’s relationship with a respective device is also described. Additionally, the profile’s registration for the schema implementation version information is described. See Section 17 for more information, including inventories for BIOSString, and BIOSInteger.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate BIOSEnumeration with the following parameters and syntax:

EXAMPLE:

Winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSEnumeration
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_BIOSEnumeration
  AttributeDisplayName = Memory Operating
  Voltage AttributeName = MemVolt
  CurrentValue = AutoVolt
  Dependency = <Dep>
    <AttrLev Op="OR">
      <ROIIf Op="NOT" Name="SysProfile">Custom</ROIIf>
    </AttrLev>
  <ValLev Val="AutoVolt" Op="OR">
    <ForceIf Name="SysProfile">PerfPerWattOptimizedDapc</ForceIf>
    <ForceIf Name="SysProfile">PerfPerWattOptimizedOs</ForceIf>
    <ForceIf Name="SysProfile">PerfOptimized</ForceIf>
    <SupIf Name="SysProfile">DenseCfgOptimized</SupIf>
  </ValLev>
  <ValLev Val="Volt15V" Op="OR">
    <ForceIf Name="SysProfile">DenseCfgOptimized</ForceIf>
    <SupIf Name="SysProfile">PerfPerWattOptimizedDapc</SupIf>
    <SupIf Name="SysProfile">PerfPerWattOptimizedOs</SupIf>
    <SupIf Name="SysProfile">PerfOptimized</SupIf>
  </ValLev>
</Dep>
DisplayOrder = 1322
FQDD = BIOS.Setup.1-1
GroupDisplayName = System Profile Settings
GroupID = SysProfileSettings
InstanceID = BIOS.Setup.1-1:MemVolt
IsReadOnly = true
PendingValue = null
PossibleValues = AutoVolt, Volt15V PossibleValuesDescription = Auto, 1.5V
DCIM_BIOSEnumeration
  AttributeDisplayName = Serial Debug Output
  AttributeName = SerialDbgOut
  CurrentValue = Disabled
  Dependency = null
  DisplayOrder = 319
  FQDD = BIOS.Setup.1-1
  GroupDisplayName = Memory Settings
  GroupID = MemSettings
  InstanceID = BIOS.Setup.1-1:SerialDbgOut
  IsReadOnly = false
  PendingValue = null
  PossibleValues = Enabled, Disabled
  PossibleValuesDescription = Enabled, Disabled

DCIM_BIOSEnumeration
  AttributeDisplayName = Password Status
  AttributeName = PasswordStatus
  CurrentValue = Unlocked
  Dependency = null
  DisplayOrder = 1405
  FQDD = BIOS.Setup.1-1
  GroupDisplayName = System Security
  GroupID = SysSecurity
  InstanceID = BIOS.Setup.1-1:PasswordStatus
  IsReadOnly = false
  PendingValue = null
  PossibleValues = Unlocked, Locked
  PossibleValuesDescription = Unlocked, Locked

### 9.12 System Inventory (Including CSIOR Attribute)

This section describes the implementation for the DCIM_SystemView class which is used to represent the higher level attributes of the System, such as asset tag, model, server manufacturer, etc.

**Profile and Associated MOFs:**

http://www.delltechcenter.com/page/DCIM.Library.Profile

Enumerate SystemView with the following parameters and syntax:

**EXAMPLE:**

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_SystemView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```
```python
DCIM_SystemView
    AssetTag = Testtg
    BIOSReleaseDate = 09/12/2011
    BIOSVersionString = 0.3.22
    BaseBoardChassisSlot = NA
    BatteryRollupStatus = 1
    BladeGeometry = 4
    BoardPartNumber = 0N051FX02
    BoardSerialNumber = CN1374092003M
    sCMCIP = null
    CPLDVersion = 0.4.7
    CPURollupStatus = 1
    ChassisName = Main System Chassis
    ChassisServiceTag = 7654321
    ChassisSystemHeight = 2
    ExpressServiceCode = 61387326761
    FQDD = System.Embedded.1
    FanRollupStatus = 3
    HostName
    InstanceID = System.Embedded.1
    LastSystemInventoryTime = 20100331101859
    LastUpdateTime = 20100325134947
    LicensingRollupStatus = 1
    LifecycleControllerVersion = 2.0.0
    Manufacturer = Dell Inc.
    MaxCPUSockets = 2
    MaxDIMMSlots = 24
    MaxPCIeSlots = 3
    MemoryOperationMode = OptimizerMode
    Model = PowerEdge R620
    PSRollupStatus = 1
    PlatformGUID = 3548474f-c0d3-4680-3810-00374c4c4544
    PopulatedCPUSockets = 1
    PopulatedDIMMSlots = 1
    PopulatedPCiESlots = 1
    PowerCap = 0
    PowerCapEnabledState = 3
    PowerState = 2
    PrimaryStatus = 3
    RollupStatus = 3
    ServerAllocation = null
    ServiceTag = S78FGH5
    StorageRollupStatus = 1
```
SysMemErrorMethodology = 6
SysMemFailOverState = NotInUse
SysMemLocation = 3
SysMemPrimaryStatus = 1
SysMemTotalSize = 2048
SystemGeneration = 12G Monolithic
SystemID = 1230
SystemRevision = 0
TempRollupStatus = 1
UUID = 4c4c4544-0037-3810-8046-d3c04f474835
VoltRollupStatus = 1
smbiosGUID = 44454c4c-3700-1038-8046-d3c04f474835

9.13 **USB Device Inventory**
This feature is new in 13th Generation of Dell PowerEdge Servers.

9.14 **System Quick Sync Hardware Inventory**
This feature is new in 13th Generation of Dell PowerEdge Servers.

9.15 **PCIe SSD Inventory**
This feature is new in 13th Generation of Dell PowerEdge Servers.

9.16 **PCIe SSD Extender Inventory**
This feature is new in 13th Generation of Dell PowerEdge Servers.

9.17 **PCIe SSD Backplane Inventory**
This feature is new in 13th Generation of Dell PowerEdge Servers.

9.18 **Host Network Interface Inventory**
This feature is new in 13th Generation of Dell PowerEdge Servers.

9.19 **Physical Computer System Inventory**
This feature is new in 13th Generation of Dell PowerEdge Servers.
10   Job Control Management

10.1 Description of Job Management

The Dell Common Information Model (CIM) class extensions for supporting update and attribute configuration job control are defined in the Dell Job Control Profile 2 and related MOF files. The diagrams representing the classes that are implemented by the Lifecycle Controller firmware can be found in Dell Job Control Profile as well.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

10.2 Remote Job Control Examples

10.2.1 Setup Job Queue

The SetupJobQueue() method takes in an array of jobids and schedules them to run immediately or at a later time. The jobids are acquired via enumerating DCIM_LifecycleJob as described in Section 10.2.3. When there is a Reboot Job, in a job array that contains multiple jobs, the System will reboot the UEFI (Unified Extensible Firmware Interface) at the scheduled time.

Invoke SetupJobQueue() with the following parameters and syntax:

JobArray: The jobids are listed in the JobArray element. Multiple jobs are listed in the order of job execution sequence. If a System is to reboot at the scheduled start time, a reboot job will need to be added to the list. This reboot job has a prefix of RID_ for its jobid.

Note, scheduling a job that is already scheduled will result in an error message.

If there is no reboot job in the job array, the System will schedule the jobs for execution at the specified start time. The jobs will not be executed until the System is rebooted by something other than Lifecycle Controller. At the specified UntilTime, any jobs that have not been executed are failed with an error indicating that the job was not executed in the specified maintenance window. For some component updates such as Diagnostics, USC, and iDRAC firmware, a System reboot is not needed.

EXAMPLE:

```
winrm invoke SetupJobQueue

cimv2/root/dcim/DCIM_JobService?CreationClassName=DCIM_JobService
+Name=JobService
+SystemName=Idrac
+SystemCreationClassName=DCIM_ComputerSystem
-file:SetupJobQueue.xml
-u:USER -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
```

The syntax for SetupJobQueue.xml is:

```
<p:SetupJobQueue_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_JobService">
  <p:JobArray>JID_001249463339</p:JobArray>
  <p:JobArray>RID_001265817718</p:JobArray>
  <p:StartTimeInterval>TIME_NOW</p:StartTimeInterval>
  <p:UntilTime>20100730121500</p:UntilTime>
</p:SetupJobQueue_INPUT>
```
Here the JobArray element shows a list of Jobids that are to be scheduled to run. TIME_NOW is a special value that represents “running the tasks immediately”. The UntilTime value specifies the “maintenance windows”. Once a task is not run after passing UntilTime, it should not be run again.

Upon successfully invocation of the SetupJobQueue() method, the aforementioned times will be listed when enumerated in Section 10.2.3.

**OUTPUT:**

Returns 0 for success or non-zero for error with messageID and message description.

```java
SetupJobQueue_OUTPUT
ReturnValue = null
```

Entering an invalid jobid or XML syntax error can yield one of the following error messages:

```java
SetupJobQueue_OUTPUT
Message = Job Cannot be Scheduled
MessageID = SUP016
ReturnValue = null
```

```java
SetupJobQueue_OUTPUT
Message = Duplicated/Invalid JOBID Entries
MessageID = SUP023
ReturnValue = null
```

### 10.2.2 Delete Job Queue

The DeleteJobQueue() method takes in a jobID and then deletes it from the job store.

**Note:** When clearing all jobs and pending data using the keyword JID_CLEARALL, as shown in example 2 below, the remote services instrumentation is restarted to clear the cache [LC 1.x ONLY]. Users should allow two minutes for this process to complete.

Invoke DeleteJobQueue() with the following parameters and syntax:

```bash
[JobID]: The jobID of a particular job instance to be deleted from a jobqueue
```

**EXAMPLE 1:**

```bash
winrm invoke DeleteJobQueue
\cimv2/root/dcim/DCIM_JobService?CreationClassName=DCIM_JobService
+Name=JobService +SystemName=Idrac
+SystemCreationClassName=DCIM_ComputerSystem @(JobID="[jobID]"
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
```

The example below uses JID_CLEARALL for the jobID, which is a predefined value that represents “deleting all jobs in the jobstore”.

**EXAMPLE 2:**

```bash
winrm invoke DeleteJobQueue
\cimv2/root/dcim/DCIM_JobService?CreationClassName=DCIM_JobService
+Name=JobService +SystemName=Idrac
+SystemCreationClassName=DCIM_ComputerSystem @(JobID="JID_CLEARALL"
-u:[USER] -p:[PASSWORD]
-r: https://[IPADDRESS]/wsman:443
-auth:basic
-encoding:utf-8
-SkipCACheck
-SkipCNCheck
```
OUTPUT:
Return 0 for success or non-zero for error with messageID and message description.

DeleteJobQueue_OUTPUT
  Message = The specified job was deleted
  MessageID = SUP020
  ReturnValue = null

An XML syntax error could yield the following message:
  Syntax Error: input must be of the form:
  {KEY="VALUE";KEY="VALUE"}

10.2.3 List Jobs in Job Store

The instances of this class will enumerate jobs in the job store along with status information.

Invoke enumerate job status with the following parameters and syntax:

[JobID]: The JobID of a particular job instance to be queried

To get the status of one particular job, use the following:

EXAMPLE 1:
  winrm get
  http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LifecycleJob?
  InstanceID=[JobID]
  -r:https://[IPADDRESS]/wsman:443
  -u:[USERNAME] -p:[PASSWORD]
  -a:basic encoding:utf-8

To get the status of all jobs, use the following:

EXAMPLE 2:
  winrm e cimv2/root/dcim/DCIM_LifecycleJob
  -u:[USERNAME] -p:[PASSWORD]
  -r:https://[IPADDRESS]/wsman:443
  -auth:basic
  -encoding:utf-8

OUTPUT 1 and 2:
The method either returns a list of Concrete job objects or an error message. Once job instanceID is returned via these status queries, they can be used for job scheduling and setup. Several examples of job objects are shown below.

DCIM_LifecycleJob
  InstanceID = JID_001275884806
  JobStartTime
  JobStatus = Completed
  JobUntilTime
  Message = Detach partition successful
  MessageArguments = null
  MessageID = VF038
  Name = VFlashDetach:Partition3

DCIM_LifecycleJob
  InstanceID = RID_001274051062
  JobStartTime = 00000101000000
  JobStatus = Reboot Completed JobUntilTime = 20100730121500
Message
MessageArguments = null
MessageID
Name = Reboot3

DCIM_LifecycleJob
InstanceID = JID_001274140369
JobStartTime = 00000101000000 JobStatus = Completed
JobUntilTime = 20111111111111 Message = Job completed successfully
MessageArguments = null
MessageID = PR19
Name = ConfigRAID:RAID.Integrated.1-1

An error message similar to the following can occur if an invalid JobID is entered:

WSManFault
Message = The WinRM client cannot process the request. The destination computer returned an empty response to the request.
Error number: -2144108299 0x803380F5
The WinRM client cannot process the request. The destination computer returned an empty response to the request.
11 Operating System Deployment

The Dell Common Information Model (CIM) class extensions for supporting remote operating System (OS) deployment are defined in the Dell OS Deployment Profile 2 and the DCIM_OSDeploymentService MOF file 3. The diagrams representing the classes that are implemented by the Lifecycle Controller firmware can be found in Dell OS Deployment Profile as well.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

11.1 OS Deployment Profile Implementation Conformance

Use the following algorithm to test the instrumentation for OS Deployment Profile version conformance and to discover the implementation namespace:

1. Enumerate (namespace='root/interop', classname="CIM_RegisteredProfile")
2. Filter the returned enumeration using property filter (RegisteredName="OS Deployment")
3. Result shall contain one instance of CIM_RegisteredProfile containing property RegisteredVersion="1.1.0"
4. Associators (objectpath= “instance returned from step 3”, AssociationClass = "CIM_ElementConformsToProfile")
5. Result shall contain one instance of DCIM_OSDeploymentService

11.2 Checking OS Deployment Service Availability

Invoke enumerate with the following syntax:

EXAMPLE:

winrm e cimv2/root/dcim/DCIM_OSDeploymentService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_OSDeploymentService
  CreationClassName = DCIM_OSDeploymentService
  ElementName = Operating System Deployment Service
  Name = DCIM:OSDeploymentService
  SystemCreationClassName = DCIM_ComputerSystem
  SystemName = DCIM:ComputerSystem

11.3 OS Deployment Method Invocation Examples

11.3.1 Get Driver Pack Information

The GetDriverPackInfo() method returns the embedded driver pack version and list of supported OSs for OS deployment that can be installed on the server using the embedded device drivers present in the Lifecycle Controller.

1. Follow the steps listed in Section 11.1 to test for profile conformance.
2. Invoke extrinsic method using the following parameters:
   a. object path = object path returned from Section 11.1 (profile conformance)
   b. Method name = “GetDriverPackInfo”
3. Invoke method returns the following output parameters:
   a. Version = String version
   b. SupportedOperatingSystems = String array of OS names
4. If the Job output parameter from Step 2 contains a non-null value, then both Version and OSList contain null values. The next call to GetDriverPackInfo() after the Job is completed will return nonnull values for output parameters Version and OSList. Invoke GetDriverPackInfo() with the following syntax:

EXAMPLE:

```plaintext
winrm i GetDriverPackInfo
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-encoding:utf-8 -a:basic
```

OUTPUT:

```plaintext
GetDriverPackInfo_OUTPUT
OSList = Microsoft Windows Server 2008 with SP2,
Microsoft Windows Server 2008, x64 with SP2,
Microsoft Windows Server 2008 R2 with SP1,
Microsoft Windows Small Business Server 2011,
Red Hat Enterprise Linux 5 SP7 x86,
Red Hat Enterprise Linux 5 SP7 x64,
Red Hat Enterprise Linux 6 SP1 x64,
SuSE Enterprise Linux 10 SP4 x64,
SuSE Enterprise Linux 11 SP2 x64,
VMware ESX 4.1 U2,
VMware ESXi 4.1 U2 HDD,
VMware ESXi 5.0 HDD,
Citrix Xen Server 6.0 FP1 HDD
Return_Value = 0
Version = 7.0.0.35
```

11.3.2 Unpack Selected Drivers and Attach to Host OS as USB Device

This method is used to unpack the drivers for the selected OS to a virtual storage partition, and to then attach this partition to the host OS as an emulated USB storage device.

1. Invoke extrinsic method using the following parameters section:
   a. object path = object path returned from Section 11.1 (profile conformance)
   b. Method name = “UnpackAndAttach”
   c. OSName = “” (Has to be a valid value from the list returned by GetDriverPackInfo)
   d. ExposureStartTime = “” (for this release the value is NULL)
   e. ExposureDuration = “” (a string formatted as an interval in CIM_DateTime format). This parameter denotes the interval of time after which the partition containing OS drivers with label OEMDRV is to be detached from the Host OS.

2. Invoke method shall return the following output parameters:
   a. Job = object path to CIM_ConcreteJob (reports the status of unpack and attach)
   b. Enumerating this instance of CIM_ConcreteJob will show the status of the current operation.

Invoke UnpackAndAttach() with the following syntax:
EXAMPLE:

```
winrm i UnpackAndAttach
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u: [USER] -p: [PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a: basic
@{OSName="[OSName]";ExposureDuration="0000000002200.000000:000"}
```

Above example uses Microsoft Windows Server 2008 with SP2 for OSName.

OUTPUT:

```
UnpackAndAttach_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet
      Selector: InstanceID = DCIM_OSDConcreteJob:1,__cimnamespace = root/dcim
ReturnValue = 4096
```

11.3.3 Detach Emulated USB Device Containing Drivers

This method is used to detach the USB device attached to the System by a previous invocation of the UnpackAndAttach() method.

Invoke DetachDrivers() with the following syntax:

EXAMPLE:

```
winrm i DetachDrivers
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u: [USER] -p: [PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a: basic
```

OUTPUT:

The return will be 0 for success or an integer for error or job in execution. An error message containing a MessageID and Message similar to the following can occur if the System is waiting to finish a previously invoked method:

```
DetachDrivers_OUTPUT
   Message = Unable to retrieve Lifecycle Controller handle
   MessageID = OSD7
   ReturnValue = 2
```
11.3.4 Unpack Selected Drivers and Copy to Network Share

The UnpackAndShare() method is used to unpack the drivers for the selected OS and copy them to a specified network share; CIFS and NFS network share technologies are supported.

Note that the values for the CIFSUSER and CIFSPASSWORD must be alphanumeric characters, and must not contain special characters.

Invoke UnpackAndShare() with the following syntax:

```
[CIFS_IPADDRESS]: This is the IP address of the file server.
[DRIVESHARE]: This is the directory path to the drivers.
[CIFS_USERNAME]: This is the username to the file share.
[CIFS_PASSWORD]: This is the password to the file share.
[OSName]: This example uses Windows Server® 2003 SP2.
[NFS_Password]: This is the corresponding password to the username containing the ISO
```

**EXAMPLE:**

```
winrm i UnpackAndShare
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]:443/wsman
-encoding:utf-8 a:basic
@{IPAddress="[CIFS_IPADDRESS]";ShareName="/[DRIVERSHARE]";ShareType="2";
Username ="[CIFS_USERNAME]";Password="[CIFS_PASSWORD]";OSName="Microsoft Windows Server 2008 with SP2"}
```

**OUTPUT:**

The return will be 0 for success or 1 if an error occurred in starting the processing the input parameters. The MessageID and Message output parameters will further contain method invocation information if an error occurred.

```
UnpackAndShare_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
    ResourceURI =
        http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_OSDConcreteJob
SelectorSet
    Selector: InstanceID = DCIM_OSDConcreteJob:1, __cimnamespace = root/dcim
ReturnValue = 4096
```

A missing command line character, such as a “{”, could result in the following syntax error: Syntax Error: input must be of the form {KEY="VALUE"[;KEY="VALUE"]}

11.3.5 Check Job Status

The following methodology is used to determine the status of the jobs generated by the invocation of the UnpackAndAttach() and UnpackAndShare() methods. The methodology involves enumerating the DCIM_OSDConcreteJob instances, and checking the JobStatus property value.

When the jobs are complete, the JobStatus property value will be “Successful” if the job completed successfully or “Failed” if an error occurred while executing the request. If the job failed, the Message property on the returned DCIM_OSDConcreteJob instance will contain more detailed error information on the cause of the failure.
For the Lifecycle Controller version of the OS Deployment Profile there is only one instance of a job generated by various method invocations, and it will persist until the next method that generates a job is invoked. The job must complete before another method that generates a job can be called successfully. This is unchanged from the Lifecycle Controller 1.2 for OS Deployment.

Invoke enumerate DCIM_OSDConcreteJob instance with the following syntax:

**EXAMPLE:**

```
winrm e cimv2/root/DCIM/DCIM_OSDConcreteJob
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNCheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**
The enumeration will return the instances of OSDConcreteJob as shown:

```
DCIM_OSDConcreteJob
    DeleteOnCompletion = false
    InstanceID = DCIM_OSDConcreteJob:1
    JobName = UnpackAndShare
    JobStatus = Failed
    Message = Installation not supported for the selected operating System
    MessageID = OSD10
    Name = UnpackAndShare
```

### 11.3.6 Boot to Network ISO

The BootToNetworkISO() method can be used to boot the target System to a bootable ISO image located on a CIFS or NFS share. The ISO image is attached to the host System as an emulated USB CD-ROM storage device. By default the ISO will be attached for around 18 hrs after which it will be detached automatically. An optional parameter ExposeDuration can be used to specify a time less than 18 hrs if the ISO needs to be detached sooner.

Invoke BootToNetworkISO() via NFS share with the following syntax:

**[NFS_IPADDRESS]:** This is the IP address of the location of the ISO image.

**[/NFS/OSISO]:** This is the directory path to the ISO image.

**[NFS_Username]:** This is the username to the IP address of the ISO image.

**[NFS_Password]:** This is the corresponding password to the username containing the ISO image.

**[OS.ISO]:** This is to be replaced by the actual name of the ISO image.

**EXAMPLE:**

```
winrm i BootToNetworkISO
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
{Name=DCIM:OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u: [USER] -p: [PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNCheck encoding:utf-8 -a:basic
@{IPAddress="[NFS_IPaddress]";ShareName="/NFS/OSISO";ShareType="0";
Username="[NFS_Username]";Password="[NFS_Password]";
Workgroup="WORKGROUP";ImageName="[OS.ISO]"}
```
11.3.7 Detach Network ISO USB Device

This method is used to detach the emulated USB device that had been attached by previously calling the BootToNetworkISO() method.

Invoke DetachISOImage() with the following syntax:

```
winrm i DetachISOImage
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM:ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a:basic
```

**OUTPUT:**
The method will return 0 for success or an integer for error or job in execution. An error such as the following can occur if an ISO image is not attached.

```
DetachISOImage_OUTPUT
Message = ISO image is not attached
MessageID = OSD32 ReturnValue = 2
```

11.3.8 Boot To PXE

The BootToPXE() method is used to boot to server using the PXE mechanism, which is to reboot the host server and boot to PXE.

Invoke to boot target System to PXE with the following syntax:
EXAMPLE:

winrm i BootToPXE
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a:basic

The return will be 0 for success or 1 if an error occurred in starting the processing the input parameters. The MessageID and Message output parameters will further contain method invocation information if an error occurred.

OUTPUT:

BootToPXE_OUTPUT
ReturnValue = 0

11.3.9 Get Host MAC Address Information

Invoke GetHostMACInfo() with the following syntax:

EXAMPLE:

winrm i GetHostMACInfo
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNCheck encoding:utf-8 -a:basic

OUTPUT:

The return will be 0 for success and a list of MAC addresses or an integer for error or job in execution. The MessageID and Message output parameters will further contain method invocation information if an error occurred.

GetHostMACInfo_OUTPUT

MACList = 00219b927057, 00219b927059, 00219b92705b, 00219b92705d
ReturnValue = 0

11.3.10 Download ISO to VFlash

The DownloadISOToVFlash() method allows using remote command to download an ISO image to VFlash. The image needs to be an ISO image. Once this image is downloaded to VFlash, it can be booted via another WS-Management command.

Invoke DownloadISOToVFlash() with the following parameters and syntax:

[IPADDRESS-ISO]: The IP address of the server that stores ISO images.
[DRIVESHARE]: This is the directory path to the ISO image.
[SHARETYPE]: The type of the remote storage. 0: NFS, 1: TFTP, 2: CIFS
[SHAREUSER]: User account for the ISO share location
[SHAREPASSWORD]: Password of the share account
[WORKGROUP]: Applicable workgroup
[IMAGENAME]: Image name of the iso image, such as boot.iso.
**[Port]:** Port number for connecting to the share, such as 2049.

**EXAMPLE:**

```shell
winrm i DownloadISOToVFlash
cimv2/root/dcm/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsmans:443
-encoding:utf-8 -a:basic @([IPADDRESS]=[IPADDRESSISO];ShareName="/([DRIVE]SHARE]";ShareType="[SHARETYPE]";Username="[SHAREUSER]";
Password="[SHAREPASSWORD]";Workgroup="[WORKGROUP]";
ImageName="[IMAGENAME]";Port="[PORT]"
```

**OUTPUT:**

The return will be 0 for success or 1 if an error occurred in starting the processing the input parameters. The MessageID and Message output parameters will further contain method invocation information if an error occurred.

**DownloadISOToVFlash\_OUTPUT**

```xml
Job
   Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
   ResourceURI =
      http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_OSDConcreteJob
   SelectorSet
      Selector: InstanceID = DCIM_OSDConcreteJob:1, __cimnamespace = root/dcim
   ReturnValue = 4096
```

The following error message is a direct result of a typo in the winRM input. Careful consideration must be applied to capitalization.

**WSManFault**

```plaintext
Message = The WinRM client cannot process the request. The destination computer returned an empty response to the request.
Error number: -2144108299 0x803380F5
```

The WinRM client cannot process the request. The destination computer returned an empty response to the request.

### 11.3.11 Boot to ISO from VFlash

This method will expose the ISO image present on VFlash as a CDROM device to the host server and boots to it.

Invoke `BootToISOFromVFlash()` with the following syntax:

**EXAMPLE:**

```shell
winrm i BootToISOFromVFlash
cimv2/root/dcm/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsmans:443
-SkipCNCheck encoding:utf-8 -a:basic
```
OUTPUT:
When this command is executed, a status or error message will be returned.

BootToISOFromVFlash_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
   ReferenceParameters
   ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_OSDConcreteJob
   SelectorSet
      Selector: InstanceID = DCIM_OSDConcreteJob:1, __cimnamespace = root/dcim
      ReturnValue = 4096

11.3.12 Delete ISO from VFlash
The DeleteISOFromVFlash() method will delete the ISO image that was downloaded to the VFlash.
Invoke DeleteISOFromVFlash() with the following syntax:
EXAMPLE:
winrm i DeleteISOFromVFlash
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+NName=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USERNAME] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a:basic

OUTPUT:
When this command is executed, a status or error message will be returned. If an image is not found the following message will display:
DeleteISOFromVFlash_OUTPUT
   Message = ISO Image not found on VFlash
   MessageID = OSD41
   ReturnValue = 2

11.3.13 Detach ISO from VFlash
The DetachISOFromVFlash() method will detach the ISO image in the VFlash from the System.
Invoke DetachISOFromVFlash() with the following syntax:
EXAMPLE:
winrm i DetachISOFromVFlash
cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+NName=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USERNAME] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a:basic
OUTPUT:
When this command is executed, a status or error message will be returned. If an image is not found the following message will display:

```
DetachISOFromVFlash_OUTPUT
    Message = Unable to detach ISO image on VFlash
    MessageID = OSD44
    ReturnValue = 2
```

11.3.14 Connect Network ISO Image
This method can be used to connect to a bootable ISO image located on a CIFS or NFS share. The ISO image is attached to the host System as an emulated USB CD-ROM storage device. Whenever the host System reboots it will boot to this ISO Image every single time until DisconnectNetworkISOImage is called. The ISO will be reattached upon iDRAC reset.

Invoke ConnectNetworkISOImage() using CIFS or NFS share with the following syntax:

```
[CIFS_or_NFS_IPADDRESS]: This is the IP address of the location of the ISO image.
/[CIFS_or_NFS/OSISO]: This is the sharename directory path to the ISO image.
[2_or_0]: 2=CIFS, 0=NFS
[CIFS_or_NFS_Username]: This is the username to the IP address of the ISO image.
[CIFS_or_NFS_Password]: This is the corresponding password to the username containing the ISO image.
[OS.ISO]: This is to be replaced by the actual name of the ISO image.
```

EXAMPLE:
```
winrm i ConnectNetworkISOImage
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
@{IPAddress="[CIFS_or_NFS_IPaddress]";ShareName="/[CIFS_or_NFS]";
ShareType="[2_or_0]";Username="[CIFS_or_NFS_Username]";
Password="[CIFS_or_NFS_Password]";Workgroup="WORKGROUP"; ImageName="[OS.ISO]"}
```

OUTPUT:
The return will be 0 for success or 1 if an error occurred in starting the processing the input parameters. The MessageID and Message output parameters will further contain method invocation information if an error occurred.

```
ConnectNetworkISOImage_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
    ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_OSDConcreteJobschema/2/DCIM_OSDConcreteJob
SelectorSet
    Selector: InstanceID = DCIM_OSDConcreteJob:1,__cimnamespace = root/dcim
ReturnValue = 4096
```
11.3.15 Disconnect Network ISO Image

This method can be used to disconnect the target System from a bootable ISO image located on a CIFS or NFS share.

Invoke DisconnectNetworkISOImage() with the following syntax:

**EXAMPLE:**

```
winrm i DisconnectNetworkISOImage
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService +SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

The return will be 0 for success or 1 if an error occurred in starting the processing the input parameters. The MessageID and Message output parameters will further contain method invocation information if an error occurred.

```
DisconnectNetworkISOImage_OUTPUT
ReturnValue = 0
```

11.3.16 Skip ISO Image Boot

This method can be used to skip the target System from booting to a bootable ISO image (connected using ConnectNetworkISOImage method) one time only for next immediate host reboot. After that host server will continue to boot to the ISO image.

Invoke SkipISOImageBoot() via NFS share with the following syntax:

**EXAMPLE:**

```
winrm i SkipISOImageBoot
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService +SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

Shown below are return messages of failure and success, 2 and 0, respectively. The MessageID and Message output parameters will further contain method invocation information if an error occurred.

**Failure:**

```
SkipISOImageBoot_OUTPUT
Message = ISO image is not attached
MessageID = OSD32
ReturnValue = 2
```

**Success:**

```
SkipISOImageBoot_OUTPUT
ReturnValue = 0
```
11.3.17 Get Network ISO Image Connection Information

This method outputs the ISO connection status of the image that has been exposed to the host.

Invoke GetNetworkISOImageConnectionInfo() with the following syntax:

EXAMPLE:

```
winrm i
GetNetworkISOImageConnectionInfo cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
GetNetworkISOImageConnectionInfo_OUTPUT
Message = ISO image is not attached
MessageID = OSD32
ReturnValue = 2
```

11.3.18 Connect RFS ISO Image

The ConnectRFSISOImage() method is used to connect the ISO image that is mounted through Remote File Share (RFS) and is exposed to the host System as a USB-based CD-ROM device. The successful execution of this method shall connect to the ISO located on NFS/CIFS share to the host server and expose it as a virtual CDROM device using RFS USB endpoint. The successful execution of the method shall not change the boot order of that device. In order to boot to the CD-ROM, the CD-ROM shall be configured in the boot order in a separate step (using BIOS and Boot Management Profile), and the host server shall boot to the CD-ROM. Unlike the ConnectNetworkISOImage() method, the Lifecycle Controller is not locked and may perform other management tasks.

Invoke ConnectRFSISOImage() with the following syntax:

```
[IPADDRESS-ISO]: The IP address of the server that stores ISO images.
[DRIVESHARE]: This is the directory path to the ISO image.
[SHARETYPE]: The type of the remote storage. 0: NFS, 2: CIFS
[SHAREUSER]: User account for the ISO share location
[SHAREPASSWORD]: Password of the share account
[WORKGROUP]: Applicable workgroup
[IMAGENAME]: Image name of the iso image, such as boot.iso.
```

EXAMPLE:

```
winrm i
ConnectRFSISOImage cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a:basic @{IPAddress=[IPADDRESSISO];ShareName="/[DRIVESHARE]";ShareType="[SHARETYPE]";Username="[SHAREUSER]";Password="[SHAREPASSWORD]";Workgroup="[WORKGROUP]";ImageName="[IMAGENAME]"}
```
11.3.19 Disconnect RFS ISO Image

The DisconnectRFSISOImage() method is used to disconnect and detach the ISO Image that is mounted through Remote File Share (RFS) and is exposed to the host System as a USB-based CD-ROM device.

Invoke DisconnectRFSISOImage() with the following syntax:

EXAMPLE:

```
winrm i DisconnectRFSISOImage cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding=utf-8 a:basic
```

OUTPUT:

```
DisconnectRFSISOImage_OUTPUT
ReturnValue = 0
```

11.3.20 Get RFS ISO Image Connection Information

The GetRFSISOImageConnectionInfo() method is used to provide the status of the ISO Image connection that has been exposed to the host System.

Invoke GetRFSISOImageConnectionInfo() with the following syntax:

EXAMPLE:

```
winrm i GetRFSISOImageConnectionInfo cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
+Name=DCIM:OSDeploymentService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding=utf-8 a:basic
```

Concrete jobs return 4096 upon successful invocation. Poll for the concrete job “JobStatus = Success”.
OUTPUT:
GetRFSISOMessageConnectionInfo_OUTPUT
  Message = Unable to connect to ISO using RFS.
  MessageID = OSD60
  ReturnValue = 2

A return value of 0 indicates success, while the above output indicates an image was not present to retrieve the connection information from.

11.3.21 Boot to Hard Disk Drive (HDD)
The BootToHD() method is used for one time boot to the host server's hard disk. After this method is executed the host is rebooted immediately and will boot to the first configured hard disk irrespective of its boot order.

Invoke BootToHD() with the following syntax:

EXAMPLE:
winrm i
  BootToHD cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
  +Name=DCIM:ODeploymentService
  +SystemCreationClassName=DCIM_ComputerSystem
  +SystemName=DCIM:ComputerSystem
  -u:[USER] -p:[PASSWORD]
  -r:https://[IPADDRESS]/wsman:443
  -encoding:utf-8 -a:basic

OUTPUT:
BootToHD_OUTPUT
  ReturnValue = 0

11.3.22 Configurable Boot to Network ISO
This method was added during the LC2 Version 1.1 release.

The ConfigurableBootToNetworkISO() works similar to BootToNetworkISO() except that the immediate boot to the ISO is not automatic and controlled by an input parameter called ResetType which will enable you to do a warm reset or cold reset or no immediate reset.

Invoke ConfigurableBootToNetworkISO() via NFS share with the following syntax:

[NFS_IPADDRESS]: This is the IP address of the location of the ISO image.
/NFS/OSISO/: This is the directory path to the ISO image.
[NFS_Username]: This is the username to the IP address of the ISO image.
[NFS_Password]: This is the corresponding password to the username containing the ISO image.
[OS.ISO]: This is to be replaced by the actual name of the ISO image.
[RESET_TYPE]: 0=No reset, 1=warm reset 2=cold reset

EXAMPLE:
winrm i
  ConfigurableBootToNetworkISO cimv2/root/dcim/DCIM_OSDeploymentService?CreationClassName=DCIM_OSDeploymentService
  +Name=DCIM:ODeploymentService
  +SystemCreationClassName=DCIM_ComputerSystem
  +SystemName=DCIM:ComputerSystem
  -u:[USER] -p:[PASSWORD]
  -r:https://[IPADDRESS]/wsman:443
  -SkipCNCheck
-encoding:utf-8 -a:basic
@{IPAddress="[NFS_IPaddress]";ShareName="/NFS/OSISO";ShareType="0"
Username="[NFS_Username]";Password="[NFS_Password]";ResetType="[RESET_TYPE]"
Workgroup="WORKGROUP";ImageName="[OS.ISO]"}

OUTPUT:
The return will be 0 for success or 1 if an error occurred in starting the processing the input parameters.
The MessageID and Message output parameters will further contain method invocation information if an error occurred.

ConfigurableBootToNetworkISO_OUTPUT

Job
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
  ResourceURI = 
    http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_OSDConcreteJob
  SelectorSet
    Selector: InstanceID = DCIM_OSDConcreteJob:1, __cimnamespace = 
      root/dcim
  ReturnValue = 4096
12 Lifecycle Controller Management Profile

The LC Management Profile describes the LC attribute configuration service and the collections and attributes instances that the service manages. The profile also describes the relationship of the LC attribute service to the DMTF/Dell profile version information and Dell Job Control profile.

The Dell Common Information Model (CIM) class extensions for supporting Lifecycle Controller feature management are defined in the Dell LC Management 2 and related MOF files. The diagrams representing the classes that are implemented by the Lifecycle Controller firmware can be found in Dell LC Management Profile. Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

12.1 Collect System Inventory on Restart (CSIOR)

By default, 'collect System inventory on restart' is disabled. To enable this feature, utilize the SetAttribute() method in the following example.

**Note:** To query the System to determine when the last CSIOR event occurred, list System inventory and examine the LastSystemInventoryTime attribute.

The Collect System Inventory on Restart attribute flags whether the System should do an automatic inventory or not. To get the current status of this attribute, see Section 12.3. The values can be:

- Disabled (default) = Disallow collecting inventory on restart
- Enabled = Allow collecting System inventory on restart

The Part Firmware Update attribute flags whether the Part Replacement automatic firmware update performed. The values can be:

- Disable (default) = firmware update is not allowed
- Allow version upgrade only = Allow firmware update only on up-revision
- Match firmware of replaced part = Always update firmware

The example below configures the Part Replacement feature to allow upgrade only and for the automatic synchronization to be on.

Invoke SetAttribute() with the following parameters and syntax:

**EXAMPLE 1:**

```plaintext
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-r:https://[IPADDRESS]:443/wsman
-u:[USER] -p:[PASSWORD]
-auth:basic
-encoding:utf-8
-SkipCNcheck -SkipCACheck
```

The input file SetAttribute_LC.xml is shown below:

```xml
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService">
  <p:AttributeName>Part Firmware Update</p:AttributeName>
  <p:AttributeValue>Allow version upgrade only</p:AttributeValue>
</p:SetAttribute_INPUT>
```

This method is used to set the values of multiple attributes.

Invoke SetAttributes() with the following parameters and syntax:

**EXAMPLE 2:**
winrm i SetAttributes
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService
?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService -file:[DIRECTORYPATH]\SetAttributes_LC.xml
-r:https://[IPADDRESS]:443/wsman
-u:[USER] -p:[PASSWORD] -auth:basic
-encoding:utf-8
-SkipCNcheck -SkipCACheck

The input file SetAttributes_LC.xml is shown below:

```xml
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
  <p:AttributeName>Part Firmware Update</p:AttributeName>
  <p:AttributeValue>Allow version upgrade only</p:AttributeValue>
  <p:AttributeName>Collect System Inventory on Restart</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
</p:SetAttributes_INPUT>

OUTPUT:
SetAttribute_OUTPUT
  RebootRequired = No
  ReturnValue = 0
  SetResult = Set PendingValue
```

12.2 Part Replacement Configuration and Management
If the SetAttribute[s]() method has been invoked, the pending values must be applied by creating a configuration job. The CreateConfigJob() method in the DCIM_LCService class creates a configuration job and executes it at the specified time.

12.2.1 Create Config Job
Invoke CreateConfigJob() with the following parameters and syntax:

EXAMPLE:

winrm i CreateConfigJob
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService file:[DIRECTORYPATH]\CreateConfigJob.xml
-r:https://[IPADDRESS]:443/wsman -u:[USER] -p:[PASSWORD]
-auth:basic
-encoding:utf-8
-SkipCNcheck -SkipCACheck

The input file CreateConfigJob.xml is shown below:

```xml
<p:CreateConfigJob_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
  <p:ScheduledStartTime>00000000002200.000000:000</p:ScheduledStartTime>
  <p:RebootIfRequired>false</p:RebootIfRequired>
```
The above command will schedule the job at 10pm. To poll for job completion, enumerate the DCIM_LifecycleJob job instance.

**OUTPUT:**

CreateConfigJob_OUTPUT

Job Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous

   ReferenceParameters


   SelectorSet

   Selector: InstanceID = JID_001265982202, __cimnamespace = root/dcim

   ReturnValue = 0

To get the status of the above jobID or list all jobIDs, see 12.2.2 and 12.2.3, respectively.

### 12.2.2 Get LC Config Job Status

**EXAMPLE:**

```
winrm g
+InstanceId=JID_001265982202
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]:wsman -encoding:utf-8 -a:basic
-SkipCNcheck -SkipCACheck
```

The method either returns a list of Concrete job objects or an error message. Check for the JobStatus property equal to Completed (shown below) to know the set has been completed.

**OUTPUT:**

```
DCIM_LifecycleJob

   InstanceID = JID_001265982202
   JobStartTime = 20191010101010
   JobStatus = COMPLETED
   JobUntilTime = 2009:8:11
   Message = The command was successful
   MessageArguments = null
   MessageID = LC001
   Name = LC Config
```

### 12.2.3 List All LC Jobs

**EXAMPLE:**

```
winrm e
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]:wsman -encoding:utf-8 -a:basic
-SkipCNcheck -SkipCACheck
```

**OUTPUT:**

```
DCIM_LifecycleJob
```
InstanceID = JID_001272324322
JobStartTime
JobStatus = Completed
JobUntilTime
Message = Detach partition successful
MessageArguments = null
MessageID = VF038
Name = VFlashDetach:Partition1
DCIM_LifecycleJob
InstanceID = JID_001273099184
JobStartTime = 20191010101010 JobStatus = COMPLETED JobUntilTime = 2009:8:11
Message = The command was successful MessageArguments = null
MessageID = LC001
Name = LC Config

12.2.4 Get CSIOR Component Configuration Recovery (CCR) Attribute

The Component Configuration Recovery (CCR) attributes are:

- Licensed
- Part Firmware Update
- Collect System Inventory on Restart (CSIOR)
- Part Configuration Update

Get the current CSIOR attribute setting as follows:

**EXAMPLE 1:**

```sh
winrm g

cimv2/root/dcim/DCIM_LCEnumeration?InstanceID=LifecycleController.Embedded.1#LCAttributes.1#CollectSystemInventoryOnRestart
-u:[USERNAME] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman encoding:utf-8 -a:basic
```

**Note:** For 11G, InstanceID=DCIM_LCEnumeration:CCR5

**OUTPUT:**

```
DCIM_LCEnumeration

AttributeName = Collect System Inventory on Restart
CurrentValue = Disabled
DefaultValue = Enabled
ElementName = LC.emb.1
InstanceID = LifecycleController.Embedded.1#LCAttributes.1#CollectSystemInventoryOnRestart
IsReadOnly = false
PendingValue = null
PossibleValues = Enabled, Disabled
```
12.2.5 Get Part Firmware Update Attribute

Get the current Part Replacement firmware update mode as follows:

EXAMPLE:

```
winrm g
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCEnumeration?
InstanceID=LifecycleController.Embedded.1#LCAtributes.1#PartFirmwareUpdate
-u:[USERNAMe] -p:[PASSWORD]
-r:[IPADDRESS]/wsman
-encoding:u\tf-8 -a:basic
```

**Note:** For 11G, InstanceID=DCIM_LCEnumeration:CCR4

**OUTPUT:**

```
DCIM_LCEnumeration
AttributeName = Part Firmware Update
CurrentValue = Allow version upgrade only
DefaultValue = Disable
ElementName = LC.emb.1
InstanceID = LifecycleController.Embedded.1#LCAtributes.1#PartFirmwareUpdate
IsReadOnly = false
PendingValue = null
PossibleValues = Disable, Allow version upgrade only, Match firmware of replaced part
```

See Section 12.5 to get the status on whether there is a valid VFlash License on the System.

12.3 Re-Initiate Auto-Discovery Client

Invoke the ReInitiateDHS() method to re-initialize and restart the Auto-Discovery client. All configuration information is replaced with the auto discovery factory defaults. Auto discovery can be disabled, enabled and initiated immediately, or delayed until next power cycle.

Invoke ReInitiateDHS() with the following parameters and syntax:

**[PS_IP_ADDRESS]:** Substitution will need to be replaced with the actual IP address(s) or DNS name(s) of the Provisioning Server(s).

PerformAutoDiscovery:
1 = off (disables auto discovery)
2 = Now (enables and initiates auto discovery immediately )
3 = NextBoot (delay reconfiguration & auto discovery until next power cycle)

**EXAMPLE:**

```
winrm i ReInitiateDHS
cimv2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USERNAME] -p:[PASSWORD]
-r:[IPADDRESS]/wsman
-encoding:u\tf-8 -a:basic
-file:ReInitiateDHS.xml
```
The input file ReInitiateDHS.xml containing the parameters for the ReInitiateDHS method is shown below:

```xml
<p:ReInitiateDHS_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCSer-
vice">
  <p:ResetToFactoryDefaults>TRUE</p:ResetToFactoryDefaults>
  <p:PerformAutoDiscovery>3</p:PerformAutoDiscovery>
</p:ReInitiateDHS_INPUT>
```

**OUTPUT:**
The output is status 0 for successfully set or an error message.

ReInitiateDHS_OUTPUT

```
  ReturnValue = 0
```

### 12.4 Clear or Set Provisioning Server

The Provisioning Server name (or a group names) can be cleared by invoking the `ClearProvisioningServer()` method on the DCIM_LCService class. Configuring the Provisioning Server name(s)

**EXAMPLE-A:**

```plaintext
winrm i ClearProvisioningServer
cimv2/root/dcim/DCIM_LCService?
+SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USERNAME] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-encoding:utf-8 -a:basic
```

**OUTPUT-A:**

This method will return status 0 or error message.

ClearProvisioningServer_OUTPUT

```
  ReturnValue = 0
```

Setting the Provisioning Server name or IP address for the provisioning service

The Provisioning Server name and/or IP Addresses can be set by invoking the `SetAttribute()` method of the DCIM_LCService class.

**[PS_IP_ADDRESS]:** Substitution will need to be replaced with the actual IP address(s) or DNS name(s) of the Provisioning Server(s).

**EXAMPLE-B:**

```plaintext
winrm i SetAttribute
cimv2/root/dcim/DCIM_LCService?
+SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-encoding:utf-8 -a:basic
-file:SetProvisioningServer.xml
```
The input file SetProvisioningServer.xml is shown below:

```xml
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCSer
vice">
    <p:AttributeName>Provisioning Server</p:AttributeName>
</p:SetAttribute_INPUT>
```

**OUTPUT-B:**
This method will return status 0 or error message.

```xml
SetAttribute_OUTPUT
    RebootRequired = No
    ReturnValue = 0
    SetResult = Set CurrentValue
```

### 12.5 Check vFlash License Enablement

The following command can be used to check VFlash License enablement. Features such as Part Replacement, downloading ISO image to VFlash, or booting from VFlash are licensed features and require Dell VFlash SD Card to be inserted in order to function.

**EXAMPLE:**

```
winrm g
cimv2/root/dcim/DCIM_LCEnumeration?InstanceID=LifecycleController.Embedded.1#LCAttributes.1#Licensed
-u:[USER] -p:[PASSWORD]
-r:_https://[IPADDRESS]/wsman:443
-encoding:utf-8 -a:basic
```

**Note:** For 11G, InstanceID=DCIM_LCEnumeration:CCR1

**OUTPUT:**
This 'get' command will return the instance of the DCIM_LCEnumeration attribute class. The CurrentValue property will contain "True" (yes) or "False" (no) indicating whether features dependent on the presence of the VFlash SD card are enabled.

```xml
DCIM_LCEnumeration
    AttributeName = Licensed
    CurrentValue = Yes
    DefaultValue = No
    ElementName = LC.emb.1
    InstanceID = LifecycleController.Embedded.1#LCAttributes.1#Licensed
    IsReadOnly = true
    PendingValue
    PossibleValues = Yes, No
```
12.6 Download Server Public Key

This method is used to download the server public key to the Lifecycle Controller. A base64 encoded string containing the certificate authentication (CA) content is required as the input.

Invoke DownloadServerPublicKey() with the following parameters and syntax:

EXAMPLE:

```
winrm i DownloadServerPublicKey
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?CreationClassName=DCIM_LCService
+Name=DCIM:LCService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:DownloadServerPublicKey.xml
```

The input file DownloadServerPublicKey.xml is shown below:

```xml
<p:DownloadServerPublicKey_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
    <p:KeyContent>
        -----BEGIN CERTIFICATE-----
        MIIEQjCCAmgAwIBAgIBADANBgkqhkiG9w0BAQQQADCBzTELMAkGA1UEBhMCVVMx
        CzAJBgNVBAgTAlNBYMRQwEgYDVQQHEwtNYWluIFN0cmVldEVQMB0GA1UEChMMSm9l
        .
        .
        qvoMCKtoqLnGBByj/H2vyN7Fe/zMKXD5pO6XwYddGfA66w3HGUA0f+IKD40NDi9
        bKFEMxBxRzysUzuKZ9c+RAIUIrzemfX3fn1Yp7k05KU9vHY=
        -----END CERTIFICATE-----
    </p:KeyContent>
</p:DownloadServerPublicKey_INPUT>
```

OUTPUT:

When this method is executed, a jobid or an error message is returned. This jobid can then be used for subsequent processing with job control provider in Section 10.

```
DownloadServerPublicKey_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet
    Selector: InstanceID = JID_001269440883, __cimnamespace = root/dcim
ReturnValue = 0
```
12.7 Download Client Certificates

This method is used to download the client private certificate, password, and root certificate to Lifecycle Controller. A base64 encoded string containing the certificate authentication (CA) private key content is required as input.

Invoke DownloadClientCerts() with the following parameters and syntax:

EXAMPLE:

```
winrm i DownloadClientCerts
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService
+Name=DCIM_LCService
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
-u:[USER]  -p:[PASSWORD]
-r:[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:DownloadClientCerts.xml
```

The input file DownloadClientCerts.xml is shown below:

```
<p:DownloadClientCerts_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
<p:KeyContent>
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,5FD6D6131DFA5A86
ulG9hrQgOi0kUjKMBk95zi8H5KnZKnUnPq9kQ1Nco9WzKynINR1FbcIIA9ToUJOM
SnSs1a8fRbtjXZVBA+Ka+341v0/FEAjjSozKMW1nA+CMzCFM7t3P+3kmD+o6a

DfcWL1vaburBpaOj5HbVhLzcWezS1tuc1aIu09dCtT8/Uyr08KAVp5zu0b8bP
BGUqBbUqKsCPTKnNSNaDh+j0sQYB6B+9yZtaLPfdWkvob93oUUwj+CxTlxLGqe

-----END RSA PRIVATE KEY-----
</p:KeyContent>

<p:Password>[PASSWORD HERE]</p:Password>

<p:CAContent>
-----BEGIN CERTIFICATE-----
MIEzCCCA80gAwIBAgIBADANBgkqhkiG9w0BAQQFADBbTLEMAkGA1UEBhMCVVMx
CzAJSBdQVAgTAIRYMQwEGYDVQHEwNTYWhuIFN0cmVldDEVMbMGA1UEChMMRsS91

8o5kZK8xCaS9UQKDh5z6uUasj8DYk6pXnd4gWl5Wc9JfsN3+dratX3lrpoPJPhk
N1hT3XHy1DjLwSg7y1kIJPf5q25gdaeJ1jUYJBehRdQ+X7HxWN2Vuk+Z1NvYyZc=

-----END CERTIFICATE-----
</p:CAContent>

</p:DownloadClientCerts_INPUT>
When this method is executed, a jobid or an error message is returned. This jobid can then be used for subsequent processing with job control provider in Section 10.

```
DownloadClientCerts_OUTPUT
Job

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_LifecycleJob
SelectorSet

Selector: InstanceID = JID_001269440659, __cimnamespace = root/dcim

ReturnValue = 0
```

### 12.8 Delete Auto-Discovery Client Certificates

This method is used to delete the client certificates set previously by the auto discovery method.

Invoke `DeleteAutoDiscoveryClientCerts()` with the following parameters and syntax:

```
EXAMPLE:
winrm i DeleteAutoDiscoveryClientCerts
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:USERNAME -p:PASSWORD
-r:https://%IPADDRESS%/wsman
-encoding:utf-8 -a:basic
-SkipCACheck -SkipCNCheck -Skiprevocationcheck
```

```
OUTPUT:
DeleteAutoDiscoveryClientCerts_OUTPUT

ReturnValue = 0
```

### 12.9 Set Public Certificates

This method is used to update a public SSL Certificate on the iDRAC.

Invoke `SetPublicCertificate()` with the following parameters and syntax: **Type**: Specifies certificate service directory

```
EXAMPLE:
winrm i SetPublicCertificate
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://%IPADDRESS%/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetPublicCertificate.xml
```

```
OUTPUT:
SetPublicCertificate_OUTPUT

ReturnValue = 0
```
The input file SetPublicCertificate.xml is shown below:

```xml
<p:SetPublicCertificate_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
    <p:Type>directoryCA</p:Type>
    <p:Certificate>
        -----BEGIN CERTIFICATE-----
        MIID9DCCA12gAwIBAgIBADANBgkqhkiG9w0BAQQQFADCBbszELMAkGA1UEBhMCVVMx
        CzAJBgNVBAgTAlRYMQ8wDQYDVQQHEwZBdXN0aW4xDTALBgNVBAoTBERlbGwxFjAQ
        .
        H/ea71Ltbr/Au2QFhqcHkeUEbQ4qXSXTmDEgeKAImKjoCAaWHcDqEwvUcxGI4ekG
        LaUEGQhQ1cLe+03Rdp05j+YFo1v/N10OGMflhWg/1J3EcV12ba2tXnCp8XvCukJC
        R0ncFRP1p7c=        
        -----END CERTIFICATE-----
    </p:Certificate>
</p:SetPublicCertificate_INPUT>

OUTPUT:
SetPublicCertificate_OUTPUT
    ReturnValue = 0

12.10 Set iDRAC Certificate and Private Key

This method is used to update an iDRAC certificate and private key pairs using the contents of a PKCS#12 file.

Invoke SetCertificateAndPrivateKey() with the following parameters and syntax:

**Type**: Specifies the service the certificate is for: server = web server

**PKCS12**: Represents the base64 encoded contents of PKCS#12 file to upload. Note this is the contents of the file and not a filename.

**PKCS12pin**: Password to decode the PKCS12

**EXAMPLE:**

```
winrm i SetCertificateAndPrivateKey
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file: SetCertificateAndPrivateKey.xml
```

The input file SetCertificateAndPrivateKey.xml is shown below:

```xml
<p:SetCertificateAndPrivateKey_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
    <p:Type>server</p:Type>
    <p:PKCS12>
```
12.11 Delete Auto-Discovery Server Public Key

This method is used to delete the public server key set previously by the set auto discovery method. Invoke DeleteAutoDiscoveryServerPublicKey() with the following parameters and syntax:

EXAMPLE:

```
winrm i DeleteAutoDiscoveryServerPublicKey
```

cimv2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-encoding:ntf-8 -a:basic
-SkipCACheck
-SkipCNCheck

OUTPUT:
DeleteAutoDiscoveryServerPublicKey_OUTPUT

ReturnValue = 0

12.12 Insert Comment in Lifecycle Controller Log

This method is used to insert additional user comments into the Lifecycle Controller log. Invoke InsertCommentInLCLog() with the following parameters and syntax: Comment: Replace INSERT COMMENT HERE with desired comment in this location

EXAMPLE:

```
winrm i InsertCommentInLCLog
```

http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
The input file InsertCommentInLCLog.xml is shown below:

```
<p:InsertCommentInLCLog_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCSer
vice">
    <p:Comment>INSERT COMMENT HERE</p:Comment>
</p:InsertCommentInLCLog_INPUT>
```

OUTPUT:
InsertCommentInLCLog_OUTPUT
    ReturnValue = 0

12.13 Export Lifecycle Controller Log

This method is used to export the log from the Lifecycle Controller after processing jobs.
Invoke ExportLCLog() with the following parameters and syntax:

**IPAddress:** This is the IP address of the target export server.
**ShareName:** This is the directory path to the mount point.
**FileName:** This is the target output file.
**ShareType:** Type of share
    NFS=0, CIFS=2
**Username:** This is the username to the target export server.
**Password:** This is the password to the target export server.
**Workgroup:** This is the applicable workgroup.

**EXAMPLE:**
```
winrmi ExportLCLog
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCSer
vice?SystemCreationClassName=DCIM_ComputerSystem
    +CreationClassName=DCIM_LCService
    +SystemName=DCIM:Computer System
    +Name=DCIM:LCService -u:[USER] -p:[PASSWORD]
    -r:https://[IPADDRESS]/wsman
    -SkipCNcheck -SkipCACheck
    -encoding:utf-8 -a:basic -file:ExportLCLog.xml
```

The input file ExportLCLog.xml is shown below:

```
<p:ExportLCLog_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCSer
vice">
    <p:IPAddress>123.456.7.8</p:IPAddress>
    <p:ShareName>sharename</p:ShareName>
    <p:FileName>filename.txt</p:FileName>
    <p:ShareType>0</p:ShareType>
    <p:Username>admin</p:Username>
    <p>Password>password</p:Password>
    <p:Workgroup>workgroup</p:Workgroup>
</p:ExportLCLog_INPUT>
```
When this method is executed, a jobid or an error message is returned.

ExportLCLog_OUTPUT

Job

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet

Selector: InstanceID = JID_001271166022, __cimnamespace = root/dcim
ReturnValue = 0

12.14 Export Complete Lifecycle Log

The ExportCompleteLCLog() method is used to export the log from the Lifecycle Controller to a remote share.

Invoke ExportCompleteLCLog() with the following parameters and syntax:

IPAddress: This is the IP address of the target export server.
ShareName: This is the directory path to the mount point.
FileName: This is the target output file.
ShareType: Type of share. NFS=0, CIFS=2
Username: This is the username to the target export server.
Password: This is the password to the target export server.
Workgroup: This is the applicable workgroup.

EXAMPLE:


OUTPUT:

ExportCompleteLCLog_OUTPUT

Job

EndpointReference

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet

Selector: InstanceID = JID_671027850472, __cimnamespace = root/dcim
ReturnValue = 4096
Export Hardware Inventory from Lifecycle Controller

This method is used to export the hardware inventory from the Lifecycle Controller to a text file on a remote share.

Invoke ExportHWInventory() with the following parameters and syntax:

**IPAddress**: This is the IP address of the target export server.

**ShareName**: This is the directory path to the mount point.

**FileName**: This is the target output file.

**ShareType**: Type of share. NFS=0, CIFS=2

**Username**: This is the username to the target export server.

**Password**: This is the password to the target export server.

**Workgroup**: This is the applicable workgroup.

**EXAMPLE**:

```bash
winrm i ExportHWInventory
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic -file:ExportHWInventory.xml
```

The input file `ExportHWInventory.xml` is shown below:

```xml
<p:ExportHWInventory_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
    <p:IPAddress>123.456.7.8</p:IPAddress>
    <p:ShareName>sharename</p:ShareName>
    <p:FileName>filename.txt</p:FileName>
    <p:ShareType>0</p:ShareType>
    <p:Username>admin</p:Username>
    <p>Password>password</p>Password>
    <p:Workgroup>workgroup</p:Workgroup>
</p:ExportHWInventory_INPUT>
```

**OUTPUT**:

When this method is executed, a jobid or an error message is returned.

```xml
ExportHWInventory_OUTPUT
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    ReferenceParameters
    SelectorSet
        Selector: InstanceID = JID_001271167557, __cimnamespace = root/dcim
ReturnVALUE = 0
```
12.16 Export Factory Configuration

This method is used to export the factory configuration from the Lifecycle Controller to a text file on a remote share.

Invoke ExportFactoryConfiguration() with the following parameters and syntax:

**IPAddress**: This is the IP address of the target export server.

**ShareName**: This is the directory path to the mount point.

**FileName**: This is the target output file.

**ShareType**: Type of share. NFS=0, CIFS=2

**Username**: This is the username to the target export server.

**Password**: This is the password to the target export server.

**Workgroup**: This is the applicable workgroup.

EXAMPLE:

```
winrm i ExportFactoryConfiguration
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:Computer
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding=utf-8 -a:basic
-file:ExportFactoryConfiguration.xml
```

The input file ExportFactoryConfiguration.xml is shown below:

```
<p:ExportFactoryConfiguration_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
  
  <p:IPAddress>123.456.7.8</p:IPAddress>
  <p:ShareName>sharename</p:ShareName>
  <p:FileName>filename.txt</p:FileName>
  <p:ShareType>0</p:ShareType>
  <p:Username>admin</p:Username>
  <p:Password>password</p:Password>
  <p:Workgroup>workgroup</p:Workgroup>

</p:ExportFactoryConfiguration_INPUT>
```

OUTPUT:

When this method is executed, a jobid or an error message is returned.

ExportFactoryConfiguration_OUTPUT

```
Job
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
  ResourceURI =
  http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_LifecycleJob
  SelectorSet
    Selector: InstanceID = JID_001271168441, __cimnamespace = root/dcim
  ReturnValue = 0
```
12.17 System Decommission

This method is called to delete all configurations from the Lifecycle controller before the System is retired.

Invoke LCWipe() with the following parameters and syntax:

EXAMPLE:

```bash
winrm i LCWipe
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
LCWipe_OUTPUT
ReturnValume = 0
```

12.18 Get Remote Services API Status

The GetRemoteServicesAPIStatus() method is used to obtain the overall remote services API status that includes both the host System status as well as the remote services (Data Manager) status. The overall rolled up status shall be reflected in the Status output parameter.

**Note:** The LCStatus output parameter value includes the status reported by the DMStatus output parameter in the GetRSSStatus() method. Thus, GetRSSStatus() method invocation is redundant.

Invoke GetRemoteServicesAPIStatus() with the following parameters and syntax:

EXAMPLE:

```bash
winrm i GetRemoteServicesAPIStatus
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
GetRemoteServicesAPIStatus_OUTPUT
LCStatus = 0
Message = Lifecycle Controller Remote Services is ready.
MessageID = LC061
ReturnValume = 0
ServerStatus = 2
Status = 0
```
12.19 Export System Configuration
This method is used to export the System configuration from the Lifecycle Controller to a file on a remote share.
Invoke ExportSystemConfiguration() with the following parameters and syntax:

**IPAddress**: This is the IP address of the target export server.

**ShareName**: This is the directory path to the mount point.

**FileName**: This is the target output file.

**ShareType**: Type of share. NFS=0, CIFS=2

**Username**: This is the username to the target export server.

**Password**: This is the password to the target export server.

**EXAMPLE:**
```
winrm i ExportSystemConfiguration
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
@{IPAddress="SHARE_IP_ADDRESS";ShareName="SHARE_NAME";ShareType="SHARE_TYPE";FileName="SHARE_OUTPUT_FILE_NAME";Username="SHARE_USERNAME";Password="SHARE_PASSWORD"}
```

**OUTPUT:**
When this method is executed, a jobId or an error message is returned.

ExportSystemConfiguration_OUTPUT
```
Job
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_LifecycleJob
SelectorSet
Selector: InstanceID = JID_001271168441, __cimnamespace = root/dcim
ReturnValue = 0
```

12.20 Import System Configuration
This method is used to import the System configuration from the Lifecycle Controller from a file on a remote share.
Invoke ImportSystemConfiguration() with the following parameters and syntax:

**IPAddress**: This is the IP address of the target export server.

**ShareName**: This is the directory path to the mount point.

**FileName**: This is the target output file.

**ShareType**: Type of share. NFS=0, CIFS=2

**Username**: This is the username to the target export server.

**Password**: This is the password to the target export server.
EXAMPLE:

```
winrm i ImportSystemConfiguration
temCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
@{IPAddress="SHARE_IP_ADDRESS"; ShareName="SHARE_NAME";ShareType="SHARE_TYPE";
FileName="SHARE_OUTPUT_FILE_NAME";Username="SHARE_USERNAME"
Password="SHARE_PASSWORD"}
```

OUTPUT:
When this method is executed, a jobid or an error message is returned.

ImportSystemConfiguration_OUTPUT

```
Job
   Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
   ReferenceParameters
   SelectorSet
      Selector: InstanceID = JID_001271168441, __cimnamespace = root/dcim
   ReturnValue = 0
```

12.21 Import System Configuration Preview

This method is used to preview the XML template ahead of the actual application.

ImportSystemConfiguration() is implemented on DCIM_LCService class.

**IPAddress:** This is the IP address of the target export server.

**ShareName:** This is the directory path to the mount point.

**FileName:** This is the target input file.

**ShareType:** Type of share. **NFS=0, CIFS=2**

**Username:** This is the username to the target server.

**Password:** This is the password to the target server.

EXAMPLE:

```
winrm i ImportSystemConfigurationPreview
temCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://10.94.161.157/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic @{IPAddress="10.94.162.74";ShareName="Share name";
ShareType="2";Username="administrator";Password="dell_123"
FileName="SystemConf ig.xml"}
```
OUTPUT:
ImportSystemConfigurationPreview_OUTPUT
Job EndpointReference
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  SelectorSet
    Selector: InstanceID = JID_656038536587, __cimnamespace = root/dcim
  ReturnValue = 4096

To view the job
  DCIM_LifecycleJob
    ElapsedTimeSinceCompletion = 1 InstanceID = JID_656038536587
    JobstartTime = NA
    JobStatus = Completed
    JobUntilTime = NA
    Message = No changes occurred. Current component configuration matched the requested configuration. MessageArguments = NA
    MessageID = SYS069
    Name = Preview Configuration
    PercentComplete = 100

12.22 Remote Diagnostics
This feature will allow the user to remotely run hardware diagnostics through console application or remote scripts and collect results from tool execution

12.22.1 Run Diagnostics
  DCIM_LCService.RunPSADiagnostics: The method is used to run the diagnostics based on the runmode switch and save the report in the internal storage area. The diagnostics can be run in either express or extended mode or as a long run which encompasses all diagnostic tests.
  EXAMPLE:
  winrm i RunPSADiagnostics
  http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
  +SystemName=DCIM:ComputerSystem
  +Name=DCIM:LCService
  -u:%Username% -p:%Password%
  -r:https://%IPAddress%/wsman
  -SkipCNcheck -SkipCACheck
  -encoding:utf-8 -a:basic
  @{RunMode="1";RebootJobType="2";ScheduledStartTime="TIME_NOW"}
OUTPUT:
RunePSADiagnostics_OUTPUT
Job
EndpointReference
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters ResourceURI =
    http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_LifecycleJob
  SelectorSet
    Selector: InstanceID = JID_668777011296, __cimnamespace = root/dcim
  ReturnValue = 4096

12.22.2 Export Diagnostics Results
DCIM_LCService.ExportPSADiagnosticsResult: This method will export the result file of the last completed diagnostics into the respective remote share path (CIFS/NFS). The result file will have time stamps to show when the diagnostics was run.
EXAMPLE:
winrm i ExportPSADiagnosticsResult
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
@{IPAddress="%SharepathIPAddress%";ShareName="%ShareName%";ShareType="%ShareType%";Username="%UserName%";Password="%PassWord%";FileName="%FileName%"}

OUTPUT:
ExportPSADiagnosticsResult_OUTPUT
Job
  EndpointReference
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters ResourceURI =
    http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_LifecycleJob
  SelectorSet
    Selector: InstanceID = JID_6687771356675, __cimnamespace = root/dcim
  ReturnValue = 4096

12.22.3 Verify the Diagnostics Job Status:
EXAMPLE
winrm e
-u:%Username% -p:%Password%
-r:https://%IPAddress%/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
OUTPUT:

DCIM_LifeCycleJob
   ElapsedTimeSinceCompletion = null
   InstanceID = JID_660055291735
   JobStartTime = TIME_NOW
   JobStatus = Scheduled
   JobUntilTime = TIME_NA
   Message = Task successfully scheduled.
   MessageArguments = NA
   MessageID = JCP001
   Name = Remote Diagnostics
   PercentComplete = 0
13  VFlash SD Card Management

The Persistent Storage Profile describes the necessary properties and methods for representing and managing the partitions on the virtual flash media (SD Card on AMEA) provided by the iDRAC in Dell platforms.

The partition management of the virtual flash media includes:

- Listing virtual flash partitions
- Creating new partitions
- Deleting existing partitions
- Formatting a partition
- Exposing the partition in the host OS
- Detaching an attached partition
- Uploading an image to a partition
- Booting to a partition
- Modifying a partition
- Copying or exporting the contents of the partition

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

13.1  Listing the SD Card Partitions

Each partition on the virtual flash media shall be represented by an instance of DCIM_OpaqueManagementData. If nothing is returned, no partitions exist. Use the CreatePartition() method to create partitions.

Enumerate the DCIM_OpaqueManagementData with the following parameters and syntax:

EXAMPLE:

winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_OpaqueManagementData
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_OpaqueManagementData
  AccessType = Read Only
  AttachedState = Detach
  CreationClassName = DCIM_OpaqueManagementData
  DataFormat = RAW
  DeviceID = DCIM_OpaqueManagementData:Partition1
  ElementName = VFlash
  Name = label1
  PartitionIndex = 1
  PartitionType = HDD
  Size = 50
  SystemCreationClassName = DCIM_ComputerSystem
  SystemName = DCIM:ComputerSystem

Note: If nothing is returned, no partitions exist. Use the CreatePartition method to create partitions.
13.2 **Initialize the Virtual Flash Media**

- Enumerate the DCIM_PersistentStorageService class
- Invoke the InitializeMedia method on the instance above
- The OUT parameter Job will refer to the instance of CIM_ConcreteJob using which the user can query the status of the initialization of the media.

13.2.1 **Get vFlash SD Card Inventory**

DCIM_VFlashView is a subclass of CIM_View that is used to represent the physical attributes of the virtual flash media, such as total size, available size, category etc. on which the partitions will reside.

Enumerate the DCIM_VFlashView with the following parameters and syntax:

**EXAMPLE:**

```plaintext
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_VFlashView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
DCIM_VFlashView
  AvailableSize = 972
  Capacity = 972
  ComponentName = vFlash SD Card
  FQDD = Disk.vFlashCard.1
  HealthStatus = OK
  InitializedState = Uninitialized
  InstanceID = Disk.vFlashCard.1
  LastSystemInventoryTime = 20100426221347.000000+000
  LastUpdateTime = 20100426221347.000000+000
  Licensed = true
  VFlashEnabledState = true
  WriteProtected = false
```

- **InitializedState:** Field indicates status of element to be initialized
- **InstanceID:** InstanceID of desired element for initialization

See Section 13.2.3 for the populated initialized fields

13.2.2 **Initialize or Format Media**

This method is used to initialize or format the virtual flash media device.

```plaintext
winrm i InitializeMedia
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?SystemCreationClassName=DCIM_ComputerSystem +CreationClassName=DCIM_PersistentStorageService +SystemName=DCIM:ComputerSystem +Name=DCIM:PersistentStorageService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```
**OUTPUT:**
When this method is executed, a jobid or an error message is returned.

```plaintext
InitializeMedia
```

Job

```plaintext
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
ResourceURI = http://schemas.dell.com/wbem  SelectorSet
  Selector: InstanceID = JID_001268732835, __cimnamespace = root/dcim
ReturnValue = 0
```

**13.2.3 Verify Initialization or Formatting**

After invoking `InitializeMedia()`, get the instance of `DCIM_VFlashView` to confirm successful initialization.

Get a specific `DCIM_VFlashView` with the following parameters and syntax:

```plaintext
[INSTANCE_ID] = Obtained from Section 13.2.1, such as Disk.vFlashCard.1
```

**EXAMPLE:**

```plaintext
-u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck -encoding:utf-8 -a:basic
```

**OUTPUT:**

```plaintext
DCIM_VFlashView
  AvailableSize = 972
  Capacity = 972
  ComponentName = vFlash SD Card
  FQDD = Disk.vFlashCard.1
  HealthStatus = OK
  InitializedState = Initialized
  InstanceID = Disk.vFlashCard.1
  LastSystemInventoryTime = 20100426221347.000000+000
  LastUpdateTime = 20100426221347.000000+000
  Licensed = true
  VFlashEnabledState = true
  WriteProtected = false
```

**InitializedState:** Field indicates status of element to be initialized

**InstanceID:** InstanceID of desired element for initialization

---

**13.3 Enable or Disable vFlash using vFlash State Change**

This method is used to enable or disable the virtual flash media device. When the `VFlashStateChange()` method is successfully executed, the change will be dictated in the `VFlashEnabledState` parameter as shown in Section 13.2.1 and Section 13.2.3.

Invoke `VFlashStateChange()` with the following parameters and syntax:

**RequestedState:** The state to set to — Enable=1, Disable=2
EXAMPLE:

winrm i VFlashStateChange
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_PersistentStorageService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:PersistentStorageService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:VFlashStateChange.xml

The input file VFlashStateChange.xml is shown below:

```xml
<p:VFlashStateChange_INPUT
xmlns:p="http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_PersistentStorageService">
  <p:RequestedState>1</p:RequestedState>
</p:VFlashStateChange_INPUT>
```

OUTPUT:

VFlashStateChange_OUTPUT
  ReturnValue = 0

13.4 Create Partition

This method is used for creating a new partition on a storage device. When this method is successfully executed, an instance of DCIM_OpaqueManagementData representing the desired partition will be created (Section 13.1) and a reference to this instance is captured in the output parameter Job.

Invoke CreatePartition() with the following parameters and syntax:

**PartitionIndex:** The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be formatted. — 1 to 16

**Size:** The size of the partition to be created

**SizeUnit:** The unit of the size. — MB=1, GB=2

**PartitionType:** The partition type floppy=1, hard disk=2

**OSVolumeLabel:** The label seen in the OS after attaching the partition

EXAMPLE:

winrm i CreatePartition
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_PersistentStorageService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:PersistentStorageService
-r:https://[IPADDRESS]:443/wsman
-u:[USER] -p:[PASSWORD]
-auth:basic -encoding:utf-8
-SkipCNcheck
-SkipCACheck
-file:[DIRECTORYPATH]\CreatePartition.xml
The input file CreatePartition.xml is shown below:

```xml
<p:CreatePartition_INPUT
xmlns:p="http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_PersistentStorageService">
  <p:PartitionIndex>1</p:PartitionIndex>
  <p:Size>50</p:Size>
  <p:SizeUnit>1</p:SizeUnit>
  <p:PartitionType>2</p:PartitionType>
  <p:OSVolumeLabel>label1</p:OSVolumeLabel>
</p:CreatePartition_INPUT>

OUTPUT:

When this method is executed, a jobid or an error message is returned.

CreatePartition_OUTPUT

Job

  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
  SelectorSet
    Selector: InstanceID = JID_001270734913, __cimnamespace = root/dcim
  ReturnValue = 0

If this method returns the following message, the VFlash must be enabled using the VFlashStateChange() (Section 13.3) method.

CreatePartition_OUTPUT Message = VFlash not enabled

  MessageID = VF015
  ReturnValue = 2

13.5 Create Partition using Image

This method creates a partition on the storage device using the image provided by the user. The partition size will be the same as the size of the image. The maximum size of the image is 4GB.

The image can be located on a NFS/CIFS share or on a TFTP server. When this method is successfully executed, an instance of DCIM_OpaqueManagementData representing the desired partition will be created (Section 13.1), and a reference to this instance is captured in the output parameter Job.

Invoke CreatePartitionUsingImage() with the following parameters and syntax:

PartitionIndex: The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be formatted. — 1 to 16

PartitionType: The format types that these partitions need to be formatted as floppy=1, hard disk=2, CD ROM=3.

OSVolumeLabel: The label seen in the OS after attaching the partition.

URI: The URI location of firmware to update a component. Supported protocols are FTP and HTTP.

IPAddress: IP address of TFTP or NFS share

ShareType: Type of share — NFS=0, TFTP=1, CIFS=2, FTP=3, HTTP=4

SharePath: NFS sharepoint address

ImageName: Name of the ISO or IMG image

Workgroup: Name of the workgroup, if applicable.

Username: The username to be used to access the file.
**Password**: The password to be used to access the file.

**Port**: The port number to be used.

**HashType**: The hash type — MD5=1, SHA1=2

**HashValue**: The hash value string based on the HashType parameter

**EXAMPLE:**

```
winrm i CreatePartitionUsingImage
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_PersistentStorageService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:PersistentStorageService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:CreatePartitionUsingImage.xml
```

The input file `CreatePartitionUsingImage.xml` is shown below:

```
<p:CreatePartitionUsingImage_INPUT
xmlns:p="http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_PersistentStorageService">
  <p:PartitionIndex>1</p:PartitionIndex>
  <p:PartitionType>2</p:PartitionType>
  <p:OSVolumeLabel>label</p:OSVolumeLabel>
  <p:URI>ftp://123.456.7.89/dir/filename.exe</p:URI>
  <p:IPAddress>123.456.7.8</p:IPAddress>
  <p:ShareType>3</p:ShareType>
  <p:SharePath></p:SharePath>
  <p:ImageName>imagename.iso</p:ImageName>
  <p:Workgroup>workgroup</p:Workgroup>
  <p:Username>Administrator</p:Username>
  <p:Password>password</p:Password>
  <p:Port></p:Port>
  <p:HashType>1</p:HashType>
  <p:HashValue>123</p:HashValue>
</p:CreatePartitionUsingImage_INPUT>
```

**OUTPUT:**

When this method is executed, a jobid or an error message is returned.

```
CreatePartitionUsingImage_OUTPUT

Job
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
  SelectorSet
    Selector: InstanceID = JID_001268833219, __cimnamespace = root/dcim
  ReturnValue = 0
```
Reference Section 13.2 to fix an uninitialized media device error:

CreatePartitionUsingImage_OUTPUT Message = SD card not initialized
MessageID = VF017
ReturnValue = 2

13.6 Delete Partition

This method is for deleting a partition on a storage device. When this method is successfully executed, the instance of DCIM_OpaqueManagementData representing the desired partition along with the association instance of DCIM_ServiceAffectsElement will be deleted. The AvailableSize property of the associated storage media will increase by the size of the deleted partition.

Invoke DeletePartition() with the following parameters and syntax:

**PartitionIndex**: The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be removed. — 1 to 16

**EXAMPLE**:

```
winrm i DeletePartition
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?
+CreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:Computer
-r:https://[IPADDRESS]:443/wsman
-u:[USER] -p:[PASSWORD]
-auth:basic
-encoding:utf-8
-SkipCNcheck -SkipCACheck
file:[DIRECTORYPATH]\DeletePartition.xml
```

The input file DeletePartition.xml is shown below:

```xml
  <p:PartitionIndex>1</p:PartitionIndex>
</p:DeletePartition_INPUT>
```

**OUTPUT**:

When this method is executed, a ReturnValue or error message is returned.

DeletePartition_OUTPUT

```
ReturnValue = 0
```

An index that does not exist in the XML file may yield the following error message:

DeletePartition_OUTPUT

```
Message = Invalid partition index
MessageID = VF018
ReturnValue = 2
```
13.7 Format Partition

This method is for formatting a partition of the type specified by the user.

Use the following algorithm to successfully format an existing partition:

- Enumerate the DCIM_PersistentStorageService class
- Invoke the FormatPartition() method on the instance above with the following parameters:
  - **PartitionIndex**: The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be formatted. — 1 to 16
  - **FormatType**: The new format type of the partition — EXT2=1, EXT3=2, FAT16=3, FAT32=4

The OUT parameter Job will refer to the instance of CIM_ConcreteJob using which the user can query the status of the formatting of the partition.

**EXAMPLE:**

```bash
winrm i FormatPartition
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_PersistentStorageService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:PersistentStorageService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:FormatPartition.xml
```

The input file FormatPartition.xml is shown below:

```xml
    <p:PartitionIndex>13</p:PartitionIndex>
    <p:FormatType>4</p:FormatType>
</p:FormatPartition_INPUT>
```

**OUTPUT:**

When this method is executed, a jobid or an error message is returned.

```xml
FormatPartition_OUTPUT
Job
    Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    ReferenceParameters
    SelectorSet
        Selector: InstanceID = JID_001270738393, __cimnamespace = root/dcim
    ReturnValue = 0
```

13.8 Modify Partition

This method is used for modifying the changeable attributes of a partition.

Use the following algorithm to successfully modify an existing partition.

Enumerate the DCIM_PersistentStorageService class
Invoke ModifyPartition() method on the instance above with the following parameters:

**PartitionIndex**: The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be modified. — 1 to 16

**AccessType**: The type of access level — Read-Only=1, Read-Write=3

The OUT parameter Job will refer to the instance of CIM_ConcreteJob using which the user can query the status of the modification of the partition.

**EXAMPLE:**

```
winrm i ModifyPartition
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_PersistentStorageService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:PersistentStorageService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:ModifyPartition.xml
```

The input file ModifyPartition.xml is shown below:

```
    <p:PartitionIndex>6</p:PartitionIndex>
    <p:AccessType>3</p:AccessType>
</p:ModifyPartition_INPUT>
```

**OUTPUT:**

```
ModifyPartition_OUTPUT
    ReturnValue = 0
```

### 13.9 Attach Partition

This method is for defining the set of partitions to be exposed as Floppy/CD/HDD endpoints to the managed System and BIOS.

Invoke AttachPartition() with the following parameters and syntax:

**PartitionIndex**: The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be attached. — 1 to 16

**EXAMPLE:**

```
winrm i AttachPartition
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_PersistentStorageService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:PersistentStorageService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:AttachPartition.xml
```
The input file AttachPartition.xml is shown below:

```xml
    <p:PartitionIndex>12</p:PartitionIndex>
</p:AttachPartition_INPUT>
```

**OUTPUT:**

When this method is executed, a jobid or an error message is returned.

**AttachPartition_OUTPUT**

**Job**

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous

ReferenceParameters

ResourceURI =
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_LifecycleJob

SelectorSet

Selector: InstanceID = JID_001270737179, __cimnamespace = root/dcim

ReturnValue = 0

---

13.10 **Detach Partition**

This method is for defining the set of partitions to be removed as USB endpoints from the managed System.

Invoke DetachPartition() with the following parameters and syntax:

**PartitionIndex**: The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be detached — 1 to 16

**EXAMPLE:**

```
winrm i DetachPartition
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_PersistentStorageService
?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_PersistentStorageService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:PersistentStorageService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:DetachPartition.xml
```

The input file DetachPartition.xml is shown below:

```xml
    <p:PartitionIndex>12</p:PartitionIndex>
</p:DetachPartition_INPUT>
```
13.11 Export Data from Partition

This method is for exporting the contents of a partition to a location specified by the user.

Use the following algorithm to successfully export data from an existing partition.

Enumerate the DCIM_PersistentStorageService class

Invoke the ExportDataFromPartition() method on the instance above with the following parameters:

**PartitionIndex**: The PartitionIndex property of the DCIM_OpaqueManagementData instance that represents the partition to be formatted — 1 to 16

**IPAddress**: IP address of TFTP or NFS share

**ShareType**: Type of share — NFS=0, TFTP=1, CIFS=2

**SharePath**: NFS sharepoint address

**ImageName**: Name of the ISO or IMG image

**Workgroup**: Name of the workgroup, if applicable

**Username**: The username to be used to access the file

**Password**: The password to be used to access the file

**Port**: The port number to be used

**HashType**: The hash type — MD5=1, SHA1=2

**HashValue**: The hash value string based on the HashType parameter

**EXAMPLE:**

```
winrm i ExportDataFromPartition "
tentStorageService",SystemName="DCIM:ComputerSystem",SystemCreationClassName= "DCIM_ComputerSystem"
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:ExportDataFromPartition.xml"
```
The input file ExportDataFromPartition.xml is shown below:

```xml
  <p:PartitionIndex>1</p:PartitionIndex>
  <p:IPAddress>123.456.7.8</p:IPAddress>
  <p:ShareType>2</p:ShareType>
  <p:SharePath>/temp</p:SharePath>
  <p:ImageName>imagename.iso</p:ImageName>
  <p:Workgroup>workgroup</p:Workgroup>
  <p:Username>Administrator</p:Username>
  <p:Password>password</p:Password>
  <p:Port></p:Port>
  <p:HashType>1</p:HashType>
  <p:HashValue>123</p:HashValue>
</p:ExportDataFromPartition_INPUT>
```

**OUTPUT:**

When this method is executed, a jobid or an error message is returned.

ExportDataFromPartition_OUTPUT

```
Job
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
  SelectorSet
    Selector: InstanceID = JID_001271681930, __cimnamespace = root/dcim
  ReturnValue = 0
```
14 Boot Control Configuration Management

This feature provides the ability to get and set the boot order configuration. The Boot Control Profile describes the classes, associations, properties, and methods used to manage the boot control configurations of a physical or virtual computer system.

Profile and Associated MOFs:

http://www.delltechcenter.com/page/DCIM.Library.Profile

14.1 Listing the Boot Inventory-ConfigSetting Class

The boot configuration settings are a collection of settings that are applied to the boot configurable System during the boot process. The current, default, and next status fields of each element are available.

Enumerate BootConfigSetting with the following parameters and syntax:

EXAMPLE:


OUTPUT:

DCIM_BootConfigSetting
  ElementName = BootSeq
  InstanceID = IPL
  IsCurrent = 1
  IsDefault = 0
  IsNext = 1
DCIM_BootConfigSetting
  ElementName = HddSeq
  InstanceID = BCV
  IsCurrent = 2
  IsDefault = 0
  IsNext = 2
DCIM_BootConfigSetting
  ElementName = UefiBootSeq
  InstanceID = UEFI
  IsCurrent = 2
  IsDefault = 0
  IsNext = 2
DCIM_BootConfigSetting
  ElementName = OneTimeBootMode
  InstanceID = OneTime
  IsCurrent = 2
  IsDefault = 0
  IsNext = 2

This InstanceID can be used as input for a 'get' operation, as shown in Section 14.2
ElementName = vFlash Boot Configuration
InstanceID = vFlash
IsCurrent = 2
IsDefault = 0
IsNext = 2

14.2 Getting a Boot ConfigSetting Instance
Getting the boot configuration current, default, and next attributes of one particular boot configuration instance is an alternative to enumerating all available instances as shown in Section 14.1.

Get a BootConfigSetting instance with the following parameters and syntax:
[INSTANCEID]: This is obtained from the enumeration in Section 14.1, in which this example would use IPL as an instanceID

EXAMPLE:
winrm g
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BootConfigSetting
?InstanceID=[INSTANCEID]
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:
DCIM_BootConfigSetting
ElementName = BootSeq
InstanceID = IPL
IsCurrent = 1
IsDefault = 0
IsNext = 1

14.3 Listing the Boot Inventory-SourceSetting Class
Each Boot Configuration Representation contains an ordered list of boot sources, which indicate the logical devices to use during the boot process.

Enumerate the BootSourceSetting class with the following parameters and syntax:
EXAMPLE:
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BootSourceSetting
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:
DCIM_BootSourceSetting
BIOSBootString = Embedded SATA Port A Optical: SATA
Optical Drive BootSeq
BootString = Embedded SATA Port A Optical: SATA Optical
Drive BootSeq
CurrentAssignedSequence = 0

The ChangeBootOrderByInstanceID method in Section 14.4 will use the InstanceID field as input.
CurrentEnabledStatus = 1
ElementName = Embedded SATA Port A Optical: SATA Optical Drive BootSeq
FailThroughSupported = 1
InstanceID = IPL:Optical.SATAEmbedded.A-1:eb8aeb15796fb85f8e1447f0cfeb8a68e
PendingAssignedSequence = 0
PendingEnabledStatus = 1

DCIM_BootSourceSetting
  BIOSBootString = Hard drive C: BootSeq
  BootString = Hard drive C: BootSeq
  CurrentAssignedSequence = 1
  CurrentEnabledStatus = 1
  ElementName = Hard drive C: BootSeq
  FailThroughSupported = 1
  InstanceID = IPL:HardDisk.List.1-1:c9203080df84781e2ca3d512883dee6f
  PendingAssignedSequence = 1
  PendingEnabledStatus = 1

14.4 Changing the Boot Order by InstanceIDChangeBootOrderByInstanceID()

The ChangeBootOrderByInstanceID() method is called to change the boot order of boot sources within a configuration. The method’s input parameter, source, is an ordered array of InstanceIDs of BootSourceSetting instances.

The CurrentAssignedSequence attribute of each instance, from Section 14.3, defines the instance’s place in the zero based indexed boot sequence. Note: In order for the changes to be applied, the CreateTargetedConfigJob() method in Section 17.7 must be executed.

Invoke ChangeBootOrderByInstanceID() with the following parameters and syntax:

[INSTANCE ID]: Obtained from the BootSourceSetting Class enumeration, this example uses the field IPL source: Reference to the InstanceID attribute from Section 14.3

EXAMPLE:

```
winrm i ChangeBootOrderByInstanceID
-u: [USER] -p: [PASSWORD]
-r: https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:ChangeBootOrderByInstanceID.xml
```

The input file ChangeBootOrderByInstanceID.xml is shown below:

```xml
<p:ChangeBootOrderByInstanceID_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BootConfigSetting">
  <p:source>IPL:Optical.SATAEmbedded.A-1:eb8aeb15796fb85f8e1447f0cfeb8a68e</p:source>
  <p:source>UEFI:Disk.iDRACVirtual.1-4:1821</p:source>
</p:ChangeBootOrderByInstanceID_INPUT>
```
<p:ChangeBootOrderByInstanceID_INPUT>

OUTPUT:

ChangeBootOrderByInstanceID_OUTPUT Message = The command was successful
   MessageID = BOOT001
   ReturnValue = 0

14.5 Enable or Disable the Boot Source-ChangeBootSourceState()

The ChangeBootSourceState() method is called to change the enabled status of BootSourceSetting instances to Disable or Enable. The input parameter, source, is an array of InstanceID of BootSourceSetting instances. Enumerating the BootSourceSetting Class in Section 14.3, displays the CurrentEnabledStatus field which provides the applicable status.

**Note 1:** In order for the changes to be applied, the CreateTargetedConfigJob() method in Section 17.7 must be executed.

**Note 2:** BIOS does not support the setting of EnabledState for BCV devices.

Invoke ChangeBootSourceState() with the following parameters and syntax:

[INSTANCE ID]: Obtained from the BootSourceSetting Class enumeration, this example uses the field IPL source: Reference to the InstanceID attribute from Section 14.3

**EnabledState:** State of boot source element — Disabled=0, Enabled=1

**EXAMPLE:**

```
winrm i ChangeBootSourceState
```

The input file ChangeBootSourceState.xml is shown below:

```
   <p:EnabledState>0</p:EnabledState>
   <p:source>IPL:Optical.SATAEmbedded.A-1:eb8aeb15796fb85f8e1447f0cfb8a68e</p:source>
</p:ChangeBootSourceState_INPUT>
```

OUTPUT:

ChangeBootSourceState_OUTPUT Message = The command was successful
   MessageID = BOOT001
   ReturnValue = 0

The source input is obtained from the BootSourceSetting inventory in Section 14.3
15 NIC or CNA Card Management

This feature provides the ability to get and set the Network Interface (NIC) Card or Converged Network Adapter (CNA) attributes that are configurable using NIC/CNA Option-ROM or NIC/CNA UEFI HII. The attributes include functionalities for the following:

- Partition and personality (CNA only)
- iSCSI boot and PXE boot that are part of the NIC/CNA firmware

The ability to configure CNAs has been added to the NIC profile that extends the management capabilities of the referencing profiles. The NICs/CNAs are modeled as views with collections of attributes where there is a view for each partition on the controller.

The NIC or CNA Inventory has the following classes and views:

- DCIM_NICEnumeration (see Section 15.1)
- DCIM_NICString (see Section 15.2)
- DCIM_NICInteger (see Section 15.3)
- DCIM_NICView (see Section 15.4)
- DCIM_NICCapabilities (see Section 15.5)
- DCIM_NICStatistics (see Section 15.6)

Profile and Associated MOFS:

http://www.delltechcenter.com/page/DCIM.Library.Profile

15.1 Listing the NIC or CNA Inventory-Enumeration Class

Enumerate the NICEnumeration class with the following parameters and syntax:

**EXAMPLE – CNA:**

```
```

**OUTPUT – CNA:** For SAMPLE PORT 1 / PARTITION 1 (all attributes on all partitions are enumerated)

```
DCIM_NICEnumeration
    AttributeDisplayName = TCP/IP Parameters via DHCP
    AttributeName = TcpIpViaDHCP
    CurrentValue = Enabled
    Dependency = <Dep><AttrLev Op="OR"><ROIf Name="IpVer">IPv6</ROIf><ROIf Name="iSCSIBootSupport">Unavailable</ROIf></AttrLev></Dep>
    FQDD = NIC.Integrated.1-1-1
    GroupDisplayName = iSCSI General Parameters
    GroupID = IscsiGenParams
    InstanceID = NIC.Integrated.1-1-1:TcpIpViaDHCP
    IsReadOnly = false
    PendingValue = null
    PossibleValues = Disabled, Enabled
    PossibleValuesDescription = Disabled, Enabled
```
DCIM_NICEnumeration
   AttributeDisplayName = IP Autoconfiguration
   AttributeName = IpAutoConfig
   CurrentValue = Enabled
   Dependency = <Dep><AttrLev Op="OR"><ROIf Name="IpVer">IPv4</ROIf><ROIf Name="iSCSIBootSupport">Unavailable</ROIf></AttrLev></Dep>
   FQDD = NIC.Integrated.1-1-1
   GroupDisplayName = iSCSI General Parameters
   GroupID = IscsiGenParams
   InstanceID = NIC.Integrated.1-1-1:IpAutoConfig
   IsReadOnly = true
   PendingValue = null
   PossibleValues = Disabled, Enabled
   PossibleValuesDescription = Disabled, Enabled

DCIM_NICEnumeration
   AttributeDisplayName = iSCSI Parameters via DHCP
   AttributeName = IscsiViaDHCP
   CurrentValue = Enabled
   Dependency = <Dep><AttrLev Op="OR"><ROIf Name="iSCSIBootSupport">Unavailable</ROIf></AttrLev></Dep>
   FQDD = NIC.Integrated.1-1-1
   GroupDisplayName = iSCSI General Parameters
   GroupID = IscsiGenParams
   InstanceID = NIC.Integrated.1-1-1:IscsiViaDHCP
   IsReadOnly = false
   PendingValue = null
   PossibleValues = Disabled, Enabled
   PossibleValuesDescription = Disabled, Enabled

DCIM_NICEnumeration
   AttributeDisplayName = CHAP Authentication
   AttributeName = ChapAuthEnable
   CurrentValue = Disabled
   Dependency = <Dep><AttrLev Op="OR"><ROIf Name="iSCSIBootSupport">Unavailable</ROIf></AttrLev></Dep>
   FQDD = NIC.Integrated.1-1-1
   GroupDisplayName = iSCSI General Parameters
   GroupID = IscsiGenParams
   InstanceID = NIC.Integrated.1-1-1:ChapAuthEnable
   IsReadOnly = false
   PendingValue = null
   PossibleValues = Disabled, Enabled
   PossibleValuesDescription = Disabled, Enable
15.2 Listing the NIC or CNA Inventory-String Class

Enumerate DCIM_NICString class with the following parameters and syntax:

EXAMPLE:

```plaintext
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICString
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```plaintext
DCIM_NICString
  AttributeDisplayName = Chip Type
  AttributeName = ChipMdl
  CurrentValue = BCM5720 A0
  Dependency = null
  FQDD = NIC.Integrated.1-1-1
  GroupDisplayName = Broadcom Main Configuration Page
  GroupID = VndrConfigPage
  InstanceID = NIC.Integrated.1-1-1:ChipMdl
  IsReadOnly = true
  MaxLength = 0
  MinLength = 0
  PendingValue = null
  ValueExpression = null

DCIM_NICString
  AttributeDisplayName = PCI Device ID
  AttributeName = PCIDeviceID
  CurrentValue = 165F
  Dependency = null
  FQDD = NIC.Integrated.1-1-1
  GroupDisplayName = Broadcom Main Configuration Page
  GroupID = VndrConfigPage
  InstanceID = NIC.Integrated.1-1-1:PCIDeviceID
  IsReadOnly = true
  MaxLength = 0
  MinLength = 0
  PendingValue = null
  ValueExpression = null

DCIM_NICString
  AttributeDisplayName = Bus:Dev:Func
  AttributeName = BusDeviceFunction
  CurrentValue = 01:00:00
  Dependency = null
  FQDD = NIC.Integrated.1-1-1
```
GroupDisplayName = Broadcom Main Configuration Page GroupID = VndrConfigPage
InstanceID = NIC.Integrated.1-1-1:BusDeviceFunction
IsReadOnly = true
MaxLength = 0
MinLength = 0
PendingValue = null
ValueExpression = null
DCIM_NICString
  AttributeDisplayName = Link Status
  AttributeName = LinkStatus
  CurrentValue = UP
  Dependency = null
  FQDD = NIC.Integrated.1-1-1
GroupDisplayName = Broadcom Main Configuration Page
GroupID = VndrConfigPage
InstanceID = NIC.Integrated.1-1-1:LinkStatus
IsReadOnly = true
MaxLength = 0
MinLength = 0 PendingValue = null ValueExpression = null.

15.3 Listing the CNA Inventory-Integer Class

Enumerate the DCIM_NICInteger class with the following parameters and syntax:

**EXAMPLE:**

```
```

**OUTPUT:**

DCIM_NICInteger
  AttributeDisplayName = Blink LEDs
  AttributeName = BlnkLeds
  CurrentValue = 0
  Dependency = null
  FQDD = NIC.Integrated.1-1-1
GroupDisplayName = Broadcom Main Configuration Page
GroupID = VndrConfigPage
InstanceID = NIC.Integrated.1-1-1:BlnkLeds
IsReadOnly = false
LowerBound = 0
PendingValue = null
UpperBound = 15
DCIM_NICInteger
AttributeDisplayName = Link Up Delay Time
AttributeName = LnkUpDelayTime
CurrentValue = 0
Dependency = <Dep><AttrLev Op="OR"><ROIf Name="iSCSIBootSupport">Unavailable</ROIf></AttrLev></Dep>
FQDD = NIC.Integrated.1-1-1
GroupDisplayName = iSCSI General Parameters
GroupID = IscsiGenParams
InstanceID = NIC.Integrated.1-1-1:LnkUpDelayTime
IsReadOnly = false
LowerBound = 0
PendingValue = null
UpperBound = 255

DCIM_NICInteger
AttributeDisplayName = LUN Busy Retry Count
AttributeName = LunBusyRetryCnt
CurrentValue = 0
Dependency = <Dep><AttrLev Op="OR"><ROIf Name="iSCSIBootSupport">Unavailable</ROIf></AttrLev></Dep>
FQDD = NIC.Integrated.1-1-1
GroupDisplayName = iSCSI General Parameters
GroupID = IscsiGenParams
InstanceID = NIC.Integrated.1-1-1:LunBusyRetryCnt
IsReadOnly = false
LowerBound = 0
PendingValue = null UpperBound = 60

DCIM_NICInteger
AttributeDisplayName = TCP Port AttributeName = FirstTgtTcpPort
CurrentValue = 3260
Dependency = <Dep><AttrLev Op="OR"><ROIf Name="iSCSIBootSupport">Unavailable</ROIf></AttrLev></Dep>
FQDD = NIC.Integrated.1-1-1
GroupDisplayName = iSCSI First Target Parameters
GroupID = IscsiFirstTgtParams
InstanceID = NIC.Integrated.1-1-1:FirstTgtTcpPort
IsReadOnly = false
LowerBound = 1
PendingValue = null
UpperBound = 65535
15.4 Listing the CNA Inventory-NICView Class

Enumerate the DCIM_NICView class with the following parameters and syntax:

EXAMPLE:

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICView
-u:[USER] -p:[PASSWORD]
-r:[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT FOR FIRST and SECOND PORT (NICView will return all ports and partitions):

```
DCIM_NICView
  AutoNegotiation = 0
  BusNumber = 1
  ControllerBIOSVersion = 1.17 CurrentMACAddress = 14:FE:B5:FF:B3:EA
  DataBusWidth = 0002
  DeviceNumber = 0
  EFIVersion = 15.0.16
  FCoEOffloadMode = 3
  FCoEWWNN = null
  FQDD = NIC.Integrated.1-1-1
  FamilyVersion = 7.0.39
  FunctionNumber = 0
  InstanceID = NIC.Integrated.1-1-1
  LastSystemInventoryTime = 20010708151620.000000+000
  LastUpdateTime = 20010708151606.000000+000
  LinkDuplex = 0
  LinkSpeed = 0
  MaxBandwidth = 0 MediaType = 4
  MinBandwidth = 0
  NicMode = 3
  PCIDeviceID = 165f
  PCISubDeviceID = 1f5b
  PCISubVendorID = 1028
  PClVendorID = 14e4
  PermanentFCOEMACAddress
  PermanentMACAddress = 14:FE:B5:FF:B3:EA
  PermanentiSCSIMACAddress
  ProductName = Broadcom Gigabit Ethernet BCM5720 - 14:FE:B5:FF:B3:EA
  ReceiveFlowControl = 0
  SlotLength = 0002 SlotType = 0002 TransmitFlowControl = 0
  VendorName = null WWPN = null iScsiOffloadMode = 3
```

```
DCIM_NICView
  AutoNegotiation = 0
  BusNumber = 1
  ControllerBIOSVersion = 1.17 CurrentMACAddress = 14:FE:B5:FF:B3:EB
```
15.5 Listing the CNA Inventory-NICCapabilities Class

Enumerate the DCIM_NICCapabilities class with the following parameters and syntax:

**EXAMPLE:**

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICCapabilities
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -tSkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
DCIM_NICCapabilities
  BPESupport = 3
  CongestionNotification = 3
```
DCBExchangeProtocol = 3
ETS = 3
EVBModesSupport = 3
EnergyEfficientEthernet = 2
FCoEBootSupport = 3
FCoEMaxIOsPerSession = 0
FCoEMaxNPIVPerPort = 0
FCoEMaxNumberExchanges = 0
FCoEMaxNumberLogins = 0
FCoEMaxNumberOfFCTargets = 0
FCoEMaxNumberOfOutStandingCommands = 0
FCoEOffloadSupport = 3
FQDD = NIC.Integrated.1-1-1
FeatureLicensingSupport = 3
FlexAddressingSupport = 2
IPSecOffloadSupport = 3
InstanceID = NIC.Integrated.1-1-1
MACSecSupport = 3
NWManagementPassThrough = 2
NicPartitioningSupport = 3
OSBMCManagementPassThrough = 2
OnChipThermalSensor = 2
OpenFlowSupport = 3
PXEBootSupport = 2
PartitionWOLSUPPORT =
PriorityFlowControl = 3
RDMASupport = 3
RXFlowControl = 3
RemotePHY = 3
TCPChimneySupport = 3
TXBandwidthControlMaximum = 3
TXBandwidthControlMinimum = 3
TXFlowControl = 3
VEBVEPAMultiChannel = 3
VEBVEPASingleChannel = 3
VFSRIOVSupport = 3
VirtualLinkControl = 3
WOLSUPPORT = 2
iSCSIBootSupport = 2
iSCSIOffloadSupport = 3
UEFISupport = 2
### 15.6 Listing the CNA Inventory - NICStatistics Class

Enumerate the DCIM_NICStatistics class with the following parameters and syntax:

**EXAMPLE:**

```
```

**OUTPUT:**

```
DCIM_NICStatistics
  DiscardedPkts = 0
  FCCRCErrorCount = null
  FCOELinkFailures = null
  FCOEPktRxCount = null
  FCOEPktTxCount = null
  FCOERxFktDroppedCount = null
  FQDD = NIC.Integrated.1-1-1
  InstanceID = NIC.Integrated.1-1-1
  LinkStatus = 1
  OSDriverState = 1
  PartitionLinkStatus = null
  PartitionOSDriverState = null
  RxBroadcast = 65177
  RxBytes = null
  RxErrorPktAlignmentErrors = 0
  RxErrorPktFCSErrors = 0
  RxFalseCarrierDetection = null
  RxJabberPkt = null
  RxMulticast = 11000
  RxPauseXOFFFrames = 0
  RxPauseXONFrames = 0
  RxRuntPkt = null
  RxUnicast = 0
  StartStatisticTime = 20111208013952.000000+000
  statisticTime = 20111208073904.000000+000
  TxBroadcast = 0
  TxBytes = null
  TxErrorPktExcessiveCollision = null
  TxErrorPktLateCollision = null
  TxErrorPktMultipleCollision = null
  TxErrorPktSingleCollision = null
  TxMulticast = 74
  TxPauseXOFFFrames = 0
  TxPauseXONFrames = 0
```
15.7 Applying the Pending Values for CNA-CreateTargetedConfigJob()

The CreateTargetedConfigJob() method is called to apply the pending values created using the SetAttribute() and SetAttributes() methods. The System automatically reboots depending on the ScheduledStartTime selected. Use the CreateTargetedConfigJob() jobID output to get the status (see Section 10.0).

Invoke CreateTargetedConfigJob() with the following parameters and syntax:

**Target:** This parameter is the FQDD, which is found by enumerating the CNA attributes in Section 15.1.

**RebootJobType:** There are three options for rebooting the System.

1. PowerCycle
2. Graceful Reboot without forced shutdown
3. Graceful reboot with forced shutdown

**Note:** When a user does not want to set a reboot type while creating a target job, users should comment out the RebootJobType in the input xml. User should not enter “0” or give no parameter in the input xml.

**ScheduledStartTime & UntilTime:** See Section 3.2.4

**EXAMPLE:**

```
winrm i CreateTargetedConfigJob
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_NICService
+SystemName=DCIM:ComputerSystem
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:CreateTargetedConfigJob_CNA.xml
```

The input file CreateTargetedConfigJob_CNA.xml is shown below:

```
<p:CreateTargetedConfigJob_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_NICSe
rvice">
  <p:Target>NIC.Integrated.1-1-1</p:Target>
  <p:RebootJobType>1</p:RebootJobType>
  <p:ScheduledStartTime>TIME_NOW</p:ScheduledStartTime>
  <p:UntilTime>20201111111111</p:UntilTime>
</p:CreateTargetedConfigJob_INPUT>
```

**OUTPUT:**

When this method is executed, a jobid or an error message is returned. The status of this jobid can be checked within the job control provider in Section 10.

```
CreateTargetedConfigJob_OUTPUT
Job
  Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
  SelectorSet
```
15.8 Deleting the Pending Values for CNADeletePendingConfiguration()

The DeletePendingConfiguration() method cancels the pending configuration changes made before the configuration job is created using the CreateTargetedConfigJob() method. This method only operates on the pending changes before running the CreateTargetedConfigJob() method. After the configuration job is created, to cancel the pending changes, call the DeleteJobQueue() method in the Job Control profile.

Invoke the DeletePendingConfiguration() method with the following parameters and syntax:

**EXAMPLE:**

```bash
winrm i DeletePendingConfiguration
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_NICService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:NICService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:DeletePendingConfiguration_CNA.xml
```

The input file DeletePendingConfiguration_CNA.xml is shown below:

```xml
<p:DeletePendingConfiguration_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_NICService">
  <p:Target>NIC.Integrated.1-1-1</p:Target>
</p:DeletePendingConfiguration_INPUT>
```

**OUTPUT:**

DeletePendingConfiguration_OUTPUT Message = The command was successful
MessageID = NIC001
ReturnValue = 0

15.9 Getting the CNA Enumeration Instance

Use the following example to get an instance of the DCIM_NICEnumeration class.

Get a DCIM_NICEnumeration class instance from the first port and first partition with the following parameters and syntax:

**[INSTANCEID]:** This is obtained from the enumeration in Section 15.1, in which this example would use NIC.Integrated.1-1-1: as an InstanceID.

**EXAMPLE:**

```bash
winrm g
-r:https://[IPADDRESS]:443/wsman
-u:[USER] -p:[PASSWORD]
-auth:basic -encoding:utf-8
-SkipCNcheck -SkipCACheck
```
15.10 Setting the IscsiOffloadMode Attribute

The SetAttribute() method is used to set or change the value of a CNA attribute. Enable the NICMode, IscsiOffloadMode, and FcoeOffloadMode personality attributes to enable the corresponding personalities: NIC, iSCSI, and FCoE.

For Broadcom CNA cards, the partitions on each port can be set to any personality. NICMode can always be enabled or disabled for any of the given partitions. For the IscsiOffloadMode and FcoeOffloadMode personalities, up to two personalities can be enabled on each port.

For the Qlogic CNA cards, partition three can be set to either NICMode or IscsiOffloadMode. Partition four can be set to either NICMode or FcoeOffloadMode.

Invoke the SetAttribute() method with the following parameters (from Section 15.1) and syntax:

**Target:** FQDD attained through DCIM_NICEnumeration

**AttributeName:** Attained from AttributeName field

**AttributeValue:** A new value to assign to the specified NICAtribute. If this value is valid, it is applied to the PendingValue property or the Currentvalue property of the specified NICAtribute. Possible choices are attained from PossibleValues field, such as:

**Possible values:** Disabled, Enabled

**EXAMPLE:**

```sh
```
The information in the input file SetAttribute_NIC.xml is shown below:

```xml
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_NICService">
    <p:Target>NIC.Integrated.1-1-1</p:Target>
    <p:AttributeName>IscsiOffloadMode</p:AttributeName>
    <p:AttributeValue>Enabled</p:AttributeValue>
</p:SetAttributes_INPUT>

OUTPUT:
SetAttribute_OUTPUT
Message = The command was successful
MessageID = NIC001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set PendingValue
```

### 15.11 Setting the MaxBandwidth Attribute

The `setAttribute()` method is used to set or change the value of a CNA attribute. The `MinBandwidth` and `MaxBandwidth` attributes control the bandwidth allocations for a given CNA partition. The values are displayed in percentage.

For Broadcom CNA cards, the `MinBandwidth` attribute values for a given port must always add up to either 0 or 100. `MaxBandwidth` is a value of 100 or less for any given partition.

For the Qlogic CNA cards, the `MinBandwidth` attribute values for a given port must add up to 100 or less. `MaxBandwidth` again is a value of 100 or less for any given partition.

Invoke `setAttribute()` with the following parameters (from Section 15.1) and syntax:

- **Target**: FQDD attained through DCIM_NICInteger
- **AttributeName**: Attained from AttributeName field
- **AttributeValue**: A new value to assign to the specified NICAttribute. If this value is valid, it is applied to the `PendingValue` property or the `Currentvalue` property of the specified NICAttribute. Range of choices is attained from the `LowerBound` and `UpperBound` fields:
  - `LowerBound = 0`
  - `UpperBound = 100`

**EXAMPLE:**

```bash
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICS\ervice?\ystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_NICS\ervice
+SystemName=DCIM:ComputerSystem
+Name=DCIM:NICS\ervice
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttribute_NIC.xml
```
The input file SetAttribute_NIC.xml is shown below:

```xml
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_NICService">
  <p:Target>NIC.Integrated.1-1-2</p:Target>
  <p:AttributeName>MaxBandwidth</p:AttributeName>
  <p:AttributeValue>75</p:AttributeValue>
</p:SetAttributes_INPUT>

OUTPUT:
SetAttribute_OUTPUT
Message = The command was successful
MessageID = NIC001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set PendingValue
```

15.12 Setting the VirtMacAddr Attribute

The SetAttribute() method is used to set or change the value of a CNA attribute. The I/O identity string attributes: (VirtMacAddr, VirtLscsiMacAddr, VirtFIPMacAddr, VirtWWN, and VirtWWPN) display a unique behavior. After setting them to a non-default value, the attribute values are retained until there is AC power supply. If the AC power supply is disconnected, the attributes revert to their default values.

Invoke the SetAttribute() method with the following parameters and syntax:

- **Target**: FQDD attained through DCIM_NICString
- **AttributeName**: Attained from AttributeName field
- **AttributeValue**: A new value to assign to the specified NICAttribute. If this value is valid, it is applied to the PendingValue property or the CurrentValue property of the specified NICAttribute. The range of acceptable strings is present in the MinLength and MaxLength fields.

**EXAMPLE:**

```
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_NICService
+SystemName=DCIM:Computer
+Name=DCIM:NICService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttribute_NIC.xml
```

The input file SetAttribute_NIC.xml is shown below:

```xml
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_NICService">
  <p:Target>NIC.Integrated.1-1-2</p:Target>
  <p:AttributeName>VirtMacAddr</p:AttributeName>
</p:SetAttributes_INPUT>
```
15.13 Setting the LegacyBootProto Attribute

The SetAttribute() method is used to set or change the value of a NIC attribute.

**WARNING:** The local BIOS setting always overwrites the LegacyBootProto option. This option is only applied in the BIOS setup. By setting this attribute remotely, it appears that the value is set, but it really did not because the local BIOS setting overrides it. Running a ‘get’ on the attribute remotely displays a different current value.

Invoke SetAttribute() with the following parameters(from Section 15.1) and syntax:

**Target:** FQDD attained through DCIM_NICEnumeration

**AttributeName:** Attained from AttributeName field

**AttributeValue:** A new value to assign to the specified NICAttribute. If this value is valid, it will be applied to the PendingValue property or the CurrentValue property of the specified NICAttribute. Possible choices are attained from PossibleValues field, such as:

**Possible values:** PXE, iSCSI, NONE

**EXAMPLE:**

```
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICService?SystemCreationClassName=DCIM_ComputerSystem +CreationClassName=DCIM_NICService +SystemName=DCIM:ComputerSystem +Name=DCIM:NICService
-u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman -SkipCNcheck -SkipCACheck -encoding:utf-8 -a:basic -file:SetAttribute_NIC.xml
```

The input file SetAttribute_NIC.xml is shown below:

```
    <p:Target>NIC.Embedded.1-1</p:Target>
    <p:AttributeName>LegacyBootProto</p:AttributeName>
    <p:AttributeValue>PXE</p:AttributeValue>
</p:SetAttributes_INPUT>
```

**OUTPUT:**

```xml
SetAttribute_OUTPUT
   Message = The command was successful
   MessageID = NIC001
   RebootRequired = Yes
   ReturnValue = 0
   SetResult = Set PendingValue
```
### 15.14 Setting CNA LAN Modes

The `SetAttributes()` method is used to set or change the values of a group of NIC attributes.

Invoke `SetAttributes()` with the following parameters (from Section 15.1) and syntax:

- **Target**: FQDD attained through `DCIM_NICEnumeration`
- **AttributeName**: Attained from `AttributeName` field
- **AttributeValue**: A new value to assign to the specified `NICAttribute`. If this value is valid, it will be applied to the `PendingValue` property or the `CurrentValue` property of the specified `NICAttribute`. Possible choices are attained from `PossibleValues` field.

**EXAMPLE:**

```
winrm i SetAttributes
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_NICService
+SystemName=DCIM:Computer
+Name=DCIM:NICService
-u:[USER] -p:[PASSWORD]
-r:[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttributes_NIC.xml
```

The input file `SetAttributes_NIC.xml` is shown below:

```xml
<p:SetAttributes_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_NICService">
  <p:Target>NIC.Embedded.1-1</p:Target>
  <p:AttributeName>LegacyBootProto</p:AttributeName>
  <p:AttributeValue>PXE</p:AttributeValue>
  <p:AttributeName>LnkSpeed</p:AttributeName>
  <p:AttributeValue>10Mbps Half</p:AttributeValue>
  <p:AttributeName>WakeOnLan</p:AttributeName>
  <p:AttributeValue>Disabled</p:AttributeValue>
  <p:AttributeName>VLANMode</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
  <p:AttributeName>IscsiTgtBoot</p:AttributeName>
  <p:AttributeValue>Disabled One Time</p:AttributeValue>
</p:SetAttributes_INPUT>
```

**OUTPUT:**

```
SetAttributes_OUTPUT
Message = The command was successful
MessageID = NIC001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set PendingValue
```
15.15 Setting the iSCSI Boot Target

The SetAttributes() method is used to set or change the values of the iSCSI boot target attributes.

Invoke the SetAttributes() method with the following parameters (from 15.1) and syntax:

**Target**: FQDD attained through DCIM_NICEnumeration

**AttributeName**: Attained from AttributeName field

**AttributeValue**: A new value to assign to the specified NICAttribute. If this value is valid, it is applied to the PendingValue property or the CurrentValue property of the specified NICAttribute. Possible choices are attained from PossibleValues field, such as:

**Possible values**: Disabled, Enabled

**EXAMPLE:**

```plaintext
winrm i SetAttributes
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_NICService
+SystemName=DCIM:Computer
+Name=DCIM:NICService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttributes_iSCSI_BootTarget.xml
```

The information in the input file SetAttributes_iSCSI_BootTarget.xml is shown below:

```xml
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_NICService">
  <p:Target>NIC.Integrated.1-1-1</p:Target>
  <p:AttributeName>BootToTarget</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorIpAddr</p:AttributeName>
  <p:AttributeValue>10.10.10.10</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorSubnet</p:AttributeName>
  <p:AttributeValue>255.255.255.0</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorGateway</p:AttributeName>
  <p:AttributeValue>10.10.10.1</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorPrimDns</p:AttributeName>
  <p:AttributeValue>10.10.10.2</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorSecDns</p:AttributeName>
  <p:AttributeValue>10.10.10.3</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorName</p:AttributeName>
  <p:AttributeValue>testname</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorChapId</p:AttributeName>
  <p:AttributeValue>testid</p:AttributeValue>
  <p:AttributeName>IscsiInitiatorChapPwd</p:AttributeName>
  <p:AttributeValue>testpassword</p:AttributeValue>
  <p:AttributeName>FirstTgtIpAddress</p:AttributeName>
  <p:AttributeValue>2.2.2.2</p:AttributeValue>
  <p:AttributeName>FirstTgtIscsiName</p:AttributeName>
</p:SetAttributes_INPUT>
```
15.16 Setting the FCoE Boot Target

The SetAttributes() method is used to set or change the values of the FCoE boot target attributes. Invoke the SetAttributes() method with the following parameters (from 15.1) and syntax:

**Target**: FQDD attained through DCIM_NICEnumeration

**AttributeName**: Attained from AttributeName field

**AttributeValue**: A new value to assign to the specified NICAttribute. If this value is valid, it is applied to the PendingValue property or the Currentvalue property of the specified NICAttribute. Possible choices are attained from PossibleValues field, such as:

**Possible values**: Disabled, Enabled

**EXAMPLE**:

```
winrm i SetAttributes
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_NICService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_NICService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:NICService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttribute_FCoE_BootTarget.xml
```

The information in the input file SetAttributes_FCoE_BootTarget.xml is shown below:

```
<p:Target>NIC.Integrated.1-1-1</p:Target>
<p:AttributeName>ConnectFirstFCoETarget</p:AttributeName>
<p:AttributeValue>Enabled</p:AttributeValue>
<p:AttributeName>FirstFCoEWWPTarget</p:AttributeName>
<p:AttributeValue>20:00:00:10:18:88:C0:03</p:AttributeValue>
<p:AttributeName>FirstFCoEBootTargetLUN</p:AttributeName>
<p:AttributeValue>33</p:AttributeValue>
<p:AttributeName>FirstFCoEFcfVLANID</p:AttributeName>
```
<p:AttributeValue>34</p:AttributeValue>
</p:SetAttributes_INPUT>

OUTPUT:

SetAttributes_OUTPUT

Message = The command was successful
MessageID = NIC001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set PendingValue
16 RAID Storage Management

The remote RAID configuration allows users to remotely query and configure the Hardware RAID of the System. The RAID profile extends the management capabilities of referencing profiles by adding the capability to represent the configuration of RAID storage. The RAID storage is modeled as collections of attributes where there are collections for the storage adaptors, physical disks, logical disks, end enclosures and parent-child relationships between the collections. Additionally, there is a configuration service that contains all the methods used to configure the RAID storage.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

The RAID Inventory contains the following attributes:
- DCIM_RAIDEnumeration (16.1)
- DCIM_RAIDInteger (16.3)
- DCIM_RAIDString (16.5)
- DCIM_ControllerView (16.7)
- DCIM_PhysicalDiskView (16.9)
- DCIM_VirtualDiskView (16.10)
- DCIM_EnclosureView (16.11)

16.1 Listing the RAID Inventory-Enumeration Class

The RAID Inventory has these attributes: DCIM_RAIDEnumeration (this section), DCIM_RAIDInteger (Section 16.3), and DCIM_RAIDString (see Section 16.5).

Enumerate the DCIM_RAIDEnumeration class to display all the RAID controllers and virtual disk attributes in a System.

Enumerate the DCIM_RAIDEnumeration class with the following parameters and syntax:

EXAMPLE:
winrme http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDEnumeration
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT
DCIM_RAIDEnumeration
  AttributeName = RAIDSupportedDiskProt
  CurrentValue = SAS, SATA
  FQDD = RAID.Integrated.1-1
  InstanceID = RAID.Integrated.1-1:RAIDSupportedDiskProt
  IsReadOnly = true
  PendingValue
  PossibleValues = SAS, SATA
DCIM_RAIDEnumeration
  AttributeName = RAIDloadBalancedMode
  CurrentValue = Automatic
  FQDD = RAID.Integrated.1-1
  InstanceID = RAID.Integrated.1-1:RAIDloadBalancedMode
  IsReadOnly = false

The 'get' instance method in section 16.2 uses this InstanceID as input.
The 'set attribute' method in section 16.19.1 uses the FQDD, AttributeName, and PossibleValues fields as input.
The 'set attributes' method in section and PossibleValues fields as input.
16.2 Getting a RAID Enumeration Instance

Use the following example to get an instance of the DCIM_RAIDEnumeration class instead of all the instances as shown in Section 16.1.

Get a RAIDEnumeration instance with the following parameters and syntax:

```
[INSTANCEID]: This is obtained from the enumeration in Section 16.1, which shows an example using RAID.Integrated.1-1:RAIDloadBalancedMode as an instanceID.
```

**EXAMPLE:**

```
```

**OUTPUT:**

```
DCIM_RAIDEnumeration
  AttributeName = RAIDloadBalancedMode
  CurrentValue = Automatic
  FQDD = RAID.Integrated.1-1
  InstanceID = RAID.Integrated.1-1:RAIDloadBalancedMode
  IsReadOnly = false
  PendingValue
  PossibleValues = Automatic, Disabled
```

16.3 Listing the RAID Inventory-Integer Class

The RAID Inventory has these attributes: DCIM_RAIDEnumeration (see Section 16.1), DCIM_RAIDInteger (this section), and DCIM_RAIDString (see Section 16.5).

Enumerate the DCIM_RAIDInteger class to display all the RAID controller attributes in a System.
Enumerate RAIDInteger with the following parameters and syntax:

**EXAMPLE:**

```
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDInteger
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT**

DCIM_RAIDInteger

- **AttributeName** = RAIDmaxPDsInSpan
- **CurrentValue** = 32
- **FQDD** = RAID.Integrated.1-1
- **InstanceId** = RAID.Integrated.1-1:RAIDmaxPDsInSpan
- **IsReadOnly** = true
- **LowerBound** = 0
- **PendingValue**
- **UpperBound** = 0

The 'get' instance method in Section 16.4 used this InstanceID as input.

DCIM_RAIDInteger

- **AttributeName** = RAIDmaxSpansInVD
- **CurrentValue** = 8
- **FQDD** = RAID.Integrated.1-1
- **InstanceId** = RAID.Integrated.1-1:RAIDmaxSpansInVD
- **IsReadOnly** = true
- **LowerBound** = 0
- **PendingValue**
- **UpperBound** = 0

The 'set attribute' method in Section 16.19.3 uses the FQDD, AttributeName, and a value equal to or between the LowerBound and UpperBound fields as input.

DCIM_RAIDInteger

- **AttributeName** = RAIDrebuildRate
- **CurrentValue** = 30
- **FQDD** = RAID.Integrated.1-1
- **InstanceId** = RAID.Integrated.1-1:RAIDrebuildRate
- **IsReadOnly** = false
- **LowerBound** = 1
- **PendingValue**
- **UpperBound** = 100

The 'set attributes' method in section 16.19.4 uses the FQDD, AttributeName, and a value equal to or between the LowerBound and UpperBound fields as input.
DCIM_RAIDInteger
AttributeName = RAIDreconstructRate
CurrentValue = 30
FQDD = RAID.Integrated.1-1
InstanceID = RAID.Integrated.1-1:RAIDreconstructRate
IsReadOnly = false
LowerBound = 1
PendingValue
UpperBound = 100

16.4 Getting a RAID Integer Instance
Use the following example to get an instance of the DCIM_RAIDInteger class, instead of all instances as shown in Section 16.3.

Get a RAIDInteger instance with the following parameters and syntax:

[INSTANCEID]: This is obtained from the enumeration in Section 16.3, which shows an example using RAID.Integrated.1-1:RAIDrebuildRate as an instanceID

EXAMPLE:
winrm g cimv2/root/dcim/DCIM_RAIDInteger?InstanceID=[INSTANCE ID]
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:
DCIM_RAIDInteger
AttributeName = RAIDreconstructRate
CurrentValue = 30
FQDD = RAID.Integrated.1-1
InstanceID = RAID.Integrated.1-1:RAIDreconstructRate
IsReadOnly = false
LowerBound = 1
PendingValue
UpperBound = 100

16.5 Listing the RAID Inventory-String Class
The RAID Inventory has these attributes: DCIM_RAIDEnumeration (see Section 16.1), DCIM_RAIDInteger (see Section 16.3), and DCIM_RAIDString (this section).

Enumerate the DCIM_RAIDString class to display all the RAID controller string attributes in a System.

Enumerate RAIDString with the following parameters and syntax:

EXAMPLE:
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim schema/2/root/dcim/DCIM_RAIDString
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
**OUTPUT:**

DCIM_RAIDString

- **AttributeName = Name**
- **CurrentValue = MyCacheCadeVD**
- **FQDD = Disk.Virtual.0:RAID.Integrated.1-1**
- **InstanceID = Disk.Virtual.0:RAID.Integrated.1-1:Name**
- **IsReadOnly = true**
- **MaxLength = 15**
- **MinLength = 0**
- **PendingValue**

DCIM_RAIDString

- **AttributeName = Name**
- **CurrentValue = raid 1 vd**
- **FQDD = Disk.Virtual.1:RAID.Integrated.1-1**
- **InstanceID = Disk.Virtual.1:RAID.Integrated.1-1:Name**
- **IsReadOnly = true**
- **MaxLength = 15**
- **MinLength = 0**
- **PendingValue**

16.6 Getting a RAID String Instance

Use the following example to get an instance of the DCIM_RAIDString class instead of all instances as shown in Section 16.5.

Get a DCIM_RAIDString instance with the following parameters and syntax:

```
[INSTANCEID]: This is obtained from the enumeration in Section 16.5, which shows an example using Disk.Virtual.0:RAID.Integrated.1-1:Name as an instanceID
```

**EXAMPLE:**

```
winrm g cimv2/root/dcim/DCIM_RAIDString?InstanceID=[INSTANCE ID]
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

DCIM_RAIDString

- **AttributeName = Name**
- **CurrentValue = MyCacheCadeVD**
- **FQDD = Disk.Virtual.0:RAID.Integrated.1-1**
- **InstanceID = Disk.Virtual.0:RAID.Integrated.1-1:Name**
- **IsReadOnly = true**
- **MaxLength = 15**
- **MinLength = 0**
- **PendingValue**
16.7 Listing the RAID Inventory-ControllerView Class

The DCIM_ControllerView class groups together a set of Controller properties.

Enumerate ControllerView with the following parameters and syntax:

EXAMPLE:

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_ControllerView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
DCIM_ControllerView
  Bus = 1
  CacheSizeInMB = 0
  CachecadeCapability = 0
  ControllerFirmwareVersion = 20.10.1-0066
  Device = 0
  DeviceCardDataBusWidth = 1
  DeviceCardManufacturer = DELL
  DeviceCardSlotLength = 4
  DeviceCardSlotType = PCI Express x8
  DriverVersion = null
  EncryptionCapability = 0
  EncryptionMode = 0
  FQDD = RAID.Slot.1-1
  Function = 0
  InstanceID = RAID.Slot.1-1
  KeyID = null
  LastSystemInventoryTime = 20120116145459.000000+000
  LastUpdateTime = 20120116145459.000000+000
  PCIDeviceID = 73
  PCISlot = 1
  PCISubDeviceID = 1F4E
  PCISubVendorID = 1028
  PCIVendorID = 1000
  PatrolReadState = 1
  PrimaryStatus = 1
  ProductName = PERC H310 Adapter
  RollupStatus = 1
  SASAddress = 5782BCB00C577600
  SecurityStatus = 0
  SlicedVDCapability = 1
```
16.8 Getting a RAID ControllerView Instance

The `get()` command can be invoked using a particular `instanceID`, attained from listing the inventory.

Get a RAID ControllerView instance with the following parameters and syntax:

**[INSTANCEID]**: This is obtained from the enumeration in Section 16.7, in which this example would use RAID.Slot.1-1 as an instanceID

**EXAMPLE:**

```
winrm g http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_ControllerView
```

**OUTPUT:**

```
DCIM_ControllerView
  Bus = 1
  CacheSizeInMB = 0
  CacheCacheCapability = 0
  ControllerFirmwareVersion = 20.10.1-0066
  Device = 0
  DeviceCardDataBusWidth = 1
  DeviceCardManufacturer = DELL
  DeviceCardSlotLength = 4
  DeviceCardSlotType = PCI Express x8
  DriverVersion = null
  EncryptionCapability = 0
  EncryptionMode = 0
  FQDD = RAID.Slot.1-1
  Function = 0
  InstanceID = RAID.Slot.1-1
  KeyID = null
  LastSystemInventoryTime = 20120116145459.000000+000
  LastUpdateTime = 20120116145459.000000+000
  PCIDeviceID = 73
  PCISlot = 1
  PCISubDeviceID = 1F4E
  PCISubVendorID = 1028
  PCIVendorID = 1000
  PatrolReadState = 1
  PrimaryStatus = 1
  ProductName = PERC H310 Adapter
  RollupStatus = 1
  SASAddress = 5782BCB00C577600
  SecurityStatus = 0
```
SlicedVDCapability = 1

16.9 Listing the RAID Inventory-PhysicalDiskView Class

Enumerating the PhysicalDiskView, results in the attributes and inventory of the available physical disks in the System.

Enumerate PhysicalDiskView with the following parameters and syntax:

**EXAMPLE:**

```plaintext
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_PhysicalDiskView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```plaintext
DCIM_PhysicalDiskView
  BusProtocol = 6
  Connector = 0
  DriveFormFactor = 3
  FQDD = Disk.Bay.0:Enclosure.Internal.0-0:RAID.Slot.1-1
  FreeSizeInBytes = 8978432
  HotSpareStatus = 0
  InstanceID = Disk.Bay.0:Enclosure.Internal.0-0:RAID.Slot.1-1
  LastSystemInventoryTime = 20120116145459.000000+000
  LastUpdateTime = 20120116145459.000000+000
  Manufacturer = SEAGATE
  ManufacturingDay = 7
  ManufacturingWeek = 50
  ManufacturingYear = 2010
  MaxCapableSpeed = 3
  MediaType = 0
  Model = ST9500430SS
  OperationName = None
  OperationPercentComplete = 0
  PPID = TH0R734K212330CG0027A00
  PredictiveFailureState = 0
  PrimaryStatus = 1
  RaidStatus = 2
  Revision = DS62
  RollupStatus = 1
  SASAddress = 5000C50025D64875
  SecurityState = 0
  SerialNumber = 9SP297S1
  SizeInBytes = 499558383616
  Slot = 0
  SupportedEncryptionTypes = None
```
16.10 Listing the RAID VirtualDiskView Inventory

Enumerating the VirtualDiskView, results in the attributes and inventory of the available virtual disks in the System.

Enumerate VirtualDiskView with the following parameters and syntax:

EXAMPLE:

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_VirtualDiskView
-u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman
```
SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:
DCIM_VirtualDiskView
  BusProtocol = 6
  cachecade = 0
  DiskCachePolicy = 1024
  FQDD = Disk.Virtual.0:RAID.Slot.1-1
  InstanceID = Disk.Virtual.0:RAID.Slot.1-1
  LastSystemInventoryTime = 20120116145459.000000+000
  LastUpdateTime = 20120116145459.000000+000
  LockStatus = 0
  MediaType = 1
  Name = Virtual Disk 00
  ObjectStatus = 0
  OperationName = None
  OperationPercentComplete = 0
  PhysicalDiskIDs = Disk.Bay.0:Enclosure.Internal.0-0:RAID.Slot.1-1,
                   Disk.Bay.1:Enclosure.Internal 0-0:RAID.Slot.1-1,
                   Disk.Bay.2:Enclosure.Internal.0-0:RAID.Slot.1-1
  PrimaryStatus = 1
  RAIDStatus = 2
  RAIDTypes = 2
  ReadCachePolicy = 16
  RemainingRedundancy = 0
  RollupStatus = 1
  SizeInBytes = 107481464832
  SpanDepth = 1
  SpanLength = 3
  StartingLBAinBlocks = 0
  StripeSize = 128
  VirtualDiskTargetID = 0
  WriteCachePolicy = 1

After successful virtual disk creation:
DCIM_VirtualDiskView
  BusProtocol = 6
  cachecade = 0
  DiskCachePolicy = 1024
  FQDD = Disk.Virtual.0:RAID.Slot.1-1
  InstanceID = Disk.Virtual.0:RAID.Slot.1-1
  LastSystemInventoryTime = 20120116145459.000000+000
  LastUpdateTime = 20120116145459.000000+000
  LockStatus = 0
  MediaType = 1
Name = Virtual Disk 00
ObjectStatus = 3
OperationName = None
OperationPercentComplete = 0
PhysicalDiskIDs = Disk.Bay.0:Enclosure.Internal.0-0:RAID.Slot.1-1,
Disk.Bay.1:Enclosure.Internal 0-0:RAID.Slot.1-1,
Disk.Bay.2:Enclosure.Internal.0-0:RAID.Slot.1-1
PrimaryStatus = 1
RAIDStatus = 2
RAIDTypes = 2
ReadCachePolicy = 16
RemainingRedundancy = 0
RollupStatus = 1
SizeInBytes = 107481464832
SpanDepth = 1
SpanLength = 3
StartingLBAinBlocks = 0
StripeSize = 128
VirtualDiskTargetID = 0
WriteCachePolicy = 1

16.11 Listing the RAID EnclosureView Inventory

Enumerating the EnclosureView, results in the attributes and inventory of the available enclosure components in the System.

Enumerate EnclosureView with the following parameters and syntax:

EXAMPLE:

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_EnclosureView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
DCIM_EnclosureView
   AssetTag
   Connector = 0
   EMMCount = 0
   FQDD = Enclosure.Internal.0-0:RAID.Integrated.1-1
   FanCount = 0
   InstanceID = Enclosure.Internal.0-0:RAID.Integrated.1-1
   LastSystemInventoryTime = 20100413194610
   LastUpdateTime = 20100413193143
   PSUCount = 0
   PrimaryStatus = 0
   ProductName = BACKPLANE 0:0
```
16.12 Listing the Controller Batteries
Enumerate the DCIM_ControllerBatteryView to list the controller batteries in the System.

16.13 Listing the Enclosure EMM

16.14 Listing the Enclosure PSU

16.15 Reset Configuration-ResetConfig()

The ResetConfig() method is used to delete all virtual disks and unassign all HotSpare physical disks. The deletions will not occur until a configuration job (Section 16.15) is scheduled and the System is rebooted. All data on the existing virtual disks will be lost!

Invoke ResetConfig with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)

**EXAMPLE:**

```plaintext
winrm i ResetConfig
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:ResetConfig.xml
```

The input file ResetConfig.xml is shown below:

```xml
<p:ResetConfig_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-l</p:Target>
</p:ResetConfig_INPUT>
```

**OUTPUT:**
ResetConfig_OUTPUT
ReturnValue = 0

16.16 Clearing the Foreign Configuration-ClearForeignConfig()

The ClearForeignConfig() method is used to prepare any foreign physical disks for inclusion in the local configuration.

Invoke ClearForeignConfig() with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)
EXAMPLE:

```
winrm i ClearForeignConfig
cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:ClearForeignConfig.xml
```

The input file ClearForeignConfig.xml is shown below:

```xml
<p:ClearForeignConfig_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
<p:Target>RAID.Integrated.1-1</p:Target>
</p:ClearForeignConfig_INPUT>
```

OUTPUT:

```
ClearForeignConfig_OUTPUT
ReturnValue = 0
```

If no foreign physical disks are available, the following message may result:

```
ClearForeignConfig_OUTPUT
Message = General failure
MessageID = STOR006
ReturnValue = 2
```

16.17 Applying the Pending Values for RAID-
CreateTargetedConfigJob()

The CreateTargetedConfigJob() method is called to apply the pending values created by RAID methods. The System will automatically reboot depending on the ScheduledStartTime selected. The CreateTargetedConfigJob() jobID output with the job control section can be used to obtain its status. Invoke CreateTargetedConfigJob() with the following parameters and syntax:

- **TARGET**: This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)
- **RebootJobType**: There are three options for rebooting the System.
  1. PowerCycle
  2. Graceful Reboot without forced shutdown
  3. Graceful reboot with forced shutdown

**Note**: When a user does not want to set a reboot type when creating a target job, users should comment out the RebootJobType in the input xml. User should not enter “0” or give no parameter at all in the input xml.

**ScheduledStartTime & UntilTime**: See Section 3.2.4

EXAMPLE:

```
winrm i CreateTargetedConfigJob
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
```
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:CreateTargetedConfigJob_RAID.xml

The input file CreateTargetedConfigJob_RAID.xml is shown below:

```xml
<p:CreateTargetedConfigJob_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:RebootJobType>3</p:RebootJobType>
  <p:ScheduledStartTime>TIME_NOW</p:ScheduledStartTime>
  <p:UntilTime>20111111111111</p:UntilTime>
</p:CreateTargetedConfigJob_INPUT>
```

OUTPUT:

When this method is executed, a jobid or an error message is returned. The status of this jobid can be checked within the job control provider in Section 10.

CreateTargetedConfigJob_OUTPUT

Job

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet
  Selector: InstanceID = JID_001271251761, __cimnamespace = root/dcim
  ReturnValue = 4096

16.18 Deleting the Pending Values for RAIDDeletePendingConfiguration()

The DeletePendingConfiguration() method cancels the pending configuration changes made before the configuration job is created with CreateTargetedConfigJob(). This method only operates on the pending changes prior to CreateTargetedConfigJob() being called. After the configuration job is created, the pending changes can only be canceled by calling DeleteJobQueue() in the Job Control profile.

Invoke DeletePendingConfiguration() with the following parameters and syntax: **TARGET**: This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)

**EXAMPLE:**

```
winrm i DeletePendingConfiguration
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService -u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:DeletePendingConfiguration.xml
```

The input file DeletePendingConfiguration.xml is shown below:
16.19 Managing Hot Spare

16.19.1 Determining Potential Disks-GetDHSDisks()

The GetDHSDisks() method is used to determine possible choices of drives to be a dedicated HotSpare for the identified virtual disk.

Invoke GetDHSDisks() with the following parameters and syntax:

**TARGET**: This parameter is the FQDD of the target virtual disk. Its value will depend on the number of virtual disks, obtainable in Section 16.10.

**EXAMPLE**:

```plaintext
winrm i GetDHSDisks
```

The input file GetDHSDisks.xml is shown below:

```xml
<p:GetDHSDisks_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>DISK.Virtual.1:RAID.Integrated.1-1</p:Target>
</p:GetDHSDisks_INPUT>
```

**OUTPUT**:

GetDHSDisks_OUTPUT
Return Value = 0

The following message may be fixed by deleting the job queue as referenced in Section 10.2.2.

GetDHSDisks_OUTPUT
Message = Configuration already committed, cannot set configuration
MessageID = STOR023
Return Value = 2

16.19.2 Assigning the Hot Spare-AssignSpare()

The AssignSpare() method is used to assign a physical disk as a dedicated HotSpare for a virtual disk (VD), or as a global HotSpare.
Invoke AssignSpare() with the following parameters and syntax:

**TARGET**: This parameter is the FQDD of the DCIM_PhysicalDiskView (Section 16.9)

**VirtualDiskArray**: Array of ElementName(s) where each identifies a different VD, currently only one VD can be passed

**EXAMPLE:**

```bash
winrm i AssignSpare
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:Computer System
+Name=DCIM:RAIDService
-u:[USER]  -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:AssignSpare.xml
```

The input file AssignSpare.xml is shown below:

```xml
<p:AssignSpare_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
<p:VirtualDiskArray>Disk.Virtual.0:RAID.Integrated.1-1</p:VirtualDiskArray>
</p:AssignSpare_INPUT>
```

**OUTPUT:**

AssignSpare_OUTPUT

RebootRequired = YES
ReturnValue = 0

Nonconformance to the following restrictions may result in the error message below.

- Virtual disk (VD) referenced (dedicated hot spare) is RAID-0, which cannot have hot spares
- Physical disk drive(PDD) is too small for the virtual disk referenced (dedicated hot spare)
- Physical disk drive is wrong type for the virtual disk (i.e. SATA PD to be used as hot spare for SAS VD)

Similar conditions when no VD referenced, which is the global hot spare attempted assignment

**ERROR MESSAGE:**

AssignSpare_OUTPUT

Message = Physical disk FQDD did not identify a valid physical disk for the operation MessageID = STOR009
ReturnValue = 2

### 16.19.3 Unassigning the Hot Spare-UnassignSpare()

The UnassignSpare() method is used to unassign a physical disk. The physical disk may be used as a dedicated hot spare to a virtual disk, or as a global hot spare. After the method executes successfully the physical disk is no longer a hotspare.

Invoke UnassignSpare() with the following parameters and syntax:**TARGET**: This parameter is the FQDD of the DCIM_PhysicalDiskView
EXAMPLE:

winrm i UnassignSpare
cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem +CreationClassName=DCIM_RAIDService +SystemName=DCIM:ComputerSystem +Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:http://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:UnassignSpare.xml

The input file UnassignSpare.xml is shown below:

```xml
</p:UnassignSpare_INPUT>
```

OUTPUT:

UnassignSpare_OUTPUT

RebootRequired = YES
ReturnValue = 0

16.20 Managing Keys for Self Encrypting Drives

**Note:** The Dell Key Manager feature is not available at this time.

16.20.1 Setting the Key-SetControllerKey()

The SetControllerKey() method sets the key on controllers that support encryption of the virtual disk drives.

Invoke SetControllerKey() with the following parameters and syntax:

- **TARGET**: This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)
- **Key**: Maximum size 32 characters
- **Keyid**: Identifier, or description, for the key (maximum size 255 characters)

**EXAMPLE:**

winrm i SetControllerKey
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem +CreationClassName=DCIM_RAIDService +SystemName=DCIM:ComputerSystem +Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:http://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetControllerKey.xml

The input file SetControllerKey.xml is shown below:

```xml
<p:SetControllerKey_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
```

Dell Lifecycle Controller Web Services Interface Guide for Windows
16.20.2 Locking the Virtual Disk - LockVirtualDisk()

The LockVirtualDisk() method encrypts the virtual disk. Note that the virtual disk must first exist for this method to be successful.

Invoke LockVirtualDisk() with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the target virtual disk

**EXAMPLE:**
```
winrm i LockVirtualDisk
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService?
SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCAcheck
-encoding:utf-8 -a:basic
-file:LockVirtualDisk.xml
```

The input file LockVirtualDisk.xml is shown below:

```xml
  <p:Target>Disk.Virtual.0:RAID.Integrated.1-1</p:Target>
</p:LockVirtualDisk_INPUT>
```

**OUTPUT:**

This method requires an H700 or H800 controller to properly function, as does the LockVirtualDisk() method. If the key is not set by LockVirtualDisk(), the following message may be displayed:

```
LockVirtualDisk_OUTPUT
Message = Controller Key is not present
MessageID = STOR021
ReturnValue = 2
```

16.20.3 Locking the Controller with a Key - EnableControllerEncryption()

The EnableControllerEncryption() method is used to set either Local Key encryption or Dell Key Manager (DKM) encryption on controllers that support encryption of the drives.

Invoke EnableControllerEncryption() method with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the DCIM_ControllerView class. See Section 16.1.
**Key**: Key – Passcode. This parameter is required if the Mode = Local Key Encryption. The Key can be maximum 32 characters in length, and must have one character from each of the following sets.

- Upper Case
- Lower Case Number
- Special Character

The special characters in the following set needs to be passed as mentioned below.

- & - &amp;
- < - &lt;
- > - &gt;
- " - &quot;
- ' - &apos;

**Keyid**: Key Identifier - Describes Key. The Keyid can be maximum 32 characters in length and must not have spaces in it.

**Mode**: Mode of the Controller

1 — Local Key Encryption
2 — Dell Key Manager

**EXAMPLE:**

```
winrm i EnableControllerEncryption
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:EnableControllerEncryption.xml
```

The information in the input file EnableControllerEncryption.xml is shown below:

```
<p:EnableControllerEncryption_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:Mode>1</p:Mode>
  <p:Key>Abcd@123</p:Key>
  <p:Keyid>LKM</p:Keyid>
</p:EnableControllerEncryption_INPUT>
```

**OUTPUT:**

This method requires an PERC controller with Local Key encryption or DKM support to function correctly.

```
EnableControllerEncryption_OUTPUT
  RebootRequired = YES
  ReturnValue = 0
```

### 16.20.4 Rekeying the Controller-ReKey()

The ReKey() method is used to reset the key on the controller that supports encryption. This method switches the controller mode between Local Key encryption or Dell Key Manager (DKM) encryption.

Invoke the ReKey() method with the following parameters and syntax:

**TARGET**: This parameter is the FQDD of the DCIM_ControllerView class. See section 16.1.
**OldKey**: Old controller key

**NewKey**: New controller key. The Key can be maximum 32 characters long, and must have one character from each of the following:

- Upper Case
- Lower Case Number
- Special Character

The special characters in the following set must be passed as mentioned below.

- & - &amp;
- < - &lt;
- > - &gt;
- " - &quot;
- ' - &apos;

**Keyid**: Key Identifier — Describes Key. The Keyid can be maximum 32 characters long and should not have spaces in it.

**Mode**: Mode of the Controller

1 — Local Key Encryption
2 — Dell Key Manager

**EXAMPLE:**

```plaintext
winrm i ReKey
cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem+CreationClassName=DCIM_RAIDService +SystemName=DCIM:ComputerSystem +Name=DCIM:RAIDService -u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman -SkipCNcheck -SkipCACheck -encoding:utf-8 -a:basic -file:ReKey.xml
```

The information in the input file ReKey.xml is shown below:

```xml
<p:ReKey_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
    <p:Target>RAID.Integrated.1-1</p:Target>
    <p:OldKey>Abcd@123</p:OldKey>
    <p:NewKey>Efgh@123</p:NewKey>
    <p:Keyid>NewLKMid</p:Keyid>
    <p:Mode>1</p:Mode>
</p:ReKey_INPUT>
```

**OUTPUT:**

This method requires a PERC controller with Local Key encryption or DKM support to function correctly. If the EnableControllerEncryption() method does not set the key, the following message is displayed:

```plaintext
ReKey_OUTPUT
    Message = Controller Key is not present
    MessageID = STOR021
    ReturnValue = 2
```
16.20.5 Removing the Key - RemoveControllerKey()

The RemoveControllerKey() method is used to erase the key on the controller along with the attached encrypted drives.

Invoke the RemoveControllerKey() method with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the DCIM_ControllerView class. See section 16.1.

**EXAMPLE:**

```
winrm i RemoveControllerKey
```

cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem+CreationName=DCIM_RAIDService +SystemName=DCIM:ComputerSystem +Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding: utf-8 -a:basic

The input file RemoveControllerKey.xml is shown below:

```xml
<p:RemoveControllerKey_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
    <p:Target>RAID.Integrated.1-1</p:Target>
</p:RemoveControllerKey_INPUT>
```

**OUTPUT:**

This method requires an H700 or H800 controller to function correctly. If the EnableControllerEncryption() method does not set the key, the following message is displayed:

```
RemoveControllerKey_OUTPUT
    Message = Controller Key is not present
    MessageID = STOR021
    ReturnValue = 2
```

16.21 Managing Virtual Disk

16.21.1 Getting the Available RAID levels - GetRAIDLevels()

The GetRAIDLevels() method is used to determine possible choices RAID levels to create virtual disks. If the list of physical disks is not provided, this method will operate on all connected disks.

Invoke GetRAIDLevels() with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)

**DiskType:** Corresponds to MediaType attribute in PhysicalDiskView (Section 16.9)

- Include all types=0, Include Magnetic Only=1, Include SSD only=2

**DiskProtocol:** Types of protocol to include

- Include all protocols=0, Include SATA=1, Include SAS=2

**DiskEncrypt:** Types of encryption to include

- 0 = Include FDE capable and non encryption capable disks
- 1 = Include FDE disks only
- 2 = Include only non FDE disks

**PDArray:** This parameter is the list of physical disk FQDDs
EXAMPLE:

```plaintext
winrm i GetRAIDLevels
cimv2/root/dcim/DCIM_RAIDSService?SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_RAIDSService
+Name=DCIM:RAIDS
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCAd
-encoding:utf-8 -a:basic -file:GetRAIDLevels.xml
```

The input file GetRAIDLevels.xml is shown below:

```xml
<p:GetRAIDLevels_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
    <p:Target>RAID.Integrated.1-1</p:Target>
    <p:DiskType>0</p:DiskType>
    <p:Diskprotocol>0</p:Diskprotocol>
    <p:DiskEncrypt>0</p:DiskEncrypt>
    <p:PDArray>Disk.Bay.0:Enclosure.Internal.0-0:RAID
        Integrated.1-1</p:PDArray>
    <p:PDArray>Disk.Bay.1:Enclosure.Internal.0-0:RAID
        Integrated.1-1</p:PDArray>
</p:GetRAIDLevels_INPUT>
```

OUTPUT:

```plaintext
GetRAIDLevels_OUTPUT
ReturnValue = 0
VDRAIDEnumArray = 2, 4
```

The VDRAIDEnumArray numbers correspond to the following RAID levels:

- RAID 0 = 2
- RAID 1 = 4
- RAID 5 = 64
- RAID 6 = 128
- RAID 10 = 2048
- RAID 50 = 8192
- RAID 60 = 16384

16.21.2 Getting the Available Disks-GetAvailableDisks()

The GetAvailableDisks() method is used to determine possible choices of drives to create virtual disks. Invoke GetAvailableDisks() with the following parameters and syntax:

**TARGET**: This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)

**DiskType**: Corresponds to MediaType attribute in PhysicalDiskView (Section 16.9)

- Include all types=0, Include Magnetic Only=1, Include SSD only=2

**Diskprotocol**: Types of protocol to include

- Include all protocols=0, Include SATA=1, Include SASTypes=2

**DiskEncrypt**: Types of encryption to include

- 0 = Include FDE capable and non encryption capable disks
- 1 = Include FDE disks only
- 2 = Include only non FDE disks
EXAMPLE:

winrm i GetAvailableDisks

The input file GetAvailableDisks.xml is shown below:

```xml
<p:GetAvailableDisks_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:DiskType>0</p:DiskType>
  <p:Diskprotocol>0</p:Diskprotocol>
  <p:DiskEncrypt>0</p:DiskEncrypt>
  <p:Raidlevel>2</p:Raidlevel>
</p:GetAvailableDisks_INPUT>

OUTPUT:

GetAvailableDisks_OUTPUT

PDArray = Disk.Bay.0:Enclosure.Internal.0-0:RAID.Integrated.1-1,
  Disk.Bay.1:Enclosure.Internal.0-0:RAID.Integrated.1-1

ReturnValue = 0

16.21.3 Checking the Create VD Parameters Validity-CheckVVDValues()

The CheckVVDValues() method is used to determine possible sizes of virtual disk as well default settings, given a RAID level and set of disks. The VDPropArray is filled in with Size and other values for a successful execution of the method.

Invoke CheckVVDValues() with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)

**PDArray:** This parameter is the list of physical disk FQDDs (Section 16.9)

**VDPropNameArrayIn:** This parameter is the list of property names with values in the VDPropValueArrayIn parameter Size, RAIDLevel, SpanDepth.

**VDPropValueArrayIn:** This parameter is the list of property values that correspond to the VDPropNameArrayIn parameter.

EXAMPLE:

winrm i CheckVVDValues
The input file CheckVDValues.xml is shown below:

```xml
<p:CheckVDValues_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService"
>
<p:Target>RAID.Integrated.1-1</p:Target>
<p:VDPropNameArrayIn>Size</p:VDPropNameArrayIn>
<p:VDPropValueArrayIn>10000</p:VDPropValueArrayIn>
<p:VDPropNameArrayIn>RAIDLevel</p:VDPropNameArrayIn>
<p:VDPropValueArrayIn>2048</p:VDPropValueArrayIn>
<p:VDPropNameArrayIn>SpanDepth</p:VDPropNameArrayIn>
<p:VDPropValueArrayIn>1</p:VDPropValueArrayIn>
</p:CheckVDValues_INPUT>

OUTPUT:
CheckVDValues_OUTPUT RebootRequired = YES ReturnValue = 0
VDPropNameArray = SizeInBytes, RAIDLevel, SpanDepth, SpanLength, StripeSize, ReadPolicy, WritePolicy, DiskCachePolicy, Name
VDPropValueArray = 1048576000, 2048, 2, 2, 128, 16, 2, 1024, Unknown

If the arrangement of physical disks prohibits a valid virtual disk arrangement from being created, such as having too few hard disks, the following error may result:
CheckVDValues_OUTPUT
Message = Virtual Disk provided is not valid for the operation MessageID = STOR017
ReturnValue = 2

16.21.4 Creating a Single Virtual Disk-CreateVirtualDisk()
The CreateVirtualDisk() method is used to create a single virtual disk on the targeted controller. The successful execution of this method results in a pending but not yet created virtual disk. The ObjectStatus property in the virtual disk view (Section 16.10) will have the value ‘3’, which represents pending. The virtual disk will not be created until a configuration job (Section 16.15) has been scheduled and the System is rebooted. Upon creation of the virtual disk, the FQDD of the formerly pending virtual disk will change.
Invoke CreateVirtualDisk() with the following parameters and syntax:
TARGET: This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)
PDArray: This parameter is the list of physical disk FQDDs that will be used to create a virtual Disk.
VDPropNameArray: This parameter is the list of property names that will be used to create a virtual disk. The parameter list contains the following names:
Size, RAIDLevel, SpanDepth, SpanLength, StripeSize, ReadPolicy, WritePolicy, DiskCachePolicy, VirtualDiskName, Initialize
VDPropValueArray: This parameter is the list of property values that will be used to create a virtual Disk. The property values are for the property names listed under VDPropNameArray.
Size: Size of the virtual disk specified in MB. If not specified, default will use full size of physical disks selected.
RAIDLevel:
- RAID 0 = 2
- RAID 1 = 4
- RAID 5 = 64
- RAID 6 = 128
- RAID 10 = 2048
- RAID 50 = 8192
- RAID 60 = 16384

SpanDepth: If not specified, default is single span which is used for RAID 0, 1, 5 and 6. Raid 10, 50 and 60 require a spandepth of at least 2.

SpanLength: Number of Physical Disks to be used per span. Minimum requirements for given RAID Level must be met.

StripeSize:
- 8KB = 16
- 16KB = 32
- 32KB = 64
- 64KB = 128
- 128KB = 256
- 256KB = 512
- 512KB = 1024
- 1MB = 2048

ReadPolicy:
- No Read Ahead = 16
- Read Ahead = 32
- Adaptive Read Ahead = 64

WritePolicy:
- Write Through = 1
- Write Back = 2
- Write Back Force = 4

DiskCachePolicy:
- Enabled = 512
- Disabled = 1024

VirtualDiskName: Name of the virtual disk (1-15 character range)

EXAMPLE:
```
winrm i CreateVirtualDisk
```

The input file CreateVirtualDisk.xml is shown below:
```
<p:CreateVirtualDisk_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
```
Creating a Sliced Virtual Disk - CreateVirtualDisk()

The CreateVirtualDisk() method is used to create a sliced virtual disk. A sliced virtual disk is created, if CreateVirtualDisk() Size input parameter value is less than total size of the physical disks. Additional sliced virtual disk can be created using the same set of physical disks and same RAID level that was used to create the first sliced virtual disk. If the physical disks have sliced virtual disks, then use the CheckVDValues() method on that set of physical disks to find the exact value for StartingLBA. Use this value as the StartingLBA parameter value of the CreateVirtualDisk() method.

The ObjectStatus property in the virtual disk view (see Section 16.10) has the value ‘3’, which represents a pending change. The virtual disk is not created until a configuration job (see Section 16.14) is scheduled and the System is rebooted. After the virtual disk creation, the FQDD of the pending virtual disk changes.

Invoke the CreateVirtualDisk() method with the following parameters and syntax:

**TARGET**: This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)

**PDArray**: This parameter is the list of physical disk FQDDs that is used to create a virtual Disk.

**VDPropNameArray**: This parameter is the list of property names that is used to create a virtual disk. The parameter list has the following names:
- Size, RAIDLevel, SpanDepth, SpanLength, StripeSize, ReadPolicy, WritePolicy, DiskCachePolicy, VirtualDiskName, Initialize.

**VDPropValueArray**: This parameter is the list of property values that is used to create a virtual Disk. The property values are for the property names listed under VDPropNameArray.
Size: Size of the virtual disk specified in MB. If not specified, default will use full size of physical disks selected.

RAIDLevel:
- RAID 0 = 2
- RAID 1 = 4
- RAID 5 = 64
- RAID 6 = 128
- RAID 10 = 2048
- RAID 50 = 8192
- RAID 60 = 16384

SpanDepth: If not specified, default is single span which is used for RAID 0, 1, 5 and 6. Raid 10, 50 and 60 require a spandepth of at least 2.

SpanLength: Number of Physical Disks to be used per span. Minimum requirements for given RAID Level must be met.

StripeSize:
- 8KB = 16
- 16KB = 32
- 32KB = 64
- 64KB = 128
- 128KB = 256
- 256KB = 512
- 512KB = 1024
- 1MB = 2048

ReadPolicy:
- No Read Ahead = 16
- Read Ahead = 32 Adaptive Read Ahead = 64

WritePolicy:
- Write Through = 1
- Write Back = 2 Write Back Force = 4

DiskCachePolicy:
- Enabled = 512
- Disabled = 1024

VirtualDiskName: Name of the virtual disk (1-15 character range)

StartingLBA: Starting logical block address of virtual disks in blocks. If 0xfffffffffffffffff, startingLBA is calculated programmatically. The value can be in hexadecimal or decimal format.
- 0xfffffffffffffffff
- 18446744073709551615

EXAMPLE:
```
winrm i CreateVirtualDisk

Example of using CreateVirtualDisk:
```

The input file CreateSlicedVirtualDisk.xml is shown below:
<p>CreateVirtualDisk_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:PDArray>DISK.Bay.0:Enclosure.Internal.0-0:RAID.Integrated.1-1</p:PDArray>
  <p:PDArray>DISK.Bay.1:Enclosure.Internal.0-0:RAID.Integrated.1-1</p:PDArray>
  <p:VDPropNameArray>RAIDLevel</p:VDPropNameArray>
  <p:VDPropNameArray>SpanDepth</p:VDPropNameArray>
  <p:VDPropNameArray>SpanLength</p:VDPropNameArray>
  <p:VDPropNameArray>Size</p:VDPropNameArray>
  <p:VDPropNameArray>VirtualDiskName</p:VDPropNameArray>
  <p:VDPropNameArray>StartingLBA</p:VDPropNameArray>
  <p:VDPropNameArray>4</p:VDPropNameArray>
  <p:VDPropNameArray>1</p:VDPropNameArray>
  <p:VDPropNameArray>2</p:VDPropNameArray>
  <p:VDPropNameArray>100</p:VDPropNameArray>
  <p:VDPropNameArray>virtualdiskname</p:VDPropNameArray>
  <p:VDPropNameArray>0xFFFFFFFFFFFFFFFF</p:VDPropNameArray>
</p>CreateVirtualDisk_INPUT>

**OUTPUT:**
The instanceID output identifies this virtual disk in the inventory before and after the CreateTargetedConfigJob() method creates it. However, the instanceID changes after successful creation.

CreateVirtualDisk_OUTPUT NewVirtualDisk
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
  SelectorSet
    Selector: InstanceID = DISK.Virtual.267386880:RAID.Integrated.1-1,__cimnamespace = root/dcim
  RebootRequired = YES
ReturnValue = 0

### 16.21.6 Creating a Cachecade Virtual Disk-CreateVirtualDisk()

The CreateVirtualDisk() method is used to create a Cachecade virtual disk on the targeted controller. This method internally creates a RAID-0 virtual disk. The creation process is the same as explained in Section 16.18.5. In this scenario, CreateVirtualDisk() method only takes VDPropNameArray-VDPropValueArray pairs mentioned below.

Invoke CreateVirtualDisk() with the following parameters and syntax:

- **TARGET**: This parameter is the FQDD of the DCIM_ControllerView (Section 16.7)
- **PDArray**: This parameter is the list of physical disk FQDDs that is used to create a virtual Disk.
- **VDPropNameArray**: This parameter is the list of property names that is used to create a virtual disk. The parameter list has the following names:
  - VirtualDiskName, CacheCade
VDPropValueArray: This parameter is the list of property values that is used to create a virtual Disk. The property values are for the property names listed under VDPropNameArray.

VirtualDiskName: Name of the virtual disk (1-15 character range)

Cachecade: The valid input value is 1. (required)

EXAMPLE:
winrm i CreateVirtualDisk

cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:CreateVDCacheCade.xml

The input file CreateVDCacheCade.xml is shown below:

```xml
<p:CreateVirtualDisk_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
    <p:Target>RAID.Integrated.1-1</p:Target>
    <p:VDPropNameArray>VirtualDiskName</p:VDPropNameArray>
    <p:VDPropValueArray>MyCacheCadeVD</p:VDPropValueArray>
    <p:VDPropNameArray>Cachecade</p:VDPropNameArray>
    <p:VDPropValueArray>1</p:VDPropValueArray>
</p:CreateVirtualDisk_INPUT>
```

OUTPUT:
The instanceID output identifies this virtual disk in the inventory before and after the CreateTargetedConfigJob() method creates it. Note however, that the instanceID will change slightly after successful creation.

CreateVirtualDisk_OUTPUT

NewVirtualDisk

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous

ReferenceParameters


SelectorSet

Selector: InstanceID = DISK.Virtual.267386880:RAID.Integrated.1-1, _cimnamespace = root/dcim

RebootRequired = YES

ReturnValue = 0

16.21.7 Deleting a Virtual Disk-DeleteVirtualDisk()

The DeleteVirtualDisk() method is used to delete a single virtual disk from the targeted controller. The successful execution of this method results in the marking of this virtual disk for deletion. The ObjectStatus property in the virtual disk view will have the value of ‘2’, which indicates pending delete. The virtual disk will not be deleted until a configuration job is scheduled and the System is rebooted (Section 16.15).
Invoke `DeleteVirtualDisk()` with the following parameters and syntax:

**TARGET:** This parameter is the FQDD of the virtual device (Section 16.10)

**EXAMPLE:**

```bash
winrm i DeleteVirtualDisk
cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:DeleteVirtualDisk.xml
```

The input file `DeleteVirtualDisk.xml` is shown below:

```xml
<p:DeleteVirtualDisk_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
    <p:Target>DISK.Virtual.0:RAID.Integrated.1-1</p:Target>
</p:DeleteVirtualDisk_INPUT>
```

**OUTPUT:**

```plaintext
DeleteVirtualDisk_OUTPUT
    RebootRequired = YES
    ReturnValue = 0
```

### 16.21.8 Rename Virtual Disk – `RenameVD()`

The `RenameVD` method is used to rename the existing virtual disk name. Maximum length of the name is dependent on the PERC. The name of the virtual disk will not be changed until a configuration job has been scheduled and performed. This configuration supported by both staged and realtime. Upon success of the method, virtual disk name will be changed in the property “Name” under “DCIM_VirtualDiskView” class.

Invoke `RenameVD()` method with the following input parameters:

**TARGET:** This parameter is the FQDD of the Virtual Disk (DCIM_VirtualDiskView, refer 17.10 section)

**NAME:** New name to be entered.

Note: Maximum length of the name is 15 characters and the name cannot start with a space, end with a space, or be left blank.

**EXAMPLE:**

```bash
winrm i RenameVD
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

@"Target="Disk.Virtual.4:RAID.Integrated.1-1";Name="[Proposed or new virtual disk name]"
```

**RenameVD_OUTPUT**

```plaintext
    RebootRequired = OPTIONAL
```
16.21.9 Cancel background initialization - CancelBackgroundInitialization()

This method is used to cancel the background initialization.

On PERC controllers, the background initialization of redundant virtual disk begins automatically after a virtual disk is created. The background initialization of redundant virtual disk prepares the virtual disk for parity information and improves write performance. It is important that the background initialization is allowed to be complete.

User has to wait for completion of background initialization till that user can’t perform any operation on virtual disk like “CheckConsistency”, OnlineCapacityExpansion etc. This feature provides the ability to cancel the background initialization manually. But if cancelled, the background initialization automatically restarts within 0 to 5 minutes.

This configuration is supported by only real time. Upon success of the method, properties "OperationalName" and "OperationPercentComplete" shows as "Back ground initialize" and "percentage in progress like 0% to 100%" in DCIM_VirtualDiskView class.

Invoke CancelBackgroundInitialization() method with the following input parameters:

**TARGET**: This parameter is the FQDD of the Virtual Disk (DCIM_VirtualDiskView, refer 17.10 section)

**EXAMPLE**:

```
winrm i CancelBackgroundInitialization
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
  +CreationClassName=DCIM_RAIDService
  +SystemName=DCIM:Computer
  +Name=DCIM:RAIDService
-u:root -p:calvin
-r:https://10.94.227.210/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic @{Target="Disk.Virtual.0:RAID.Integrated.1-1"}"
```

CancelBackgroundInitialization_OUTPUT

```
  RebootRequired = OPTIONAL
  ReturnValue = 0
```

16.21.10 Rebuild physical disk - RebuildPhysicalDisk()

This method is used to reconstruct the contents of failed disk onto a new disk or a hot spare.

If there is a redundant virtual disk, the rebuild operation can reconstruct the contents of a failed physical disk onto a new disk or a hot spare. We can perform this method manually only when “auto rebuild” option is set to false otherwise rebuild will automatically starts when one disk is fault in VD and the VD has hot spare.

This configuration is supported by both staged and realtime. Upon success of the method, properties "OperationalName" and "OperationPercentComplete" shows as "Rebuilding" and "percentage in progress like 0% to 100%" in DCIM_PhysicalDiskView class.

Invoke RebuildPhysicalDisk() method with the following input parameters:

**TARGET**: FQDD of the physical disk which is hot spare of the virtual disk.

**EXAMPLE**:

```
winrm i RebuildPhysicalDisk
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
  +CreationClassName=DCIM_RAIDService
  +SystemName=DCIM:ComputerSystem
  +Name=DCIM:RAIDService
-u:[USERNAME] -p:[PASSWORD]
```
16.21.11  Cancel rebuild physical disk -CancelRebuildPhysicalDisk()

This method is used to cancel the rebuild or reconstruct the contents of failed disk of the virtual disk to a new disk or a hot spare.

This configuration is supported by both staged and realtime. Upon success of the method, the “PrimaryStatus” in “DCIM_PhysicalDiskView” class has the value ‘3’, which represents “Error” status.

Invoke CancelRebuildPhysicalDisk() method with the following input parameters:

**TARGET:** FQDD of the physical disk which is hot spare of the virtual disk.

**EXAMPLE:**

```plaintext
winrm i CancelRebuildPhysicalDisk
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USERNAME] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SIncludeInvalidate
-encoding: utf-8 -a:basic
@{Target="Disk.Bay.11:Enclosure.Internal.0-1:RAID.Integrated.1-1"}
```

`CancelRebuildPhysicalDisk_OUTPUT`

- RebootRequired = OPTIONAL
- ReturnValue = 0

16.21.12  Online capacity expansion -OnlineCapacityExpansion()

This method allows to increase the storage capacity of selected RAID levels while the System remains online.

The operation can perform in two ways:

1. Increase the virtual disk capacity up to maximum if there is free space in physical disks.
2. Increase the virtual disk capacity by adding new physical disks.

This configuration supported by both staged and real time. Upon success of the method, properties "OperationalName" and "OperationPercentComplete" shows as "Reconstruction" and "percentage in progress like 0% to 100%" in DCIM_VirtualDiskView class.

Notes:

- OCE feature shall not be available when the Disk Group has more than one VD.
- OCE is not possible if sliced VD is available in disk group.
- If some operation like BGI, copyback, rebuild, ccheck, patrolread etc is already running on a VD then it will not be available for OCE.
- No OCE is supported on spanned VDs – thus no OCE is possible on RAID10, RAID50 and RAID60 VDs. But OCE is supported on single span RAID10 VDs.

If the disk group has reached its maximum span length limit then no OCE is possible by adding new drives to the disk group.

Invoke OnlineCapacityExpansion() method with the following input parameters:
TARGET: FQDD of the controller.
SIZE: New VD size in MB. This parameter is not required if new drive added for OCE
Note: Minimum size is greater than 100MB.
EXAMPLE: If user wants to increase the VD capacity from 1GB to 2GB then user has to pass 2GB in the parameter “Size”.
PDArray: FQDD of new physical disks.
Note: User required to pass either input parameter “Size” or “New physical disk” but not both. Hence both are required parameters at the same time both are optional parameters when one input parameter has provided.
EXAMPLE:
```
winrm i OnlineCapacityExpansion
```
```
OnlineCapacityExpansion_OUTPUT
  RebootRequired = OPTIONAL
  ReturnValue = 0
```

16.21.13 RAID level migration -RAIDLevelMigration()
This method is used to change the VD raid level.RAID level migration is the process of converting a VD with one RAID Level to another. The controller changes the write cache policy of all virtual disks undergoing a RLM to Write-Through until the RLM is complete.
This configuration is supported by both staged and realtime. Upon success of the method, properties "OperationalName" and "OperationPercentComplete" shows as "Reconstruction" and "percentage in progress like 0% to 100%" in DCIM_VirtualDiskView class.
The operation can perform in two ways:
1. By adding new physical disks (change R0 to R1 by adding new drive, if the source raid level has only one disk)
2. Without adding physical disks (change R0 to R1 without adding new drive, if the source raid level already has minimum required disks of 2).

Notes:
- No RLM is supported on spanned VDs – thus no RLM s possible on RAID10, RAID50 and RAID60 VDs.
- No RLM is supported if other operations like BGI, rebuild, copyback, ccheck, patrolread etc is running on a VD.

Possible reconfigurable VD layout:

<table>
<thead>
<tr>
<th>Source VD RAID level</th>
<th>Possible migrated VD RAID levels with new disk addition.</th>
<th>Possible migrated VD RAID levels without new disk addition.</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0 (single drive)</td>
<td>R1</td>
<td>NA</td>
</tr>
<tr>
<td>R0</td>
<td>R5/R6</td>
<td>NA</td>
</tr>
<tr>
<td>R1</td>
<td>R0/R5/R6</td>
<td>R0</td>
</tr>
<tr>
<td>Source VD RAID level</td>
<td>Possible migrated VD RAID levels with new disk addition.</td>
<td>Possible migrated VD RAID levels without new disk addition.</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------</td>
<td>---------------------------------------------------------</td>
</tr>
<tr>
<td>R5</td>
<td>R0/R6</td>
<td>R0</td>
</tr>
<tr>
<td>R6</td>
<td>R0/R5</td>
<td>R0/R5</td>
</tr>
</tbody>
</table>

Invoke RAIDLevelMigration() method with the following input parameters:

**TARGET**: FQDD of the controller.

**NewRAIDLevel**: New RAID level to be migrated.

**PDArray**: FQDD of the physical disks to be included in the New RAID-Level.

**EXAMPLE**:

```plaintext
```

**RAIDLevelMigration_OUTPUT**

```
RebootRequired = OPTIONAL
ReturnValue = 0
```

### 16.21.14 Set enclosure asset name -SetAssetName()

The method is used to set the external enclosure asset name. This configuration is supported by both staged and real time. Upon success of the method, the property “AssetName” under DCIM_EnclosureView class will be changed with the proposed name.

Invoke SetAssetName() method with the following input parameters:

**TARGET**: FQDD of the enclosure.

**AssetName**: New asset name to be entered. The AssetName value shall be maximum of 32 ASCII alphanumeric characters length.

**EXAMPLE**:

```plaintext
```

### 16.21.15 Set enclosure asset tag -SetAssetTag()

The method is used to set the external enclosure asset tag. This configuration is supported by both staged and real time. Upon success of the method, the property “AssetTag” under DCIM_EnclosureView class will be changed with the proposed value.
Invoke SetAssetTag() method with the following input parameters:

**TARGET**: FQDD of the enclosure.

**AssetTag**: New asset tag to be entered. The AssetName value shall be maximum of 10 ASCII alphanumeric characters length.

**EXAMPLE:**

```
winrm i SetAssetTag
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_RAIDService?
+SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService
+Name=DCIM:RAIDService
-u:[USERNAME] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
@{Target="Enclosure.External.0-0:RAID.Slot.1-1";AssetTag="enc123"}
```

### 16.22 Setting Controller Attributes

#### 16.22.1 Changing the Value of a RAID Controller Enumeration Attribute

The SetAttribute() method is used to set or change the value of a RAID controller or a virtual disk attribute. The example below shows setting a RAID controller enumeration attribute. To set a virtual disk attribute, use the FQDD of the virtual disk attribute for the Target, and the AttributeName and AttributeValue.

Invoke SetAttribute() with the following parameters (from Section 16.1) and syntax:

**TARGET**: Obtained from the FQDD field

**AttributeName**: Obtained from the AttributeName field

**AttributeValue**: Obtained from the PossibleValues field

**EXAMPLE:**

```
winrm i SetAttribute
cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService+System Name=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttribute_Enumeration_RAID_Controller.xml
```

The input file `SetAttribute_Enumeration_RAID_Controller.xml` is shown below:

```
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:AttributeName>RAIDBatteryLearnMode</p:AttributeName>
  <p:AttributeValue>Disabled</p:AttributeValue>
</p:SetAttribute_INPUT>
```

**OUTPUT:**

```
SetAttribute_OUTPUT
Message = The method was successful.
MessageID = STOR001
```
Changing Multiple Values of RAID Controller Enumeration Attributes

The `SetAttributes()` method is used to set or change multiple values of RAID controller or virtual disk attributes. The following example shows setting multiple virtual disk attributes. To set multiple controller attributes, use the FQDD of the controller for the Target, and the `AttributeName` and `AttributeValue`.

Invoke `SetAttributes()` with the following parameters (from Section 16.1) and syntax:

**TARGET**: Obtained from the FQDD field

**AttributeName**: Obtained from the `AttributeName` field

**AttributeValue**: Obtained from the `PossibleValues` field

**EXAMPLE:**

```
winrm i SetAttributes
  cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
  +CreationClassName=DCIM_RAIDService+System Name=DCIM:ComputerSystem
  -u:[USER] -p:[PASSWORD]
  -r:https://[IPADDRESS]/wsman
  -SkipCNcheck -SkipCACheck
  -encoding:utf-8 -a:basic
  -file:SetAttributes_Enumeration_RAID_Controller.xml
```

The input file `SetAttributes_Enumeration_RAID_Controller.xml` is shown below:

```xml
<p:SetAttributes_INPUT
  xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:AttributeName>RAIDloadBalancedMode</p:AttributeName>
  <p:AttributeValue>Disabled</p:AttributeValue>
  <p:AttributeName>RAIDBatteryLearnMode</p:AttributeName>
  <p:AttributeValue>Warn only</p:AttributeValue>
  <p:AttributeName>RAIDccMode</p:AttributeName>
  <p:AttributeValue>Normal</p:AttributeValue>
  <p:AttributeName>RAIDprMode</p:AttributeName>
  <p:AttributeValue>Disabled</p:AttributeValue>
  <p:AttributeName>RAIDcopybackMode</p:AttributeName>
  <p:AttributeValue>SMART</p:AttributeValue>
</p:SetAttributes_INPUT>
```

**OUTPUT:**

```
SetAttributes_OUTPUT
Message = The method was successful.
MessageID = STOR001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set Pending Value
```
16.22.3 Changing the Value of a RAID Controller Integer Attribute

The SetAttribute() method is used to set or change the value of a RAID controller integer attribute. The example below shows setting an controller attribute.

Invoke the SetAttribute() method with the following parameters (from Section 16.1) and syntax:

**TARGET**: Obtained from the FQDD field

**AttributeName**: Obtained from the AttributeName field

**AttributeValue**: Obtained from the PossibleValues field

**EXAMPLE**:

```plaintext
winrm i SetAttribute

cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService+System Name=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:b
-file:SetAttribute_Integer_RAID_Controller.xml
```

The input file SetAttribute_Integer_RAID_Controller.xml is shown below:

```xml
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:AttributeName>RAIDccRate</p:AttributeName>
  <p:AttributeValue>60</p:AttributeValue>
</p:SetAttribute_INPUT>
```

**OUTPUT**:

```
SetAttribute_OUTPUT
  Message = The method was successful.
  MessageID = STOR001
  RebootRequired = Yes
  ReturnValue = 0
  SetResult = Set Pending Value
```

16.22.4 Changing Multiple Values of RAID Controller Integer Attributes

The SetAttributes() method is used to set or change multiple values of RAID controller attributes. The following example shows setting multiple RAID controller integer attributes.

Invoke SetAttributes with the following parameters (from Section 16.1) and syntax:

**TARGET**: Obtained from the FQDD field

**AttributeName**: Obtained from the AttributeName field

**AttributeValue**: Obtained from the PossibleValues field

**EXAMPLE**:

```plaintext
winrm i SetAttributes

cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_RAIDService+System Name=DCIM:ComputerSystem
+Name=DCIM:RAIDService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:b
-file:SetAttributes_Integer_RAID_Controller.xml
```
The input file SetAttributes_Integer_RAID_Controller.xml is shown below:

```xml
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_RAIDService">
  <p:Target>RAID.Integrated.1-1</p:Target>
  <p:AttributeName>RAIDccRate</p:AttributeName>
  <p:AttributeValue>60</p:AttributeValue>
  <p:AttributeName>RAIDreconstructRate</p:AttributeName>
  <p:AttributeValue>60</p:AttributeValue>
  <p:AttributeName>RAIDbgiRate</p:AttributeName>
  <p:AttributeValue>60</p:AttributeValue>
</p:SetAttributes_INPUT>

OUTPUT:
SetAttributes_OUTPUT
Message = The method was successful.
MessageID = STOR001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set Pending Value

16.23 Convert Physical Disks to RAID-ConvertToRAID()

The ConvertToRAID() method is used to convert physical disks in Non-RAID state to a state usable for RAID. After the method is successfully executed the PendingValue property of RAIDPDState should reflect the pending changes. After the CreateTargetedConfigJob() method is successfully executed the RAIDStatus property, which is enumerated in the DCIM_PhysicalDiskView from Section 16.9, of that physical disk should reflect the new state.

Invoke ConvertToRAID() with the following parameters and syntax:

**Physical Disk TARGET**: Obtained from the FQDD field (Section 16.9)

An example of Disk.Bay.2:Enclosure.Internal.0-0:RAID.Slot.1-1 is shown below.

**EXAMPLE:**

```bash
winrm invoke ConvertToRAID
"cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem+CreationClassName=DCIM_RAIDService+
SystemName=DCIM:ComputerSystem+
Name=DCIM:RAIDService"
@{PDArray="Disk.Bay.2:Enclosure.Internal.0-0:RAID.Slot.1-1"}
-u: [USER] -p: [PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding: utf-8 -a: basic
-format: pretty

OUTPUT:
ConvertToRAID_OUTPUT
RebootRequired = 1
ReturnValue = 0
```
16.24 Convert Physical Disks to Non RAID-ConvertToNonRAID()

The ConvertToNonRAID() method is used to convert a physical disks in RAID state of “Ready” to a NonRAID state. After the method is successfully executed, the PendingValue property of RAIDPDState should reflect the pending changes. After the CreateTargetedConfigJob method is successfully executed, the RAIDStatus property, which is enumerated in the DCIM_PhysicalDiskView from Section 16.9, of that physical disk should reflect the new state.

Invoke ConvertToNonRAID() with the following parameters and syntax:

**Physical Disk TARGET**: Obtained from the FQDD field (Section 16.9)

An example of Disk.Bay.2:Enclosure.Internal.0-0:RAID.Slot.1-1 is shown below.

**EXAMPLE:**

```
winrm invoke ConvertToNonRAID
"cimv2/root/dcim/DCIM_RAIDService?SystemCreationClassName=DCIM_ComputerSystem+
CreationClassName=DCIM_RAIDService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:RAIDService"
@{PDArray="Disk.Bay.2:Enclosure.Internal.0-0:RAID.Slot.1-1"}
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-format:pretty
```

**OUTPUT:**

```
ConvertToNonRAID_OUTPUT
  RebootRequired = 1
  ReturnValue = 0
```
17 Managing BIOS Configuration

This feature provides the ability to get and set any configurable BIOS attributes that are exposed in BIOS UEFI HII. The BIOS Management Profile extends the management capabilities of referencing profiles by adding the capability to represent and configure BIOS attributes, such as a Network Controller or IDE Controller.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

17.1 Listing the BIOS Inventory-Enumeration Class

The BIOS Inventory contains the following attributes: DCIM_BIOSEnumeration (17.1), DCIM_BIOSInteger (17.5), DCIM_BIOSString (17.6), and DCIM_BIOSPassword (17.10). Enumerating the BIOSEnumeration Class will display all BIOS attributes in a computer System.

Enumerate BIOSEnumeration with the following parameters and syntax:

EXAMPLE:

```bash
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSEnumeration
-u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman -SkipCNcheck -SkipCACheck -encoding: utf-8 -a: basic
```

OUTPUT:

```
DCIM_BIOSEnumeration
  AttributeName = NumLock
  CurrentValue = On
  DefaultValue = null
  FQDD = BIOS.Setup.1-1
  InstanceID = BIOS.Setup.1-1:NumLock
  IsReadOnly = false
  PendingValue = null
  PossibleValues = On, Off

DCIM_BIOSEnumeration
  AttributeName = ReportKbdErr
  CurrentValue = Report
  DefaultValue = null
  FQDD = BIOS.Setup.1-1
  InstanceID = BIOS.Setup.1-1:ReportKbdErr
  IsReadOnly = false
  PendingValue = null
  PossibleValues = Report, NoReport

DCIM_BIOSEnumeration
  AttributeName = BootMode
  CurrentValue = Bios
  DefaultValue = null
  FQDD = BIOS.Setup.1-1
```
InstanceID = BIOS.Setup.1-1:BootMode
IsReadOnly = false
PendingValue = null
PossibleValues = Bios, Uefi.

DCIM_BIOSEnumeration
AttributeName = BootSeqRetry
CurrentValue = Disabled
DefaultValue = null
FQDD = BIOS.Setup.1-1
InstanceID = BIOS.Setup.1-1:BootSeqRetry
IsReadOnly = false
PendingValue = null
PossibleValues = Disabled, Enabled

17.2 Getting a BIOS Enumeration Instance

Getting one particular instance of the BIOSEnumeration, instead of all instances as shown in Section 17.1, is shown below.

Get a BIOSEnumeration instance with the following parameters and syntax:

[INSTANCEID]: This is obtained from the enumeration in Section 17.1, which shows an example using BIOS.Setup.1-1:NumLock as an instanceID

EXAMPLE:

winrm g
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSEnumeration?
InstanceID=[INSTANCE ID]
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_BIOSEnumeration
AttributeName = NumLock
CurrentValue = On
DefaultValue = null
FQDD = BIOS.Setup.1-1
InstanceID = BIOS.Setup.1-1:NumLock
IsReadOnly = false
PendingValue = null
PossibleValues = On, Off

17.3 Changing the BIOS BootMode-SetAttribute ()

The SetAttribute () method can be used to apply changes to setting the BootMode configuration to a given instance.

Invoke SetAttribute () with the following parameters (from Section 17.1) and syntax:

TARGET: Obtained from the InstanceID field
AttributeName: Obtained from the AttributeName field
**AttributeValue**: Obtained from the PossibleValues field

**EXAMPLE:**

```plaintext
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSService?
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic -file:SetAttribute_BIOS.xml
```

The input file `SetAttribute_BIOS.xml` is shown below:

```xml
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOS"/>
  <p:Target>BIOS.Setup.1-1</p:Target>
  <p:AttributeName>BootMode</p:AttributeName>
  <p:AttributeValue>Bios</p:AttributeValue>
</p:SetAttribute_INPUT>
```

**OUTPUT:**

```plaintext
SetAttribute_OUTPUT
Message = The command was successful
MessageID = BIOS001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set PendingValue
```

### 17.4 Setting Multiple BIOS BootMode Parameters

Users can find and set multiple BIOS attributes associated with a specific device using the `SetAttributes()` method. This example illustrates how to set the BiosMode and BootSeqRetry parameters.

Invoke `SetAttributes()` with the following parameters (from Section 17.1) and syntax:

**TARGET**: Obtained from the InstanceID field

**AttributeName**: Obtained from the AttributeName field

**AttributeValue**: Obtained from the PossibleValues field

**EXAMPLE:**

```plaintext
winrm i SetAttributes
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSService?
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+Name=DCIM:BIOSService -u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttributes_BIOS.xml
```

The input file `SetAttributes_BIOS.xml` is shown below:

```xml
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOS"/>
```

```xml
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOS"/>
  <p:Target>BIOS.Setup.1-1</p:Target>
  <p:AttributeName>BootMode</p:AttributeName>
  <p:AttributeValue>Bios</p:AttributeValue>
</p:SetAttribute_INPUT>
```

**OUTPUT:**

```plaintext
SetAttribute_OUTPUT
Message = The command was successful
MessageID = BIOS001
RebootRequired = Yes
ReturnValue = 0
SetResult = Set PendingValue
```
<p>OUTPUT:  
SetAttribute_OUTPUT  
Message = The command was successful  
MessageID = BIOS001  
RebootRequired = Yes  
ReturnValue = 0  
SetResult = Set PendingValue</p>

17.5 **Listing the BIOS Inventory-Integer Class**  
Enumerate BIOSInteger with the following parameters and syntax:  
**EXAMPLE:**  
winrm e  
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSInteger  
-u:[USER] -p:[PASSWORD]  
-r:https://[IPADDRESS]/wsman  
-SkipCNcheck -SkipCACheck  
-encoding:utf-8 -a:basic  
**OUTPUT:**  
DCIM_BIOSInteger  
AttributeName = AcPwrRcvryUserDelay  
CurrentValue = 0  
DefaultValue = null  
FQDD = BIOS.Setup.1-1  
InstanceID = BIOS.Setup.1-1:AcPwrRcvryUserDelay  
IsReadOnly = true  
LowerBound = 30  
PendingValue = null  
UpperBound = 240

17.6 **Listing the BIOS Inventory-String Class**  
Enumerate BIOSString with the following parameters and syntax:  
**EXAMPLE:**  
winrm e  
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSString  
-u:[USER] -p:[PASSWORD]  
-r:https://[IPADDRESS]/wsman  
-SkipCNcheck -SkipCACheck  
-encoding:utf-8 -a:basic  
**OUTPUT:**  
DCIM_BIOSString  
AttributeName = OneTimeCustomBootStr
17.7 Applying the Pending Values for BIOS & BootCreateTargetedConfigJob()

This method is called to apply the pending values created by the SetAttribute(), SetAttributes(), ChangeBootOrderByInstanceID(), and ChangeBootSourceState() methods. The System will automatically reboot depending on the ScheduledStartTime selected. Using the CreateTargetedConfigJob() jobId output with the job control section can be used to obtain its status.

Invoke CreateTargetedConfigJob() with the following parameters and syntax:

**TARGET**: This Parameter is the FQDD of the BIOSAttribute instances, obtained from the InstanceID field in Section 17.1

**RebootJobType**: There are three options for rebooting the System.

1 = PowerCycle
2 = Graceful Reboot without forced shutdown
3 = Graceful reboot with forced shutdown
Note: When a user does not want to set a reboot type when creating a target job, users should comment out the RebootJobType in the input xml. User should not enter “0” or give no parameter at all in the input xml.

EXAMPLE:

winrm i CreateTargetedConfigJob
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_BIOSService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:BIOSService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:CreateTargetedConfigJob_BIOS.xml

The input file CreateTargetedConfigJob_BIOS.xml is shown below:

```xml
<p:CreateTargetedConfigJob_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOSService">
  <p:Target>BIOS.Setup.1-1</p:Target>
  <p:RebootJobType>2</p:RebootJobType>
  <p:ScheduledStartTime>TIME_NOW</p:ScheduledStartTime>
  <p:UntilTime>20111111111111</p:UntilTime>
</p:CreateTargetedConfigJob_INPUT>
```

OUTPUT:

When this method is executed, a jobid or an error message is returned. The status of this jobid can be checked within the job control provider in Section 10.

CreateTargetedConfigJob_OUTPUT

Job

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet
  Selector: InstanceID = JID_001269609760, __cimnamespace = root/dcim
ReturnValue = 4096

17.8 Deleting the Pending Values for BIOS & BootDeletePendingConfiguration()

This method is called to cancel the pending values created by the SetAttribute() and SetAttributes() methods. The DeletePendingConfiguration() method cancels the pending configuration changes made before the configuration job is created with CreateTargetedConfigJob(). This method only operates on the pending changes prior to CreateTargetedConfigJob() being called. After the configuration job is created, the pending changes can only be canceled by calling DeleteJobQueue() in the Job Control profile.

Invoke CreateTargetedConfigJob() with the following parameters and syntax:

**Target**: This parameter is the FQDD of the BIOSAttribute instances (from Section 17.1)
**EXAMPLE:**

```
winrm i DeletePendingConfiguration
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSService?
+SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_BIOSService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:BIOSService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNCHECK -SkipCACheck
-encoding:utf-8 -a:basic
-file:DeletePendingConfiguration_BIOS.xml
```

The input file `DeletePendingConfiguration_BIOS.xml` is shown below:

```xml
<p:DeletePendingConfiguration_INPUT xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOSService">
  <p:Target>BIOS.Setup.1-1</p:Target>
</p:DeletePendingConfiguration_INPUT>
```

**OUTPUT:**

```
DeletePendingConfiguration_OUTPUT Message = The command was successful
  MessageID = BIOS001
  ReturnValue = 0
  ReturnValue = 4096
```

### 17.9 Managing BIOS Passwords

The `ChangePassword()` method is used to set the BIOS passwords. The user can either set, change or delete the BIOS System or setup password. Setting the BIOS password is performed in several stages as described in the following sections.

#### 17.9.1 Setting the BIOS Password

The following example sets the BIOS System password to "NEW_PASSWORD". Three instances of XML are shown below to demonstrate the following scenarios:

- No BIOS password is set
- Changing an existing BIOS password
- Deleting an existing BIOS password

Invoke `ChangePassword()` method with the following parameters:

- **Target** - Obtained from any BIOS enumerate WSMAN command
- **PasswordType** - Either 1 for System or 2 for setup
- **OldPassword** – Reference following XML case A), B) or C) **NewPassword** - Reference following XML case A), B) or C)

**EXAMPLE:**

```
winrm i ChangePassword
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSService?
+SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_BIOSService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:BIOSService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
```
The input file change_bios_password.xml is shown below:

- No BIOS password is set: The OldPassword parameter is not required. It may be set to “null” or left blank as shown below.
- Changing an existing BIOS password: Both the OldPassword and NewPassword parameters are required.

**Note:** Entering only the NewPassword parameter indicates a “pass” in the setting and creating a new job, however the job fails.

- Deleting an existing BIOS password: The OldPassword parameter is required. The NewPassword parameter may be set to “null”, set to blank, or omitted completely.

```xml
<p:ChangePassword_INPUT
  <p:Target>BIOS.Setup.1-1</p:Target>
  <p:PasswordType>1</p:PasswordType>
  <p:OldPassword></p:OldPassword>
  <p:NewPassword>NEW_PASSWORD</p:NewPassword>
</p:ChangePassword_INPUT>
```

**OUTPUT:**

Either of the following may result:

ChangePassword_OUTPUT
- Message = BIOS does not support Change Password feature
  MessageID = BIOS019
  ReturnValue = 2

ChangePassword_OUTPUT
- Message = The command was successful
  MessageID = BIOS001

### 17.9.2 Create Target Configuration Job
Create a configuration job as shown in Section 17.7.

### 17.9.3 Monitor Set BIOS Password Status
To monitor the job status for setting the BIOS password, get the instance of the corresponding job as described within the job control provider in Section 10.

Replace [INSTANCE ID] with the actual jobid from Section 17.9.1.

**EXAMPLE:**

```
winrm get
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LifecycleJob?
InstanceID=[INSTANCE_ID]
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-a:basic encoding:utf-8
```

**OUTPUT:**

```
DCIM_LifecycleJob
  InstanceID = JID_00129609760
```
JobStartTime = 00000101000000
JobStatus = Scheduled
JobUntilTime = TIME_NA
Message = Task successfully scheduled
MessageID = JCP001
Name = ConfigBIOS:BIOS.Setup.1-1
PercentComplete = NA

The status may be one of the following:
- Ready for execution — Job is created, but waiting for scheduled start time to pass to schedule the job
- Scheduled — Job is scheduled and ready for System reboot to execute the job
- Failed — Problem with setting the BIOS password, check message for more information
- Completed — Setting the BIOS password completed with no issues

17.10 Listing the BIOS Inventory-Password Class

Enumerate BIOSPassword with the following parameters and syntax:

**EXAMPLE:**

```
winrm e cimv2/root/dcim/DCIM_BIOSPassword
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
DCIM_BIOSPassword
    AttributeDisplayName = System Password
    AttributeName = SysPassword
    Dependency = <Dep><AttrLev Op="OR"><ROIf Name="PasswordStatus">Locked</ROIf></AttrLev></Dep>
    DisplayOrder = 1402
    FQDD = BIOS.Setup.1-1
    GroupDisplayName = System Security
    GroupID = SysSecurity
    InstanceID = BIOS.Setup.1-1:SysPassword
    IsReadOnly = false
    IsSet = false
    MaxLength = 32
    MinLength = 0
    PasswordState = 3
    PendingValue = null
    ValueExpression = ^[\]0-9a-z "+,-/.\"\{\}0,32$`
```

**DCIM_BIOSPassword**

```
AttributeDisplayName = Setup Password
AttributeName = SetupPassword
Dependency = null
DisplayOrder = 1403
```
FQDD = BIOS.Setup.1-1
GroupDisplayName = System Security
GroupID = SysSecurity
InstanceID = BIOS.Setup.1-1:SetupPassword
IsReadOnly = false
IsSet = false
MaxLength = 32
MinLength = 0
PasswordState = 3
PendingValue = null
ValueExpression = ^\[0-9a-z "+,-.\[\]`]{0,32}$
Exporting and Importing Server Profile

Use this feature to back up and restore host server profile. You can take a backup of current System configuration that is stored in a backup image file. Use Restore at anytime to put the System to pre-backup state.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

18.1 Exporting Server Profile

To backup host System server profile, invoke the BackupImage() method in the class DCIM_LCService. Backup feature gathers System information, firmware images, hardware configuration, Lifecycle Controller, iDRAC firmware, and configuration and stores the information in a file. You can save the file on either iDRAC vFlash SD card or network share.

[IP ADDRESS]: This is the IP address of the file server.
[DRIVESHARE]: This is the directory path for the image.
[USERNAME]: This is the username to the file share.
[PASSWORD]: This is the password to the file share.
[IMAGENAME]: This is the desired name of the image.
[PASSPHRASE]: This can be used to password protect NFS and CIFS images.

For NFS and CIFS shares, the entire “Passphrase="[PASSPHRASE];" argument is optional. Note: To restore this backup file, the same passphrase must passed as an argument for the operation to be successful.

The following examples back up the server profile and execute it immediately, using the TIME_NOW parameter.

18.1.1 Exporting Server Profile to iDRAC vFlash Card-BackupImage()

iDRAC vFlash Card:
ShareType is “4”.

EXAMPLE:

winrm i BackupImage
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:Computer
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]:443/wsman
-SkipCNcheck -SkipCACheck
-encoding=utf-8 a:basic @(IPAddress="[IP ADDRESS]";
ShareType="4";ScheduledStartTime="TIME_NOW")

18.1.2 Exporting Server Profile to NFS Share-BackupImage()

NFS Share:
ShareType is “0”. The entire “Passphrase="passphrase";” argument is optional.

EXAMPLE:

winrm i BackupImage
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

@{IPAddress="[IPADDRESS]";ShareName="/[DRIVESHARE]";ShareType="0";ImageName="[IMAGENAME]";
Username="[USERNAME]";Password="[PASSWORD]";Passphrase="[PASSPHRASE]";
ScheduledStartTime="TIME_NOW"
}

Note: The ShareName field should only be the folder exposed by the System to the network. Any subfolder information should be attached to the ImageName field. Otherwise, there can be connection issues when trying to locate/create the backup file.

Correct EXAMPLE: ShareName="/folder1";ImageName="subfolder/image_name"
Incorrect EXAMPLE: ShareName="/folder1/subfolder";ImageName="image_name"

18.1.3 Exporting Server Profile to CIFS Share-BackupImage()

CIFS Share: ShareType is “2”. The entire “Passphrase="passphrase";” argument is optional.

EXAMPLE:

```
winrm i BackupImage
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

@{IPAddress="[IP ADDRESS]";ShareName="/[DRIVESHARE]";ShareType="2";ImageName="[IMAGENAME]";
Username="[USERNAME]";Password="[PASSWORD]";Passphrase="[PASSPHRASE]";
ScheduledStartTime="TIME_NOW"
}
```

Note: The ShareName field should only be the folder exposed by the System to the network. Any subfolder information should be attached to the ImageName field. Otherwise, there can be connection issues when trying to locate/create the backup file.

Correct EXAMPLE: ShareName="/folder1";ImageName="subfolder/image_name"
Incorrect EXAMPLE: ShareName="/folder1/subfolder";ImageName="image_name"

OUTPUT:

```
BackupImage_OUTPUT
Job
  Address = http://schemas.dxmlsoap.org/ws/2004/08/addressing/role/anonymous
  ReferenceParameters
  ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_LifecycleJob
  SelectorSet
    Selector: InstanceID = JID_001293618214, __cimnamespace = root/dcim
  ReturnValue = 4096
```

The response contains a reference to the job class that will provide the status of the operation. The return value is 4096 which indicates that the method operation is not yet complete.
18.1.4 Monitoring Export status
Backup process may take up to 30 minutes depending on host System configuration. To monitor the backup status, get the instance of the corresponding job.
Replace [INSTANCE ID] with the actual jobid from Section 18.1.1, 18.1.2, or 18.1.3.

EXAMPLE:

```
winrm get http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LifecycleJob?
InstanceID=[INSTANCE ID]
-u:[USER] p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-a:basic encoding:utf-8
```

OUTPUT:

```
DCIM_LifecycleJob
   InstanceID = JID_001293618214
   JobStartTime = 00000100000000
   JobStatus = Backup In Progress
   JobUntilTime = TIME_NA
   Message = Collecting Lifecycle Controller Firmware images
   MessageID = BAR063
   Name = Backup:Image
   PercentComplete = 50
```

The status may be one of the following:

- **Ready for Backup** — Request is received
- **Backup In Progress** — Backup process is currently in process
- **Failed** — Problem with the backup process, check message for more information
- **Completed** — Backup process is complete with no issues

18.2 Automatic Backup
Automatic Backup feature allows for creating backup server profiles periodically and exporting to a CIFS/NFS share or to the vFlash.

18.2.1 Enable the Automatic Backup
Enable the Automatic Backup feature by setting the attribute and configuring a job to update the attribute setting

```
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:<USER> -p:<PASSWORD>
-r:https://<IDRAC_IP_ADDRESS>/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic @{AttributeName="Automatic Backup Feature";AttributeValue="Enabled"}
```
winrm i CreateConfigJob
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:<USER> -p:<PASSWORD>
-r:https://<IDRAC_IP_ADDRESS>/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:
CreateConfigJob_OUTPUT

Job
EndpointReference
Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
ResourceURI = http://schemas.dell.com/wbem/wscim/1/cimschema/2/DCIM_LifecycleJob
SelectorSet
Selector: InstanceID = JID_859945354433, __cimnamespace = root/dcim
ReturnValue = 4096

18.2.2 Set Backup Schedule
The SetBackupSchedule () method used by the DCIM_LCService class is used to configure automatic backup schedule using WSMAN

winrm i SetBackupSchedule
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:<USER> -p:<PASSWORD>
-r:https://<IDRAC_IP_ADDRESS>/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic -file:autobackup.xml

OUTPUT:
The autobackup.xml file contains the parameters to be passed to the SetBackupSchedule() method in XML format. A sample autobackup.xml file is given here.

<p:SetBackupSchedule_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">'
<p:ShareType>4</p:ShareType>
<p:ImageName>IMAGENAME.img</p:ImageName>
<p:Time>12:56</p:Time>
<p:DayofMonth>*</p:DayofMonth>
<p:DayOfWeek>Mon</p:DayOfWeek>
<p:WeekOfMonth>L</p:WeekOfMonth>
<p:Passphrase>PASSPHRASE</p:Passphrase>
<p:Repeat>1</p:Repeat>
18.2.3 Get the Backup Schedule
The GetBackupSchedule() method used by the DCIM_LCService class is used to get the automatic backup schedule

```
winrm i GetBackupSchedule
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:<USER> -p:<PASS>
-r:https://<IPADDRESS>/wsman
-SkipCNcheck -SkipCACheck encoding:utf-8 -a:basic
```

18.2.4 Clear the Backup Schedule
The ClearBackupSchedule() method used by the DCIM_LCService class is used to configure automatic backup schedule

```
winrm i ClearBackupSchedule
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:<USER> -p:<PASS>
-r:https://<IPADDRESS>/wsman
-SkipCNcheck -SkipCACheck encoding:utf-8 -a:basic
```

18.3 Importing Server Profile
To restore host System server profile, invoke the RestoreImage() method in the class DCIM_LCService.

Restore process restores the System information, firmware images, hardware configuration, Lifecycle Controller, iDRAC firmware, and configuration from the backup image file located on either iDRAC vFlash SD card or network share.

**[IP ADDRESS]**: This is the IP address of the file server.

**[DRIVESHARE]**: This is the directory path for the image.

**[USERNAME]**: This is the username to the file share.

**[PASSWORD]**: This is the password to the file share.

**[IMAGENAME]**: This is the desired name of the image.

**[PASSPHRASE]**: This can be used to password protect NFS and CIFS images.

For NFS and CIFS shares, the entire "Passphrase=[PASSPHRASE]"; argument is only required when the backup image used a passphrase.

The following examples restore the server profile and execute it immediately, using the TIME_NOW parameter.

18.3.1 Importing Server Profile from iDRAC vFlash Card-RestoreImage()

iDRAC vFlash Card:
ShareType is “4”.

```
winrm i RestoreImage
```
18.3.2 Importing Server Profile from NFS share—RestoreImage()

NFS Share:

- ShareType is "0".

**EXAMPLE:**

```bash
winrm i RestoreImage http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem +CreationClassName=DCIM_LCService +SystemName=DCIM:ComputerSystem +Name=DCIM:LCService -u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman -SkipCNCheck -SkipCACheck -encoding:utf-8 -a:basic @ {IPAddress="[IP ADDRESS]";ShareName="/SHARENAME";ShareType="2";Username="[USERNAME]";Password="[PASSWORD]";ImageName="[IMAGENAME]";Passphrase="[PASSPHRASE]";ScheduledStartTime="TIME_NOW"}
```

**Note:** The ShareName field should only be the folder exposed by the System to the network. Any sub folder information should be attached to the ImageName field. Otherwise, there can be connection issues when trying to locate or create the backup file.

**Correct EXAMPLE:** ShareName="/folder1";ImageName="subfolder/image_name"

**Incorrect EXAMPLE:** ShareName="/folder1/subfolder";ImageName="image_name"

18.3.3 Importing Server Profile from CIFS share—RestoreImage()

CIFS Share:

- ShareType is "2".

**EXAMPLE:**

```bash
winrm i RestoreImage http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?SystemCreationClassName=DCIM_ComputerSystem +CreationClassName=DCIM_LCService +SystemName=DCIM:ComputerSystem +Name=DCIM:LCService -u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman -SkipCNCheck -SkipCACheck -encoding:utf-8 -a:basic @ {IPAddress="[IP ADDRESS]";ShareName="/SHARENAME";ShareType="2";Username="[USERNAME]";Password="[PASSWORD]";ImageName="[IMAGENAME]";Passphrase="[PASSPHRASE]";ScheduledStartTime="TIME_NOW"}
```

**Note:** The ShareName field should only be the folder exposed by the System to the network. Any sub folder information should be attached to the ImageName field. Otherwise, there can be connection issues when trying to locate or create the backup file.
Correct EXAMPLE: ShareName="/folder1";ImageName="subfolder/image_name"  Incorrect EXAMPLE: ShareName="/folder1/subfolder";ImageName="image_name"

OUTPUT:

```xml
restoreimage_output

job
  address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
  referenceparameters
  resourceuri = http://schemas.dell.com/wbem/wscim/1/cimschema/2/dcim_LifecycleJob
  selectorset
    selector: instanceid = JID_001293618214, __cimnamespace = root/dcim
  returnvalue = 4096
```

The response contains a reference to the job class that will provide the status of the operation. The return value is 4096 which indicates that the method operation is not yet complete.

18.3.4 Monitoring Import Status

Restore process may take up to 60 minutes depending on host System configuration. To monitor the backup status, get the instance of the corresponding job.

Replace [INSTANCE ID] with the actual jobid from Section 18.2.1, 18.2.2, or 18.2.3.

EXAMPLE:

```
winrm get
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LifecycleJob
?InstanceID=[INSTANCE ID]
-u:[USER]  -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck  -SkipCACheck
-a:basic encoding:utf-8
```

OUTPUT:

```xml
dcim_LifecycleJob
  instanceid = JID_001293618214
  jobstarttime = 00000101000000
  jobstatus = Restore In Progress
  jobuntiltime = TIME_NA
  message = Restoring Lifecycle Controller Firmware images
  messagetype = BAR081
  name = Restore:Image
  percentcomplete = 20
```

The status may be one of the following:

- Ready for Restore — Request has been received
- Restore In Progress — Restore process is currently in process
- Failed — Problem with the restore process, check message for more information
- Completed — Restore process has completed with no issues
iDRAC Configuration
This feature provides the ability to remotely list, get, and set the attributes on various monolithic and modular servers for the three Dell iDRAC classes through the command line.

- DCIM_iDRACCardEnumeration (19.1)
- DCIM_iDRACCardInteger (19.4)
- DCIM_iDRACCardString (19.6)

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

19.1 Listing the iDRAC Card Inventory-Enumeration Class
Enumerate the iDRACCardEnumeration class to list all the enumerate, integer, and string type iDRAC attributes.

Enumerate the iDRACCardEnumeration class with the following parameters and syntax:
EXAMPLE:

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardEnumeration
-u:[USER]  -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
DCIM_iDRACCardEnumeration
  AttributeDisplayName = Nic Enable
  AttributeName = Enable
  CurrentValue = Enabled
  DefaultValue = Enabled
  Dependency = null
  DisplayOrder = 0
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = NIC
  GroupID = NIC.1
  InstanceID = iDRAC.Embedded.1#NIC.1#Enable
  IsReadOnly = false
  PossibleValues = Disabled, Enabled
DCIM_iDRACCardEnumeration
  AttributeDisplayName = Virtual Media Attached
  AttributeName = Attached
  CurrentValue = Detached
  DefaultValue = Detached
  Dependency = null
  DisplayOrder = 0
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = VirtualMedia
  GroupID = VirtualMedia.1
```
InstanceID = iDRAC.Embedded.1#VirtualMedia.1#Attached
IsReadOnly = false
PossibleValues = Detached, Attached, Autoattach

DCIM_iDRACCardEnumeration
  AttributeDisplayName = IPv4 Enable
  AttributeName = Enable
  CurrentValue = Enabled
  DefaultValue = Enabled
  Dependency = null
  DisplayOrder = 0
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = IPv4
  GroupID = IPv4.1
  InstanceID = iDRAC.Embedded.1#IPv4.1#Enable
  IsReadOnly = false
  PossibleValues = Disabled, Enabled

DCIM_iDRACCardEnumeration
  AttributeDisplayName = User Admin IPMI LAN Privilege
  AttributeName = IpmiLanPrivilege
  CurrentValue = Administrator
  DefaultValue = NoAccess
  Dependency = null
  DisplayOrder = 0
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = Users
  GroupID = Users.3
  InstanceID = iDRAC.Embedded.1#Users.3#IpmiLanPrivilege
  IsReadOnly = false
  PossibleValues = User, Operator, Administrator, NoAccess

19.2 Getting an iDRAC Card Enumeration Instance

Use the following example to get an instance of the DCIM_iDRACCardEnumeration class instead of all the instances as shown in Section 19.1.

Get an iDRACCardEnumeration instance with the following parameters and syntax:

[INSTANCEID]: This is obtained from the enumeration in Section 19.1, which shows an example using iDRAC.Embedded.1#NIC.1#Enable as an instanceID.

EXAMPLE:

winrm g
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
19.3 Listing the iDRAC Card Inventory-Enumeration Class using groupID

Enumerate the DCIM_iDRACCardEnumeration class to list all the enumerate type iDRAC attributes using the group IDs of these groups: NIC, VirtualMedia, IPv4, and Users. To retrieve the attributes of the groups, set the GroupID to one of the following: NIC, VirtualMedia, IPv4, or Users.

Enumerate the iDRACCardEnumeration class using the following parameters and syntax:

**EXAMPLE:**
```
winrme
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardEnum-
eration
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-dialect:http://schemas.microsoft.com/wbem/wsman/1/WQL -filter:"select * from DCIM_iDRACCardEnumeration WHERE GroupID='NIC.1'"
```

The possible inputs for GroupID are:
- NIC.1
- VirtualMedia.1
- IPv4.1
- Users.3

**OUTPUT:**
```
DCIM_iDRACCardEnumeration
  AttributeDisplayName = Nic Enable
  AttributeName = Enable
  CurrentValue = Enabled
  DefaultValue = Enabled
  Dependency = null
  DisplayOrder = 0
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = NIC
  GroupID = NIC.1
  InstanceID = iDRAC.Embedded.1#NIC.1#Enable
  IsReadOnly = false
  PossibleValues = Disabled, Enabled
```
**GroupID = NIC.1**

*InstanceID = iDRAC.Embedded.1#NIC.1#Enable*

*IsReadOnly = false*

*PossibleValues = Disabled, Enabled*

**DCIM_iDRACCardEnumeration**

*AttributeDisplayName = Virtual Media Attached*

*AttributeName = Attached*

*CurrentValue = Attached*

*DefaultValue = Detached*

*Dependency = null DisplayOrder = 0*

*FQDD = iDRAC.Embedded.1*

*GroupDisplayName = VirtualMedia*

*GroupID = VirtualMedia.1*

*InstanceID = iDRAC.Embedded.1#VirtualMedia.1#Attached*

*IsReadOnly = false*

*PossibleValues = Detached, Attached, Autoattach*

**DCIM_iDRACCardEnumeration**

*AttributeDisplayName = IPv4 Enable*

*AttributeName = Enable*

*CurrentValue = Enabled*

*DefaultValue = Enabled*

*Dependency = null*

*DisplayOrder = 0*

*FQDD = iDRAC.Embedded.1*

*GroupDisplayName = IPv4*

*GroupID = IPv4.1*

*InstanceID = iDRAC.Embedded.1#IPv4.1#Enable*

*IsReadOnly = false*

*PossibleValues = Disabled, Enabled*

**DCIM_iDRACCardEnumeration**

*AttributeDisplayName = User Admin IPMI LAN Privilege*

*AttributeName = IpmiLanPrivilege*

*CurrentValue = Administrator*

*DefaultValue = NoAccess*

*Dependency = null*

*DisplayOrder = 0*

*FQDD = iDRAC.Embedded.1*

*GroupDisplayName = Users*

*GroupID = Users.3*

*InstanceID = iDRAC.Embedded.1#Users.3#IpmiLanPrivilege*

*IsReadOnly = false*

*PossibleValues = User, Operator, Administrator, NoAccess*
19.4 Applying the Attributes and Polling Job Completion

19.4.1 Changing iDRAC Values-ApplyAttributes() (Immediate)

Invoke the ApplyAttributes() method on the DCIM_iDRACCardService class to set or change the value of one or many enumerate type attributes. This method takes an xml file as input. The changes to the attributes are defined in this xml file. This method returns a JobID that is used as input in the next section (Section 19.3.2).

Invoke ApplyAttributes() method with the following parameters and syntax:

EXAMPLE:

```
winrm i ApplyAttributes
```

The input file DRACService_SetAttribute_groupEnumerate.xml is shown below.

```
<p:ApplyAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService">
  <p:Target>iDRAC.Embedded.1</p:Target>
  <p:AttributeName>NIC.1#Enable</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
  <p:AttributeName>NIC.1#Selection</p:AttributeName>
  <p:AttributeValue>Dedicated</p:AttributeValue>
  <p:AttributeName>NIC.1#Speed</p:AttributeName>
  <p:AttributeValue>100</p:AttributeValue>
  <p:AttributeName>NIC.1#Autoneg</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
  <p:AttributeName>NIC.1#Duplex</p:AttributeName>
  <p:AttributeValue>Full</p:AttributeValue>
  <p:AttributeName>NIC.1#DNSRegister</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
  <p:AttributeName>NIC.1#DNSDomainNameFromDHCP</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
  <p:AttributeName>NIC.1#VLanEnable</p:AttributeName>
  <p:AttributeValue>Disabled</p:AttributeValue>
  <p:AttributeName>VirtualMedia.1#Attached</p:AttributeName>
  <p:AttributeValue>Dettached</p:AttributeValue>
  <p:AttributeName>IPv4.1#Enable</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
  <p:AttributeName>IPv4.1#DHCPEnable</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
</p:ApplyAttributes_INPUT>
```
<p:AttributeName>IPv4.1#DNSFromDHCP</p:AttributeName><p:AttributeValue>Enabled</p:AttributeValue><p:AttributeName>Users.3#Enable</p:AttributeName><p:AttributeValue>Enabled</p:AttributeValue> ...

OUTPUT:
ApplyAttributes_OUTPUT

Job

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
SelectorSet
Selector: InstanceID = JID_001293705757, __cimnamespace = root/dcim

ReturnValue = 4096

19.4.2 Polling Job Completion

Use the Get() command to check the progress of the ApplyAttributes() method. It polls for job completion.

This method takes the InstanceID from the previous section (19.3.1) as input. The JobStatus value is either “Successful” or “Failed”. If the job failed, the Message value contains more detailed error information on the cause of the failure.

Run the Get() command on DCIM_LifecycleJob with the following parameters and syntax:

EXAMPLE:

winrm g http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LifecycleJob?
InstanceID=[INSTANCE_ID]
-u:[USER] p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

The input parameter is the InstanceID from the output of the ApplyAttributes() method. An example InstanceID is as follows: InstanceID = JID_001293705757
19.4.3 Set Attribute Verification
To verify the changes made to the attributes, enumerate the DCIM_iDRACCardEnumeration class. For more information, see Section 19.1.

**OUTPUT #2:**

```plaintext
DCIM_iDRACCardEnumeration
  AttributeDisplayName = Nic Enable
  AttributeName = Enable
  CurrentValue = Enabled
  DefaultValue = Enabled
  Dependency = null
  DisplayOrder = 0
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = NIC
  GroupID = NIC.1
  InstanceID = iDRAC.Embedded.1#NIC.1#Enable
  IsReadOnly = false
  PossibleValues = Disabled, Enabled

DCIM_iDRACCardEnumeration
  AttributeDisplayName = Virtual Media
  Attached AttributeName = Attached
  CurrentValue = Attached
  DefaultValue = Detached
  Dependency = null DisplayOrder = 0
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = VirtualMedia
  GroupID = VirtualMedia.1
  InstanceID = iDRAC.Embedded.1#VirtualMedia.1#Attached
  IsReadOnly = false
  PossibleValues = Detached, Attached, Autoattach

DCIM_iDRACCardEnumeration
  AttributeDisplayName = IPv4 Enable
  AttributeName = Enable
  CurrentValue = Enabled
```
212 Dell Lifecycle Controller Web Services Interface Guide for Windows

DefaultValue = Enabled
Dependency = null
DisplayOrder = 0
FQDD = iDRAC.Embedded.1
GroupDisplayName = IPv4
GroupID = IPv4.1
InstanceID = iDRAC.Embedded.1#IPv4.1#Enable
IsReadOnly = false
PossibleValues = Disabled, Enabled

DCIM_iDRACCardEnumeration
AttributeDisplayName = IPv4
AttributeName = IPv4
CurrentValue = Enabled
Dependency = null
DisplayOrder = 0
FQDD = iDRAC.Embedded.1
GroupDisplayName = IPv4
GroupID = IPv4.1
InstanceID = iDRAC.Embedded.1#IPv4.1#Enable
IsReadOnly = false
PossibleValues = Disabled, Enabled

GroupDisplayName = IPv4
GroupID = IPv4.1
InstanceID = iDRAC.Embedded.1#IPv4.1#Enable
IsReadOnly = false
PossibleValues = Disabled, Enabled

19.5 Listing the iDRAC Card Inventory-Integer Class
Enumerate the DCIM_iDRACCardInteger class to list all the integer type iDRAC attributes.
Enumerate the DCIM_iDRACCardInteger class with the following parameters and syntax:

EXAMPLE:

winrm e
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardInteg
er -u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_iDRACCardInteger
AttributeDisplayName = Vlan Priority
AttributeName = VLanPriority
CurrentValue = 0
Dependency = null
DisplayOrder = 0
FQDD = iDRAC.Embedded.1
GroupDisplayName = NIC
GroupID = NIC.1
InstanceID = iDRAC.Embedded.1#NIC.1#VLanPriority
IsReadOnly = false
LowerBound = 0
UpperBound = 7

DCIM_iDRACCardInteger
AttributeDisplayName = User Admin Privilege
AttributeName = Privilege
CurrentValue = 511
DefaultValue = 0
Dependency = null
DisplayOrder = 0
FQDD = iDRAC.Embedded.1
GroupDisplayName = Users
GroupID = Users.3
InstanceID = iDRAC.Embedded.1#Users.3#Privilege
IsReadOnly = false
LowerBound = 0
UpperBound = 511

19.6 Listing the iDRAC Card Inventory-Integer Class using groupID

Enumerate the DCIM_iDRACCardInteger class to list all the integer type iDRAC attributes using the group IDs of these groups: NIC and Users. To retrieve the attributes of the groups, set the GroupID to one of the following: NIC or Users.

All the iDRAC attributes of type integer that are part of a given Group (NIC and Users) are retrieved. In order to do this, “GroupID” needs to be set to one of the following: NIC or Users.

Enumerate the iDRACCardInteger class with the following parameters and syntax:

EXAMPLE:

```bash
```

The possible inputs for GroupID are:

NIC.1
Users.3

OUTPUT:

DCIM_iDRACCardInteger
AttributeDisplayName = VLan Priority
AttributeName = VLanPriority
CurrentValue = 1
DefaultValue = 0
Dependency = null
DisplayOrder = 0
FQDD = iDRAC.Embedded.1
GroupDisplayName = NIC
GroupID = NIC.1
InstanceID = iDRAC.Embedded.1#NIC.1#VLanPriority
IsReadOnly = false
LowerBound = 0
UpperBound = 7

DCIM_iDRACCardInteger
    AttributeDisplayName = User Admin Privilege
    AttributeName = Privilege
    CurrentValue = 511
    DefaultValue = 0
    Dependency = null
    DisplayOrder = 0
    FQDD = iDRAC.Embedded.1
    Group DisplayName = Users
    GroupID = Users.3
    InstanceID = iDRAC.Embedded.1#Users.3#Privilege
    IsReadOnly = false
    LowerBound = 0
    UpperBound = 511

19.7 Listing the iDRAC Card Inventory-String Class

Enumerate the DCIM_iDRACC ardString class to list all the string type iDRAC attributes.

Enumerate the iDRACCardString class with the following parameters and syntax:

EXAMPLE:

winrm e
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardString
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_iDRACCardString
    AttributeDisplayName = DNS RAC Name
    AttributeName = DNSRacName
    CurrentValue = idrac
    DefaultValue
    Dependency = null
    DisplayOrder = 0
    FQDD = iDRAC.Embedded.1
    GroupDisplayName = NIC
    GroupID = NIC.1
    InstanceID = iDRAC.Embedded.1#NIC.1#DNSRacName
19.8 Listing the iDRAC Card Inventory-String Class using groupId

Enumerate the DCIM_iDRACCardString class to list all the string type iDRAC attributes using the group IDs of these groups: NIC, IPv4, and Users. To retrieve the attributes of the groups, set the GroupID to one of the following: NIC, IPv4, or Users.

Invoke dracgetgroupid_string with the following parameters and syntax:

**EXAMPLE:**

```
```
The possible inputs for GroupID are:

- NIC.1
- IPv4.1
- Users.3

**OUTPUT:**

DCIM_iDRACCardString
- AttributeDisplayName = DNS RAC Name
- AttributeName = DNSRacName
- CurrentValue = IDRAC
- DefaultValue
- Dependency = null
- DisplayOrder = 0
- FQDD = iDRAC.Embedded.1
- GroupDisplayName = NIC
- GroupID = NIC.1
- InstanceID = iDRAC.Embedded.1#NIC.1#DNSRacName
- IsReadOnly = false
- MaxLength = 63
- MinLength = 1

DCIM_iDRACCardString
- AttributeDisplayName = IP Address
- AttributeName = Address
- CurrentValue = 172.27.36.55
- DefaultValue = 192.168.0.120
- Dependency = null
- DisplayOrder = 0
- FQDD = iDRAC.Embedded.1
- GroupDisplayName = IPv4
- GroupID = IPv4.1
- InstanceID = iDRAC.Embedded.1#IPv4.1#Address
- IsReadOnly = false
- MaxLength = 16
- MinLength = 1

DCIM_iDRACCardString
- AttributeDisplayName = User Admin User Name
- AttributeName = UserName
- CurrentValue = dell3
- DefaultValue
- Dependency = null
- DisplayOrder = 0
- FQDD = iDRAC.Embedded.1
- GroupDisplayName = Users
- GroupID = Users.3
- InstanceID = iDRAC.Embedded.1#Users.3#UserName
19.9 Changing the iDRAC IP Change Notification

19.9.1 Getting the Current iDRAC IPChange State

Get the IPChangeNotifyPS attribute from the DCIM_LCAttribute class to display. The CurrentValue field
indicates the current status of this attribute.

**EXAMPLE:**

```plaintext
winrm get
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCAttribute
?InstanceID=DCIM_LCEnumeration:DHS3
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
DCIM_LCEnumeration
    AttributeName = IPChangeNotifyPS
    Caption = null
   CurrentValue = Off
    DefaultValue = Off
    Description = null
    ElementName = LC.emb.1
    InstanceID = DCIM_LCEnumeration:DHS3
    IsOrderedList = null
    IsReadOnly = true
    PendingValue = null
    PossibleValues = On, Off
    PossibleValuesDescription = null
```

19.9.2 Setting the iDRAC IPChange Notification-SetAttribute()

The SetAttribute() method is used to set the attribute IPChangeNotifyPS to “ON” or “OFF”. When set to
“ON”, a user notification is sent when the IP address is changed. While set to “OFF”, a user notification is
not sent.

Invoke SetAttribute() with the following syntax:

**EXAMPLE:**

```plaintext
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService+System
+CreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
```
The input file setattribute.xml is shown below:

```xml
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_LCService">
  <p:AttributeName>IPChangeNotifyPS</p:AttributeName>
  <p:AttributeValue>on</p:AttributeValue>
</p:SetAttribute_INPUT>
```

**OUTPUT:**

```
SetAttribute_OUTPUT
  ReturnValue = 0
```

To verify the changes after setattribute was executed, list the LC attributes as shown in Section 19.8.1.

19.10 Reset to Defaults keep Network and users

DCIM iDRACCard profile describes the iDRACResetCfg method details.

iDRACResetCfg method used to reset the iDRAC to defaults, but keep Nework details and user details depends on the input values.

Invoke iDRACResetCfg method with the following parameters and syntax:

**Force:** reset option

- 0 – Graceful reset; 1- Force reset

**Preserve:**

- 0 - Default (reset all configuration to default except network and users)
- 1 - All (reset all configuration to default including network and users)
- 2 - RC (reset all configuration to default including network preserve default user as root calvin)

**EXAMPLE:**

```bash
winrm i iDRACResetCfg
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService?__cimnamespace=root/dcim+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+Name=DCIM:iDRACCardService
-u:root -p:calvin
-r:https://[IP]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic @(Force="1";Preserve="1")
```

**OUTPUT:**

```
iDRACResetCfg_OUTPUT
  Message = iDRAC is successfully reset to factory default properties.
  MessageID = RAC066
  ReturnValue = 0
```
20 Remote Service Status

To get the remote service status, invoke the GetRemoteServicesAPIStatus() method in the class DCIM_LCService. This method is used to obtain the overall remote services API status that includes both the host System status as well as the Lifecycle Controller (Data Manager included) status. The overall rolled up status shall be reflected in the Status output parameter.

**Note:** The LCStatus output parameter value includes the status reported by the DMStatus output parameter in the GetRSSStatus() method. Thus, GetRSSStatus() method invocation is redundant.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

20.1 Getting Remote Service Status

**EXAMPLE:**

```
winrm i GetRemoteServicesAPIStatus
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService
+SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
GetRemoteServicesAPIStatus_OUTPUT

LCStatus = 0
Message = Lifecycle Controller Remote Services is ready.
MessageID = LC061
ReturnValue = 0
ServerStatus = 2
Status = 0
```

Details on each output parameter is described below:

<table>
<thead>
<tr>
<th>Output Parameter Name</th>
<th>Possible Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>0 (Ready)</td>
<td>Lifecycle Controller Remote Services is ready to accept any web services request.</td>
</tr>
<tr>
<td></td>
<td>1 (Not Ready)</td>
<td>Lifecycle Controller Remote Services is currently not ready to accept web services request. This could be because the instrumentation in iDRAC might be reloading /not_ready or server is in POST or performing scheduled provisioning requests or Lifecycle Controller Unified Server Configurator is in use.</td>
</tr>
<tr>
<td>MessageID</td>
<td>LC060</td>
<td>Lifecycle Controller Remote Services is not ready. Message for ID LC060</td>
</tr>
<tr>
<td></td>
<td>LC061</td>
<td>Lifecycle Controller Remote Services is ready. Message for ID LC060</td>
</tr>
<tr>
<td>Message</td>
<td>Lifecycle Controller Remote Services is not ready.</td>
<td>Message for ID LC060</td>
</tr>
<tr>
<td></td>
<td>Lifecycle Controller Remote Services is ready.</td>
<td>Message for ID LC060</td>
</tr>
<tr>
<td>ServerStatus</td>
<td>1. (Powered off)</td>
<td>Server is powered off</td>
</tr>
<tr>
<td></td>
<td>2. (In POST)</td>
<td>Server is performing normal POST operation</td>
</tr>
<tr>
<td></td>
<td>3. (Out of POST)</td>
<td>Server is out of POST</td>
</tr>
</tbody>
</table>
### 4. (Collecting System Inventory)
Server is currently executing UEFI Collect System Inventory On Restart application

### 7. (Lifecycle Controller Unified Server Configurator)
Server is executing UEFI Lifecycle Controller Unified Server Configurator application

### 1. Ready
Lifecycle Controller instrumentation is up to date and enabled

### 2. (Not Initialized)
Lifecycle Controller instrumentation is not initialized. The initialization operation may take up to a minute.

### 3. (Reloading Data)
Lifecycle Controller is currently refreshing its cache because of a recent configuration change. The reloading operation typically takes few seconds and could take up to few minutes to complete.

### 4. (Disabled)
Lifecycle Controller is disabled on the server. Lifecycle Controller can be enabled thru Remote Services or F2 iDRAC configuration.

### 5. (In Recovery)
Lifecycle Controller is in Recovery mode. Refer to iDRAC users guide on instructions on how to repair Lifecycle Controller.

### 6. (In Use)
Lifecycle Controller is being currently used by another process.

## 20.2 Repeating Remote Service Status
If you continue to get “Not Ready” remote service status, invoke the DeleteJobQueue() method with JID_CLEARALL job id to restart the remote service [LC1.5.x ONLY].

**EXAMPLE:**
```
winrm invoke DeleteJobQueue
cimv2/root/dcim/DCIM_JobService?CreationClassName=DCIM_JobService
+Name=JobService
+SystemName=Idrac +SystemCreationClassName=DCIM_ComputerSystem
@{JobID="JID_CLEARALL" }
-u: [USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman:443
-SkipCNcheck -SkipCACheck
-auth:basic encoding:utf-8
```

**OUTPUT:**
```
DeleteJobQueue_OUTPUT
Message = The specified job was deleted
MessageID = SUP020
ReturnValue = 0
```
### 21 System Information

The DCIM System Info Profile describes the properties and interfaces for executing System management tasks related to the management of the host System. The profile standardizes and aggregates the description for the platform’s basic properties into a System view representation and provides static methodology for the clients to query the System views without substantial traversal of the model.

**Profile and Associated MOFs:**


### 21.1 Listing the System Inventory-SystemView Class

The System view returns various information about the System, including the currently installed Lifecycle Controller version as shown below.

Enumerate the DCIM_SystemView class with the following parameters and syntax:

**EXAMPLE:**

```
winrm e cimv2/root/dcim/DCIM_SystemView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCAcheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
DCIM_SystemView
  AssetTag
  BIOSReleaseDate = 01/09/2012
  BIOSVersionString = 0.3.37
  BaseBoardChassisSlot = NA
  BatteryRollupStatus = 1
  BladeGeometry = 4
  BoardPartNumber = 0MX4YFX04
  BoardSerialNumber = CN13740184000Q
  CMCIP = null
  CPLDVersion = 1.0.0
  CPURollupStatus = 1
  ChassisName = Main System Chassis
  ChassisServiceTag = 7654321
  ChassisSystemHeight = 5
  ExpressServiceCode = 15608862073
  FQDD = System.Embedded.1
  FanRollupStatus = 3
  HostName
  InstanceID = System.Embedded.1
  LastSystemInventoryTime = 20120116145530.000000+000
  LastUpdateTime = 20120116124210.000000+000
  LicensingRollupStatus = 1
  LifecycleControllerVersion = 2.0.0
  Manufacturer = Dell Inc.
  MaxCPUSockets = 2
```
MaxDIMMSlots = 24
MaxPCIeSlots = 7
MemoryOperationMode = OptimizerMode
Model = PowerEdge T620
PSRollupStatus = 1
PlatformGUID = 3132334f-c0b7-3480-3510-00364c4c4544
PopulatedCPUSockets = 1
PopulatedDIMMSlots = 1
PopulatedPCIeSlots = 1
PowerCap = 336
PowerCapEnabledState = 3
PowerState = 2
PrimaryStatus = 3
RollupStatus = 3
ServerAllocation = null
ServiceTag = 7654321
StorageRollupStatus = 1
SysMemErrorMethodology = 6
SysMemFailOverState = NotInUse
SysMemLocation = 3
SysMemPrimaryStatus = 1
SysMemTotalSize = 2048
SystemGeneration = 12G Monolithic
SystemID = 1231
SystemRevision = 0
TempRollupStatus = 1
UUID = 4c4c4544-0036-3510-8034-b7c04f333231
VoltRollupStatus = 1
smbiosGUID = 44454c4c-3600-1035-8034-b7c04f333231
Sensor Information

The DCIM Sensors Profile describes the properties and interfaces for executing System management tasks related to the management of sensors within a System.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

22.1 Listing the Sensors Inventory-PSNumericSensor Class

Enumerate the DCIM_PSNumericSensor class with the following parameters and syntax:

EXAMPLE:
winrm e "cimv2/root/dcim/DCIM_PSNumericSensor"
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_PSNumericSensor
  BaseUnits = 6
  CreationClassName = DCIM_PSNumericSensor
  CurrentReading = 11
  CurrentState = Normal
  Description = Power Supply Power Consumption
  DeviceID = iDRAC.Embedded.1#PS1Current1
  ElementName = PS1 Current 1
  EnabledDefault = 2 EnabledState = 2
  HealthState = 5
  LowerThresholdCritical
  LowerThresholdNonCritical
  OperationalStatus = 2
  PossibleStates = Unknown
  PossibleStates = Fatal
  PossibleStates = Normal
  PossibleStates = Upper Fatal
  PossibleStates = Upper Critical
  PossibleStates = Upper Non-Critical
  PossibleStates = Lower Non-Critical
  PossibleStates = Lower Critical
  PrimaryStatus = 1
  RateUnits = 0
  RequestedState = 12
  Resolution = 1
  SensorType = 13
  SettableThresholds
  SupportedThresholds
  SystemCreationClassName = DCIM_ComputerSystem
SystemName = srv: System
TransitioningToState = 12
UnitModifier = -1
UpperThresholdCritical
UpperThresholdNonCritical
ValueFormulation = 2
DCIM_PSNumericSensor
BaseUnits = 7
CreationClassName = DCIM_PSNumericSensor
CurrentReading = 126
CurrentState = Normal
Description = System Power Consumption in Watts
DeviceID = iDRAC.Embedded.1#SystemBoardPwrConsumption
ElementName = System Board Pwr Consumption
EnabledDefault = 2
EnabledState = 2
HealthState = 5
LowerThresholdCritical
LowerThresholdNonCritical
OperationalStatus = 2
PossibleStates = Unknown
PossibleStates = Fatal
PossibleStates = Normal
PossibleStates = Upper Fatal
PossibleStates = Upper Critical
PossibleStates = Upper Non-Critical
PossibleStates = Lower Non-Critical
PossibleStates = Lower Critical
PrimaryStatus = 1
RateUnits = 0
RequestedState = 12
Resolution = 1
SensorType = 13
SettableThresholds = 1
SupportedThresholds = 1
SupportedThresholds = 3
SystemCreationClassName = DCIM_ComputerSystem
SystemName = srv: System
TransitioningToState = 12
UnitModifier = 0
UpperThresholdCritical = 1344
UpperThresholdNonCritical = 1232
ValueFormulation = 2
Managing Fiber Channel (FC) Configuration

The Fiber Channel Profile extends the management capabilities of referencing profiles by adding the capability to represent the configuration of fiber channel host bus adapters (FC HBA). The FC HBAs are modeled as views and attributes where there is a view for each individual controller and multiple attributes that allow FC HBA configuration.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

23.1 Listing the FC Inventory-Attribute Class

The FC Inventory contains the following attributes: DCIM_FCIAttribute (23.1), DCIM_FCStatistics (23.2), DCIM_FCString (23.3), DCIM_FCInteger (23.4), and DCIM_FCEnumeration (23.5).

Enumerate FCAtribute class ith the following parameters and syntax:

EXAMPLE:
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cimchema/2/root/dcim/DCIM_FCAtribute
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_FCString
  AttributeDisplayName
  AttributeName = DeviceName
  CurrentValue = QLogic QLE2662 16Gb FC Adapter
  Dependency
  FQDD = FC.Slot.3-1
  InstanceID = FC.Slot.3-1:DeviceName IsReadOnly = true
  MaxLength = 32
  MinLength = 0
  PendingValue
  ValueExpression

DCIM_FCInteger
  AttributeDisplayName
  AttributeName = LinkDownTimeout
  CurrentValue = 30000
  Dependency
  FQDD = FC.Slot.3-2
  InstanceID = FC.Slot.3-2:LinkDownTimeout
  IsReadOnly = false
  LowerBound = 1
  PendingValue
  UpperBound = 255000
23.2 Listing the FC Inventory-Statistics Class

If RT-CEM is disabled on the System, this method will return failure.

Enumerate FCStatistics class with the following parameters and syntax:

**EXAMPLE:**

```
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_FCStatistics
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
DCIM_FCStatistics
   FCIInvalidCRCs = 0
   FCLinkFailures = 0
   FCLossOfSignals = 0
   FCRxKBCount = 0
   FCRxSequences
   FCRxTotalFrames = 0
   FCTxKBCount = 0
   FCTSequences
   FCTxTotalFrames = 0
   FQDD = FC.Slot.2-1
   InstanceID = FC.Slot.2-1
   OSDiverState = 2
   PortSpeed = 2
   PortStatus = 3

DCIM_FCStatistics
   FCIInvalidCRCs = 0
   FCLinkFailures = 0
   FCLossOfSignals = 0
   FCRxKBCount = 0
   FCRxSequences
   FCRxTotalFrames = 0
   FCTxKBCount = 0
   FCTSequences
   FCTxTotalFrames = 0
   FQDD = FC.Slot.2-2
   InstanceID = FC.Slot.2-2
   OSDiverState = 2
   PortSpeed = 2
   PortStatus = 3
```
23.3 Listing the FC Inventory-String Class
Enumerate FCStatistics class with the following parameters and syntax:

**EXAMPLE:**
```
```

**OUTPUT:**
```
DCIM_FCString
    AttributeDisplayName
    AttributeName = DeviceName
    CurrentValue = QLogic QLE2662 16Gb FC Adapter
    Dependency
    FQDD = FC.Slot.3-1
    InstanceID = FC.Slot.3-1:DeviceName
    IsReadOnly = true
    MaxLength = 32
    MinLength = 0
    PendingValue ValueExpression
```

23.4 Listing the FC Inventory-Integer Class
Enumerate FCInteger class with the following parameters and syntax:

**EXAMPLE:**
```
```

**OUTPUT:**
```
DCIM_FCInteger
    AttributeDisplayName
    AttributeName = LinkDownTimeout
    CurrentValue = 30000
    Dependency
    FQDD = FC.Slot.3-2
    InstanceID = FC.Slot.3-2:LinkDownTimeout
    IsReadOnly = false
    LowerBound = 1
    PendingValue UpperBound = 255000
```
23.5 Listing the FC Inventory-Enumeration Class

Enumerate FCEnumeration class ith the following parameters and syntax:

**EXAMPLE:**

```
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_FCEnumeration
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```DCIM_FCEnumeration```
```
AttributeDisplayName
AttributeName = PortEnable
CurrentValue = Disabled
Dependency
FQDD = FC.Slot.4-1
InstanceID = FC.Slot.4-1:PortEnable
IsReadOnly = false
PendingValue
PossibleValues = Disabled
PossibleValues = Enabled
PossibleValuesDescription
```

23.6 Changing the FC Attributes-SetAttribute()

The SetAttribute() method can be used to change the FC configuration.

Invoke SetAttribute() with the following parameters and syntax:

**TARGET:** Obtained from the InstanceID field

**AttributeName:** Obtained from the AttributeName field

**AttributeValue:** Obtained from the PossibleValues field

**EXAMPLE:**

```
winrm i SetAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_FCService
+CreationClassName=DCIM_ComputerSystem
+SystemCreationClassName=DCIM_FCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:FCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:SetAttribute_FC.xml
```

The input file `SetAttribute_FC.xml` is shown below:

```xml
<p:SetAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_FCService">
  
  <p:Target>FC.Slot.2-2</p:Target>

  <p:AttributeName>PortSpeed</p:AttributeName>
```
23.7 Applying the Pending Values for FC-CreateTargetedConfigJob()

This method is called to apply the pending values created by the SetAttribute() and SetAttributes() methods. The System will automatically reboot depending on the ScheduledStartTime selected. Using the CreateTargetedConfigJob() jobID output with the job control section can be used to obtain its status.

Invoke CreateTargetedConfigJob() with the following parameters and syntax:

TARGET: This Parameter is the FQDD of the instances, obtained from the InstanceID field

RebootJobType: There are three options for rebooting the System.

1 = PowerCycle
2 = Graceful Reboot without forced shutdown
3 = Graceful reboot with forced shutdown

Note: When a user does not want to set a reboot type when creating a target job, users should comment out the RebootJobType in the input xml. User should not enter “0” or give no parameter at all in the input xml.

EXAMPLE:

```plaintext
gwinrm i CreateTargetedConfigJob
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_FCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_FCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:FCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:apply_pending_fc.xml
```

The input file apply_pending_fc.xml is shown below:

```xml
  <p:Target>FC.Slot.2-2</p:Target>
  <p:RebootJobType>2</p:RebootJobType>
  <p:ScheduledStartTime.TIME_NOW</p:ScheduledStartTime>
  <p:UntilTime>20151111111111</p:UntilTime>
</p:CreateTargetedConfigJob_INPUT>
```

OUTPUT:

When this method is executed, a jobid or an error message is returned. The status of this jobid can be checked within the job control provider in Section 10.
CreateTargetedConfigJob_OUTPUT
Job

Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
ResourceURI = http://schemas.dell.com/wbem/wscim/1/cim schema/2/DCIM_LifecycleJob
SelectorSet
    Selector: InstanceID = JID_001269609760, __cimnamespace = root/dcim
ReturnValue = 4096

23.8 Deleting the Pending Values for FC-DeletePendingConfiguration()

This method is called to cancel the pending values created by the SetAttribute() and SetAttributes() methods. The DeletePendingConfiguration() method cancels the pending configuration changes made before the configuration job is created with CreateTargetedConfigJob(). This method only operates on the pending changes prior to CreateTargetedConfigJob() being called. After the configuration job is created, the pending changes can only be canceled by calling DeleteJobQueue() in the Job Control profile.

Invoke CreateTargetedConfigJob() with the following parameters and syntax: Target: This parameter is the FQDD of the instances

EXAMPLE:

winrm i DeletePendingConfiguration
http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_FCService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_FCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:FCService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:Delete_Pending_fc.xml

The input file Delete_Pending_fc.xml is shown below:
<p:DeletePendingConfiguration_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_FCServic e">
  <p:Target>FC.Slot.2-2</p:Target>
</p:DeletePendingConfiguration_INPUT>

OUTPUT:
DeletePendingConfiguration_OUTPUT Message = The command was successful
MessageID = FC001
ReturnValue = 0
ReturnValue = 4096
23.9 Listing the FC Views

Enumerate FCView class with the following parameters and syntax:

EXAMPLE:

```
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_FCView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
DCIM_FCView
  Bus = 5
  ChipType = 8324, Rev. 01
  Device = 0
  DeviceName = QLogic QLE2662 16Gb FC Adapter - 2001000E1E099026
  EFIVersion = 5.30
  FCTapeEnable = 3
  FQDD = FC.Slot.3-1
  FabricLoginRetryCount = 0
  FabricLoginTimeout = 0
  FamilyVersion = 02.00.84
  FirstFCTargetLUN = 0
  FirstFCTargetWWPN = 00:00:00:00:00:00:00:00
  FramePayloadSize = 2048
  Function = 0
  HardZoneAddress = 0
  HardZoneEnable = 3
  InstanceID = FC.Slot.3-1
  LinkDownTimeout = 30000
  LinkStatus = 0
  LoopResetDelay = 5
  PCIDeviceID = 2031
  PortDownRetryCount = 30
  PortDownTimeout = 0
  PortLoginRetryCount = 8
  PortLoginTimeout = 30000
  PortNumber = 1
  PortSpeed = 2
  SecondFCTargetLUN = 0
  SecondFCTargetWWPN = 00:00:00:00:00:00:00:00
  VendorName
  VirtualWWN = 20:00:00:0E:1E:09:90:26
  VirtualWWPN = 20:01:00:0E:1E:09:90:26
  WWN = 20:00:00:0E:1E:09:90:26
  WWPN = 20:01:00:0E:1E:09:90:26
```
Support for HTTP and HTTPS Network Shares in 14G

The support for HTTP and HTTPS Network Shares got added in 14th Generation of iDRAC for most of the WSMAn APIs that deal with network shares. The following are the list of methods which are now supporting this new addition,

1. DCIM_LCService.ExportLCLog()
2. DCIM_LCService.ExportCompleteLCLog()
3. DCIM_LCService.ExportHWInventory()
4. DCIM_LCService.ExportFactoryConfiguration()
5. DCIM_LCService.ExportSystemConfiguration()
6. DCIM_LCService.ImportSystemConfiguration()
7. DCIM_LCService.ImportSystemConfigurationPreview()
8. DCIM_LCService.ExportCertificate()
9. DCIM_LCService.ExportPSADiagnosticResult()
10. DCIM_LCService.BackupImage()
11. DCIM_LCService.RestoreImage()
12. DCIM_LCService.TestNetworkShare()
13. DCIM_LCService.SetBackupSchedule()
14. DCIM_LicenseManagementService.ExportLicenseToNetworkShare()
15. DCIM_LicenseManagementService.ImportLicenseFromNetworkShare()
16. DCIM_BIOSCertService.ExportBootCertificate()
17. DCIM_BIOSCertService.ImportBootCertificate()
18. DCIM_SoftwareInstallationService.SetUpdateSchedule() (support since 13G, enhanced in 14G)
19. DCIM_SoftwareInstallationService.InstallFromURI()
20. DCIM_SoftwareInstallationService.InstallFromRepository()

We have added/updated the following Input parameters related to Network Share access for this feature in all the relevant WSMAn APIs mentioned above,

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>DataType</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ShareName</td>
<td>String</td>
<td>Optional for HTTP/HTTPS shares, this may be treated as the path to the directory containing the file.</td>
</tr>
<tr>
<td>ShareType</td>
<td>Uint16</td>
<td>Two new possible values has been added, 5 = HTTP, 6 = HTTPS</td>
</tr>
<tr>
<td>UserName</td>
<td>String</td>
<td>Username for Network share authentication</td>
</tr>
<tr>
<td>Password</td>
<td>String</td>
<td>Password for Network share authentication</td>
</tr>
<tr>
<td>IgnoreCertWarning</td>
<td>Uint16</td>
<td>Specifies if certificate warning is to be ignored when HTTPS is specified. Default is 1 (Off). Possible values are, 1 = OFF (Default Value), 2 = ON</td>
</tr>
<tr>
<td>ProxySupport</td>
<td>Uint16</td>
<td>It specifies if proxy is to be used or not. Possible values, 1 = Off (Default Value), 2 = On - use default proxy settings, 3 = On - use passed in parameters for proxy</td>
</tr>
<tr>
<td>ProxyType</td>
<td>Uint16</td>
<td>It specifies the proxy type of the proxy server. Possible Values, 0 = HTTP (Default Value), 1 = SOCKS</td>
</tr>
<tr>
<td>ProxyServer</td>
<td>String</td>
<td>Represents the IP Address of the proxy server</td>
</tr>
<tr>
<td>ProxyPort</td>
<td>String</td>
<td>Represents the proxy port for the proxy server</td>
</tr>
<tr>
<td>ProxyUname</td>
<td>String</td>
<td>Represents the username for proxy server</td>
</tr>
<tr>
<td>ProxyPasswd</td>
<td>String</td>
<td>Represents the password to login into the proxy server</td>
</tr>
</tbody>
</table>

1. Not applicable for InstallFromURI() method as this information is embedded into the URI input parameter
2. For SetUpdateSchedule(), this parameter is available by the name ProxyHostName
3. For SetUpdateSchedule(), this parameter is available by the name ProxyUserName
4. For SetUpdateSchedule(), this parameter is available by the name ProxyPassword

Please check the respective sections of the APIs for the actual command examples which may be modified in accordance to the given new parameters above
Support for file transfer without need of network share in 14G

WSMAN file transfer mechanism had introduced in 14th Generation of iDRAC for transferring a file from management application local folder to iDRAC local storage and vice-versa.

To perform this operation WSMAN introduces the following extrinsic methods /APIs.

- ImportData()
- ExportData()
- ClearTransferSession()

25.1 ImportData():

This method is used to import to idrac local, based on the input parameter type. Management application can import the file in series of chunks, if file size is large using this method. This method is generic, which can transfer any file. Current scope of this method is “Server Configuration Profile (SCP)” file.

Invoke “ImportData” method with the following input parameters.

FileType: Import file type
  1 – XML Config
  2 – Reserved

InSessionID: Empty for first transaction

ChunkSize: Chunk size for the packet being transmitted. Max size of the chunk is 65535Bytes

FileSize: First Packet will have File Size

TxfrDescriptor: Transfer descriptor value
  1 – Start of Transmit
  2 – Normal Transmission
  3 – End of Packet

PayLoadEncoding: Payload encoding format

PayLoad: Base-64 encoded payload value
  1 – Text
  2 – Base64

CRC: CRC for complete file, using “md5” algorithm for CRC calculation.

After completion of file transfer, management application invokes the existing method “ImportSystemConfiguration()” where share type is equal to local to perform System configuration profile updates.

The following methods are currently supporting to import files, where the share type is local:

<table>
<thead>
<tr>
<th>Methods</th>
<th>Input Parameter (Unit16)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCIM_iDRACCardService.ImportSystemConfiguration</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_iDRACCardService.ImportSystemConfigurationPreview</td>
<td>ShareType = 4 for local</td>
</tr>
</tbody>
</table>

EXAMPLE:

winrm i ImportData
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem +SystemName=DCIM:ComputerSystem +CreationClassName=DCIM_iDRACCardService +Name=DCIM:IDRACCardService
-u:<USERNAME> -p:<PASSWORD>
-r:https://<IPADDRESS>/wsman
25.2 **ExportData():**

The method is used to export the files from iDRAC to a local folder in the management application. For example, if the management application wants to export the SCP file, it can invoke the “ExportData” method defined in DCIM_iDRACCardService class, but before invoking the ExportData method, management applications must invoke the existing export methods with share type equal to local to get export file into the iDRAC local folder.

Currently, following export methods are supported for local share type:

<table>
<thead>
<tr>
<th>Methods</th>
<th>Input Parameter (Unit16)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCIM_LCService.ExportLCLog</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_LCService.ExportCompleteLCLog()</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_LCService.ExportHWInventory()</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_LCService.ExportFactoryConfiguration()</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_LCService.ExportSystemConfiguration()</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_LCService.SupportAssistCollection()</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_LCService.ExportPSADiagnosticsResult()</td>
<td>ShareType = 4 for local</td>
</tr>
<tr>
<td>DCIM_LCService.ExportVideoLog()</td>
<td>ShareType = 4 for local</td>
</tr>
</tbody>
</table>

After completion of the export method (mentioned in the table), management application needs to invoke the “ExportData” method with the following input parameter to get the export file to their local folder.

**InChunkSize:** The size of each block the management application expects to receive.

**SessionID:** Empty in the first packet.

**FileOffset:** Offset of the file. “FileOffset” will be “0” for the first chunk.

**TxDataSize:** Transmitted data size. “TxDataSize” will be “0” for the first chunk.

**FileType:** Export file type.

- 1 – SystemConfiguration file
- 2 – LC logs
- 3 – Hardware inventory file
- 4 – Factory configuration file
- 5 – Support assist collection file
- 6 – Boot video logs
Method response contains the following output parameters.

**FileSize**: Size of the exported file

**SessionID**: A unique session ID is generated and it will be used in the subsequent packets. This ensures that blocks are pertaining to the same file.

**TxfrDescriptor**: Transfer descriptor value

1. Start of Transmit
2. Normal Transmission
3. End of Packet

**RetFileOffset**: Current position (offset) of the exported file.

**RetTxDataSize**: Transmitted data size of the export file.

**PayloadEncoding**: Payload encoding format.

1. Text
2. Base64

**Payload**: iDRAC will send the payload as base64 encoded format.

**CRC**: CRC of entire export file. md5 checksum algorithm is used for CRC calculation, it's value is 128 bytes.

**EXAMPLE:**

```
winrm i ExportData
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?
SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+Name=DCIM:iDRACCardService
-u:<USERNAME> -p:<PASSWORD>
-r:https://<IPADDRESS>/wsman
-SkipCNcheck -SkipCACCheck -SkipRevocationCheck
-encoding:utf-8 -a:basic
-file:C:\Sreelakshmi\Project\14G\ConfigXML\ExportFile.xml
```

ExportFile.xml: Contains the following parameters.

```
<p:ExportData_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService">
  <p:FileType>“file type 1/2/3 etc”</p:FileType>
  <p:TxDataSize>“Transmitted data size”</p:TxDataSize>
  <p:FileOffset>“file offset ”</p:FileOffset>
  <p:InChunkSize>“chunk size in bytes”</p:InChunkSize>
  <p:InSessionID>“session ID”</p:InSessionID>
</p:ExportData_INPUT>
```

### 25.3 ClearTransferSession()

The method is used to delete the import/export files, which are stored in iDRAC local.

Invoke the “ClearTransferSession” method with the following input parameters.
**FileOperation:** It specifies the delete file, whether it is import or export.

1 – Import
2 – Export
3 – Both

**FileType:** It specifies the file type.

0 – All
1 – System Configuration Profile file
2 – LC logs
3 – Hardware inventory file
4 – Factory configuration file
5 – SupportAssist collection file
6 – Boot video logs
7 – Diagnostics
8 – LC full logs
9 – Crash video logs.

**EXAMPLE:**

```
winrm i ClearTransferSession
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService
+SystemCreationClassName=DCIM_ComputerSystem
 +SystemName=DCIM:Computer
 +CreationClassName=DCIM_iDRACCardService
 +Name=DCIM:iDRACCardService
 -u:<USERNAME> -p:<PASSWORD>
 -r:https://<IPADDRESS>/wsman
 -SkipCNCheck -SkipCACheck -SkipRevocationCheck
 -encoding:utf-8 -a:basic @{FileOperation="2";FileType="3"}
```
GroupManager Configuration

The DCIM iDRAC Card Profile describes the group manager configuration.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

26.1 Enabling GroupManager Feature

The iDRACCardService.ApplyAttribute() method can be used to change the GroupManager.1#Status attribute.

EXAMPLE:

winrm i ApplyAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService
+SystemCreationClassName=DCIM_ComputerSystem
+Name=DCIM:iDRACCardService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:ApplyAttribute_iDRAC.xml

The input file ApplyAttribute_iDRAC.xml is shown below:

```xml
<p:ApplyAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService">
   <p:Target>iDRAC.Embedded.1</p:Target>
   <p:AttributeName>GroupManager.1#Status</p:AttributeName>
   <p:AttributeValue>Enabled</p:AttributeValue>
</p:ApplyAttribute_INPUT>

OUTPUT:

ApplyAttribute_OUTPUT
   Message = The command was successful
   MessageID = RAC001
   RebootRequired = No
   ReturnValue = 0
   SetResult = Set CurrentValue
```

26.2 To View GroupName and GroupUUID Details

Two attribute GroupManager.1#GroupName and GroupManager.1#GroupUUID available in DCIM_iDracCardString to get the GroupName and Group UUID details.

EXAMPLE:

winrm g
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDracCardString
+InstanceID=iDRAC.Embedded.1#GroupManager.1#GroupName
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
OUTPUT:

DCIM_iDracCardString
  AttributeDisplayName = Group Name
  AttributeName = GroupName
  CurrentValue = <john&john>
  DefaultValue = null
  Dependency = null
  DisplayOrder = 2
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = Group Manager
  GroupID = GroupManager.1
  InstanceID = iDRAC.Embedded.1#GroupManager.1#GroupName
  IsReadOnly = true
  MaxLength = 32
  MinLength = 0
  PendingValue = null

EXAMPLE:

winrm g
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDracCardString?InstanceID=iDRAC.Embedded.1#GroupManager.1#GroupUUID
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCAcheck
-encoding:utf-8 -a:basic

OUTPUT:

DCIM_iDracCardString
  AttributeDisplayName = Group UUID
  AttributeName = GroupUUID
  CurrentValue = 96FB7C5AD83223B549A9A32EE168ED69
  DefaultValue = null
  Dependency = null
  DisplayOrder = 3
  FQDD = iDRAC.Embedded.1
  GroupDisplayName = Group Manager
  GroupID = GroupManager.1
  InstanceID = iDRAC.Embedded.1#GroupManager.1#GroupUUID
  IsReadOnly = true
  MaxLength = 32
  MinLength = 0
  PendingValue = null
26.3 To Join Group with existing master group

Client can invoke the DCIM_iDRACCardService.JoinGroup() method available in iDRAC Card Profile to join with existing master group.

EXAMPLE:

```bash
winrm i JoinGroup
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+Name=DCIM_iDRACCardService -u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic -file:join_group.xml
```

The input file `join_group.xml` is shown below:

```xml
<p:JoinGroup_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService">
  <p:GroupName>john</p:GroupName>
  <p:GroupPasscode>John</p:GroupPasscode>
  <p:GroupUUID>96FB7C5AD83223B549A9A32EE168ED69</p:GroupUUID>
  <p:CloneConfiguration>1</p:CloneConfiguration>
</p:JoinGroup_INPUT>
```

OUTPUT:

```text
JoinGroup_OUTPUT
Message = The server GMGR026 has successfully joined the local iDRAC group
+john&john
MessageArguments = GMGR026,+john&john
MessageID = GMGR0007
ReturnValue = 0
```

26.4 To Remove Self from Group

Client can invoke the DCIM_iDRACCardService.RemoveSelf() method available in iDRAC Card Profile to remove from the master.

EXAMPLE:

```bash
winrm i RemoveSelf
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+Name=DCIM_iDRACCardService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file:remove_self.xml
```

The input file `remove_self.xml` is shown below:

```xml
<p:RemoveSelf_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService">
</p:RemoveSelf_INPUT>
```
<p:GroupName>john</p:GroupName>

</p:RemoveSelf_INPUT>

OUTPUT:

RemoveSelf_OUTPUT

Message = iDRAC successfully removed itself from the local group <john&john>.
MessageArguments = <john&john>
MessageID = GMGR0006
ReturnValue = 0

26.5 To Delete the Master Group

Client can invoke the DCIM_iDRACCardService.DeleteGroup() method available in iDRAC Card Profile to Delete the group.

EXAMPLE:

winrm i DelteGroup
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+Name=DCIM:iDRACCardService -u:[USER] -p:[PASSWORD] -r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic -file:delete_group.xml

The input file delete_group.xml is shown below:

<p:DeleteGroup_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService"

<p:GroupName>john</p:GroupName>

</p:DeleteGroup_INPUT>

OUTPUT:

DeleteGroup_OUTPUT

Message = A new group job GMGR027001 is successfully created for DeleteGroup. The job progress can be viewed in the Group Manager job view.
MessageArguments = GMGR027001, DeleteGroup
MessageID = GMGR0033
ReturnValue = 0
27 System configure Lock down mode

If System configure lock down mode enabled then the System configuration features are blocked, except for the power related operation.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

27.1 Enabling the System Configure Lock down mode

The iDRACCardService.ApplyAttribute() method can be used to change the Lockdown.1#SystemLockdown attribute.

Lockdown.1#SystemLockdown is the enumeration attribute and it is available in DCIM_IDRACCardEnumeration, it can be enabled or Disabled.

EXAMPLE:
winrm i ApplyAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+Name=DCIM:iDRACCardService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file: ApplyAttribute_iDRAC.xml

The input file ApplyAttribute_iDRAC.xml is shown below:

```xml
<p:ApplyAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService">
  <p:Target>iDRAC.Embedded.1</p:Target>
  <p:AttributeName> Lockdown.1#SystemLockDown </p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
</p:ApplyAttribute_INPUT>
```

OUTPUT:

ApplyAttribute_OUTPUT

Message = The command was successful
MessageID = RAC001
RebootRequired = No
ReturnValue = 0
SetResult = Set CurrentValue
28 Connection View
The DCIM Switch Connection Profile describes the Connection View details of iDRAC dedicated port and shared LOM and configuration.
Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

28.1 Enabling the Connection View feature
SwitchConnectionView.1#Enable is the enumeration attribute and it is available in DCIM_IDRACCardEnumeration, it can be enabled or Disabled.
EXAMPLE:
winrm i ApplyAttribute
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_iDRACCardService?SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_iDRACCardService
+Name=DCIM:iDRACCardService -u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file: ApplyAttribute_iDRAC.xml
The input file ApplyAttribute_iDRAC.xml is shown below:
<p:ApplyAttribute_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_iDRACCardService">
  <p:Target>iDRAC.Embedded.1</p:Target>
  <p:AttributeName>SwitchConnectionView.1#Enable</p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
</p:ApplyAttribute_INPUT>
OUTPUT:
ApplyAttribute_OUTPUT
  Message = The command was successful
  MessageID = RAC001
  RebootRequired = No
  ReturnValue = 0
  SetResult = Set CurrentValue

28.2 Connection View Instances:
DCIM_iDRACSwitchConnectionView class will list all the instance of the Connection View details.
EXAMPLE:
winrm e
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_SwitchConnectionView
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
**OUTPUT:**

```
DCIM_SwitchConnectionView
   FQDD = iDRAC.Embedded.1
   InstanceID = iDRAC.Embedded.1
   StaleData = 0
   SwitchConnection = 5c:26:0a:ae:5b:21
   SwitchPortConnection = 1/0/41

DCIM_SwitchConnectionView
   FQDD = NIC.Integrated.1-1-1
   InstanceID = NIC.Integrated.1-1-1
   StaleData = 0
   SwitchConnection
   SwitchPortConnection
```

### 28.3 Refresh the connection view instance

DCIM_SwitchConnectionView instance are auto refreshed by 5 min if the user want to refresh it immediately ServerPortConnectionRefresh() method available in DCIM_SwitchConnectionService.

**EXAMPLE:**

```
winrm i ServerPortConnectionRefresh
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_SwitchConnectionService?SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_SwitchConnectionService
+Name=DCIM:SwitchConnectionService
-u:[USER] -p:[PASSWORD]
-r:https://[IPADDRESS]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

```
ServerPortConnectionRefresh_OUTPUT
   Message = The refresh operation of switch connection and switch port connection information of all network ports is completed successfully.
   MessageID = RAC089
   ReturnValue = 0
```
29 SecureBoot Configuration

The DCIM BIOS and Boot Management Profile describes the secure boot configuration.

Profile and Associated MOFs:
http://www.delltechcenter.com/page/DCIM.Library.Profile

29.1 Enabling Secure Boot configuration

To enable or disable the secure boot, management applications can use the DCIM_BIOSService.SetAttribute() method from Dell_BIOSandBOOT management profile to set the BIOS.Setup.1-1:SecureBoot attribute and create a job using DCIM_BIOSService.CreateTargetedConfigJob() method to execute this change. This enumeration attribute can be set to Enabled or Disabled.

EXAMPLE:

```
winrm i setattributes
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/dcim_biosservice?
_cimnamespace=root/dcim+Systemcreationclassname=dcim_computerSystem
+SystemName=dcim:computerSystem
+CreationClassName=dcim_biosservice
+Name=dcim:biosservice -u:[USER] -p:[PASSWORD]
-r:https://[iDRAC_IP]/wsman
-SkipCNcheck -skipcachck -skiprevocationCheck
-encoding:utf-8 -a:basic
-file: setattributes_bios.xml
```

The input file setattributes_bios.xml is shown below:

```
<p:SetAttributes_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_biosservice">
  <p:Target> bios.setup.1-1 </p:Target>
  <p:AttributeName>secureboot </p:AttributeName>
  <p:AttributeValue>Enabled</p:AttributeValue>
</p:SetAttributes_INPUT>
```

```
winrm i createtargetedconfigjob
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/root/dcim/dcim_biosservice?
_cimnamespace=root/dcim+Systemcreationclassname=dcim_computerSystem
+SystemName=dcim:computerSystem
+CreationClassName=dcim_biosservice
+Name=dcim:biosservice
-u:[USER] -p:[PASSWORD]
-r:https://[iDRAC_IP]/wsman
-SkipCNcheck -skipcachck -skiprevocationCheck
-encoding:utf-8 -a:basic
-file: create_target_config.xml
```

The input file create_target_config.xml is shown below:

```
<p:createTargetedConfigJob_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOSService">
  <p:Target> bios.setup.1-1 </p:Target>
  <p:rebootjobtype>1</p:rebootjobtype>
  <p:scheduledstarttime>time_now</p:scheduledstarttime>
</p:createTargetedConfigJob_INPUT>
```
29.2 View Secure boot Certificate

Management application can enumerate the DCIM_BIOSCertView class to see all the stored Secure Boot Certificate information.

EXAMPLE:

```
winrm e http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/dcim_bioscertview
-u:[USER] -p:[PASSWORD]
-r:https://[iDRAC_IP]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:

```
DCIM_BIOSCertView
CertificateSubType = 1
CertificateType = 1
InstanceID = iDRAC.Embedded.1#CustSecbootpolicy.1
Issuer = issuer= /DC=com/DC=dell/CN=Dell Inc. Issuing CA 1
SerialNumber = 1
Subject = subject= /C=US/ST=TX/L=Round Rock/O=Dell Inc./OU=Products/CN=Dell UEFI PK PowerEdge
ValidFrom = Feb 21 19:01:31 2014 GMT
ValidTo = Feb 21 19:31:31 2017 GMT
```

29.3 Import boot Certificate

The DCIM BIOS and Boot Management Profile describes the secure boot configuration. This method used to import the secure boot certificate in to iDRAC. The secure boot certificate can be located on a NFS/CIFS/HTTP/HTTPS share. When this method is successfully executed, certificate will store in iDRAC, management application has to restart the host server for the changes to take effect.

Invoke ImportBootCertificate() with the following parameters and syntax:

**IPAddress**: IP address of the machine hosting the network share

**ShareType**: Type of share.

- NFS=0, CIFS=2, HTTP=5, HTTPS=6

**ShareName**: Name of the CIFS share or full path to the NFS share. This may be treated as the path of the directory containing the file.

**Username**: The username to be used to access the file.

**Password**: The password to be used to access the file.

**Workgroup**: Name of the workgroup for CIFS Share authentication.

**FileName**: file name of the certificate to be import from the network share.

**CertificateType**: Type of the certificate.

- 1=PK, 2 = KeK, 3 = DB, 4 = DBX.

**CertificateSubType**: Value of the certificate sub type.

- 1 = Certificate, 2 = SHA256, 3 = SHA384, 4 = SHA512.

Proxy parameters are used when the share type is HTTPS.

**ProxySupport**: It specifies if the proxy is to be used or not.

- 1 = OFF, 2 = On-use user default proxy setting, 3 = On-Use passed in parameters for proxy.
IgnoreCertWarning: It specifies if certificate warning is to be ignored when the share type is HTTPS.
   1 = Off, 2 = On
ProxyType: It specifies the proxy type of the proxy server
   0 = HTTP, 1 = SOCKS
ProxyServer: It specifies the IP address of the proxy server.
ProxyUname: It specifies the username of the proxy server.
ProxyPasswd: It specifies the password of the proxy server.
ProxyPort: It specifies the port of the proxy server.

EXAMPLE:
ShareType = 0
winrm invoke ImportBootCertificate
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BIOSCertService?
SystemCreationClassName=DCIM_ComputerSystem +CreationClassName=DCIM_BIOSCertService +SystemName=DCIM:ComputerSystem +Name=DCIM:BIOSCertService
-file:import_boot_cert.xml
-r:https://[iDRAC_IP]
-u:[USER] -p:[PASSWORD]
-SkipCNcheck -SkipCACheck -SkipRevocationCheck
-encoding:utf-8 -a:basic -format:pretty

The input file import_boot_cert.xml is shown below:

```xml
<p:importbootcertificate_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOSCertService"
>
   <p:certificatetype>1</p:certificatetype>
   <p:CertificateSubType>1</p:CertificateSubType>
   <p:FileName>pk.cer</p:FileName>
   <p:Username>NetworkShareUserName</p:Username>
   <p:Password>NetworkSharePassword</p:Password>
   <p:ShareType>0</p:ShareType>
   <p:ShareName>ShareName</p:ShareName>
   <p:IPAddress>ShareIP</p:IPAddress>
</p:importbootcertificate_INPUT>
```

OUTPUT:
ImportBootCertificate_OUTPUT
   Message = The SecureBoot Certificate Import operation is successfully scheduled. Restart the host server for the changes to take effect.
   MessageID = SWC9010
   ReturnValue = 0
29.4 Export boot Certificate

The DCIM BIOS and Boot Management Profile describes the secure boot configuration.

This method used to export the secure boot certificate from the iDRAC to the network share either NFS/CIFS/HTTPS.

Invoke ExportBootCertificate() with the following parameters and syntax:

**IPAddress**: IP address of the machine hosting the network share

**ShareType**: Type of share.
- NFS=0, CIFS=2, HTTP=5, HTTPS=6

**ShareName**: Name of the CIFS share or full path to the NFS share. This may be treated as the path of the directory containing the file.

**Username**: The username to be used to access the file.

**Password**: The password to be used to access the file.

**Workgroup**: Name of the workgroup for CIFS Share authentication.

**FileName**: File name of the certificate to be import from the network share.

**CertificateType**: Type of the certificate.
- 1=PK, 2 = KeK , 3 = DB, 4 = DBX.

**CertificateSubType**: Value of the certificate sub type.
- 1 = Certificate, 2 = SHA256 , 3 = SHA384, 4 = SHA512.

**FileName**: File Name to be stored in the network share.

**CertificateIdentifier**: Unique identifier of the particular certificate.

Proxy parameters are used when the share type is HTTPS.

**ProxySupport**: It specifies if the proxy is to be used or not.
- 1 = OFF, 2 = On-use user default proxy setting, 3 = On-Use passed in parameters for proxy.

**IgnoreCertWarning**: It specifies if certificate warning is to be ignored when the share type is HTTPS.
- 1 = Off, 2 = On

**ProxyType**: It specifies the proxy type of the proxy server
- 0 = HTTP , 1 = SOCKS

**ProxyServer**: It specify the IP address of the proxy server.

**ProxyUname**: It specify the username of the proxy server.

**ProxyPasswd**: It specify the password of the proxy server.

**ProxyPort**: It specify the port of the proxy server.

**EXAMPLE**: 

```
ShareType = 6 (HTTPS)
winrm i ExportBootCertificate
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_BIOSCertService
+Name=DCIM:BIOSCertService
-u:[USER] -p:[PASSWORD]
-r:https://[iDRAC_IP_Address]/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
-file: export_boot_cert.xml
```
The input file `export_boot_cert.xml` is shown below:

```xml
<p:exportbootcertificate_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOSCertService">
  <p:certificatetype>3</p:certificatetype>
  <p:CertificateSubType>1</p:CertificateSubType>
  <p:FileName>DB.cert</p:FileName>
  <p:Username>NetworkShareUserName</p:Username>
  <p:Password>NetworkSharePassword</p:Password>
  <p:ShareType>6</p:ShareType>
  <p:ShareName>ShareName</p:ShareName>
  <p:IPAddress>ShareIP</p:IPAddress>
</p:exportbootcertificate_INPUT>

OUTPUT:
ExportBootCertificate_OUTPUT
Message = The SecureBoot Certificate Export operation is successfully completed.
MessageID = SWC9011
ReturnValue = 0

29.5 **Delete boot Certificate**

The Dell BIOS and Boot Management Profile describes the secure boot configuration. DeleteBootCertificate method used to delete the secure boot certificate stored in iDRAC.

Invoke `DeleteBootCertificate()` with the following parameters and syntax:

- **CertificateType:** It specifies the Certificate Type.
  - 0 – ALL, 1 = PK, 2 = KeK, 3=DB, 4 = DBX.
- **CertificateSubType:** It specifies the Certificate sub type.
  - 1 = Certificate, 2 = SHA256, 3 = SHA384, 4 = SHA512.
- **CertificateIdentifier:** Unique identifier of the certificate.

**EXAMPLE:**

```bash
winrm invoke deletebootcertificate
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_BiosCertService
+CreationClassName=dcim_bioscertservice
+SystemName=dcim:computerSystem
+Name=dcim:bioscertservice
-file:delete_boot_Cert.xml
-r:https://[IDRAC_IPADDRESS]/wsman
-u:[USER] -p:[PASSWORD]
-SkipCNcheck -SkipCACheck -SkipRevocationCheck
-encoding=utf-8 -a:basic -format:pretty
```
The input file delete_boot_cert.xml is shown below:

```xml
<p:Deletebootcertificate_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOSCertService">
  <p:certificatetype>0</p:certificatetype>
</p:Deletebootcertificate_INPUT>
```

**OUTPUT:**

```
DeleteBootCertificate_OUTPUT
Message = The SecureBoot Certificate DeleteAll operation is successfully completed. Restart the host server for the changes to take effect.
MessageID = SWC9007
ReturnValue = 0
```

## 29.6 Reset boot Certificate

The Dell BIOS and Boot Management Profile describes the secure boot configuration.

**ResetBootCertificate** method used to reset the secure boot certificate stored in iDRAC, mean it will copy all the certificate from standard mode to custom mode.

Invoke `ResetBootCertificate()` with the following parameters and syntax:

**CertificateType:** It Specifies the Certificate Type.

- 0 – ALL, 1 = PK, 2 = KeK, 3=DB, 4 = DBX.

**EXAMPLE:**

```
winrm invoke resetbootcertificate
http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/dcim_bioscertservice?
+CreationClassName=dcim_computerSystem
+SystemName=dcim:computerSystem
+Name=dcim:bioscertservice
-file:reset_boot_Cert.xml
-r:https://[iDRAC_IP_Address]/wsman
-u:[USER] -p:[PASSWORD]
-SkipCNcheck -SkipCACheck -SkipRevocationCheck
-encoding:utf-8 -a:basic -format:pretty
-file:reset_boot_cert.xml
```

The input file reset_boot_cert.xml is shown below:

```xml
<p:resetbootcertificate_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cimschema/2/root/dcim/DCIM_BIOSCertService">
  <p:certificatetype>1</p:certificatetype>
</p:resetbootcertificate_INPUT>
```

**OUTPUT:**

```
ResetBootCertificate_OUTPUT
Message = The SecureBoot Certificate Reset operation is successfully completed. Restart the host server for the changes to take effect.
MessageID = SWC9008
ReturnValue = 0
```
30 SupportAssist

30.1 SupportAssistRegister

User needs to register server with Dell to avail SupportAssist features. User needs to provide FirstName, LastName, Phone Number, Alternate Number, Email address, Company name, Service Address. User needs to accept EULA before executing the SupportAssist registration method. On successful execution, SupportAssistRegister method returns a Job Identifier which user can monitor for the status of registration.

EXAMPLE:

```plaintext
winrm i
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_LCService
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://10.94.225.48/wsman
-SkipCNCheck -SkipCACheck
-encoding:utf-8 -a:basic -file:SARegister.xml
```

Content of SARegister.xml file is as below:

```xml
    <p:PrimaryFirstName>PrimaryFirstName</p:PrimaryFirstName>
    <p:PrimaryLastName>PrimaryLastName</p:PrimaryLastName>
    <p:PrimaryPhoneNumber>987-654-3210</p:PrimaryPhoneNumber>
    <p:PrimaryEmail>PrimaryEmail@CompanyName.com</p:PrimaryEmail>
    <p:CompanyName>CompanyName</p:CompanyName>
    <p:Street1>Street1</p:Street1>
    <p:Street2>Street2</p:Street2>
    <p:City>City</p:City>
    <p:State>State</p:State>
    <p:Country>Country</p:Country>
    <p:Zip>Zip</p:Zip>
    <p:PrimaryAlternateNumber>987-654-3211</p:PrimaryAlternateNumber>
    <p:SecondaryFirstName>SecondaryFirstName</p:SecondaryFirstName>
    <p:SecondaryLastName>SecondaryLastName</p:SecondaryLastName>
    <p:SecondaryPhoneNumber>987-654-3212</p:SecondaryPhoneNumber>
    <p:SecondaryAlternateNumber>987-654-3213</p:SecondaryAlternateNumber>
    <p:SecondaryEmail>SecondaryEmail@CompanyName.com</p:SecondaryEmail>
    <p:ProxyPort>1234</p:ProxyPort>
    <p:ProxyPassword>ProxyPassword</p:ProxyPassword>
    <p:ProxyUserName>ProxyUserName</p:ProxyUserName>
    <p:ProxyHostName>ProxyHostName</p:ProxyHostName>
</p:SupportAssistRegister_INPUT>
OUTPUT:
SupportAssistRegister_OUTPUT
Job
   EndpointReference
   Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
   ReferenceParameters
   SelectorSet
       Selector: InstanceID = JID_904041728335, __cimnamespace = root/dcim
   ReturnValue = 4096

30.2 SupportAssistAcceptEULA
User needs to accept End User License Agreement (EULA) before registering for SupportAssist. User can accept EULA by executing the below command.

```
+SystemName=DCIM:ComputerSystem
+ClassName=DCIM_LCService
-u:root -p:calvin
-r:https://10.94.225.48/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:
SupportAssistAcceptEULA_OUTPUT
Message = The SupportAssist End User License Agreement (EULA) is accepted by iDRAC user root via iDRAC interface WSMAN.
MessageArguments = root, WSMAN
MessageID = SRV074
ReturnValue = 0

30.3 SupportAssistGetEULAStatus
User can get the information about EULA acceptance by executing the below command.

```
+SystemName=DCIM:ComputerSystem
+ClassName=DCIM_LCService
-u:root -p:calvin
-r:https://10.94.225.48/wsman
-SkipCNcheck
-skipCACheck
-encoding:utf-8 -a:basic
```
OUTPUT:
SupportAssistRegister_OUTPUT
Job
    EndpointReference
    Address = http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    ReferenceParameters
    SelectorSet
        Selector: InstanceID = JID_904041728335, __cimnamespace = root/dcim
    ReturnValue = 4096

30.4 SupportAssistSetAutoCollectSchedule
Once user has registered the server with SupportAssist, this method can be used to update the SupportAssist auto collections schedule. By default, monthly schedule is created for the 1st Sunday (between 12 a.m. – 5 p.m.) for auto collections post successful registration.

User can specify the schedule to be weekly, montly or quarterly for auto collections.

Command: winrm i SupportAssistSetAutoCollectSchedule
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_LCService
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://10.94.225.48/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic @{Time="11:38 PM"; DayOfMonth="5"; WeekOfMonth="L"; Recurrence="1"}

OUTPUT:
SupportAssistGetAutoCollectSchedule_OUTPUT
    DayOfMonth = *
    DayOfWeek = sun
    Message = Command successful
    MessageID = LC001
    Recurrence = 3
    ReturnValue = 0
    Time = 03:00 AM
    WeekOfMonth = *

30.5 SupportAssistGetAutoCollectSchedule
Once user has registered the server with SupportAssist, this method can be used to get the SupportAssist auto collections schedule. By default, monthly schedule is created for the 1st Sunday (between 12 a.m. – 5 p.m.) for auto collections post successful registration.

winrm i SupportAssistGetAutoCollectSchedule
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+CreationClassName=DCIM_LCService
30.6 SupportAssistClearAutoCollectSchedule

Once user has registered the server with SupportAssist, this method can be used to clear the SupportAssist auto collections schedule. By default, monthly schedule is created for the 1st Sunday (between 12 a.m. – 5 p.m.) for auto collections post successful registration.

Command:
```
winrm i SupportAssistClearAutoCollectSchedule
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://10.94.225.48/wsman
-SkipCNcheck -SkipCACheck
-encoding:utf-8 -a:basic
```

OUTPUT:
```
SupportAssistClearAutoCollectSchedule_OUTPUT
Message = Command successful
MessageID = LC001
ReturnValue = 0
```

30.7 SupportAssistCollection

This method is used to trigger a SupportAssist collection and optionally send the collection to network share or to Dell (on registered servers).

Command:
```
winrm i SupportAssistCollection
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService?
+SystemCreationClassName=DCIM_ComputerSystem
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:root -p:calvin
-r:https://10.94.225.31/wsman
```
Contents of sa_collection.xml is as below:

```xml
<p:SupportAssistCollection_INPUT
xmlns:p="http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2/root/dcim/DCIM_LCService">
<!-- 0 = System Information, 1 = OSApp Data, 2 = Storage Logs, 3 = Debug logs -->
<p:DataSelectorArrayIn>0</p:DataSelectorArrayIn>
<p:DataSelectorArrayIn>1</p:DataSelectorArrayIn>
<p:DataSelectorArrayIn>2</p:DataSelectorArrayIn>
<p:DataSelectorArrayIn>3</p:DataSelectorArrayIn>
<p:IPAddress>Server IP Address</p:IPAddress>
<p:ShareName>Share Name</p:ShareName>
<p:ShareType>2</p:ShareType>
<p:Filter>0</p:Filter>
<p:Username>Share UserName</p:Username>
<p:Password>Share Password</p:Password>
<!--p:Workgroup-->
<p:Transmit>0</p:Transmit>
</p:SupportAssistCollection_INPUT>

OUTPUT:
SupportAssistCollection_OUTPUT
Job
EndpointReference Address =
http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters
ResourceURI =
http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_LifecycleJob
SelectorSet
Selector: InstanceID = JID_921209199264, __cimnamespace = root/dcim
ReturnValue = 4096

30.8 SupportAssistExportLastCollection
This method is used to export an existing SupportAssist collection to the user specified network share or to default network share if set (See Setting Default Network Section).

winrm i SupportAssistExportLastCollection
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:<iDRAC UserName> -p:<iDRAC Password>
-r:<iDRAC IP>/wsman
-SkipCNcheck -skipCACheck -SkipRevocationCheck
-encoding:utf-8 -a:basic
-file:sa_export.xml
Contents of sa_export.xml file is as follows:

```xml
<p:SupportAssistExportLastCollection_INPUT
    <p:ShareType>2</p:ShareType>
    <p:ShareName> Server ShareName </p:ShareName>
    <p:IPAddress> Server IP Address </p:IPAddress>
    <p:Username> CIFS UserName </p:Username>
    <p:Password> CIFS Password </p:Password>
</p:SupportAssistExportLastCollection_INPUT>
```

30.9 SupportAssistUploadLastCollection

```bash
winrm i SupportAssistUploadLastCollection
+CreationClassName=DCIM_LCService
+SystemName=DCIM:ComputerSystem
+Name=DCIM:LCService
-u:<iDRAC UserName> -p:<iDRAC Password>
-r:https://<iDRAC IP>/wsman
-SkipCNcheck -SkipCACheck -SkipRevocationCheck
-encoding:utf-8 -a:basic
```

**OUTPUT:**

SupportAssistUploadLastCollection_OUTPUT

```xml
Job
    EndpointReference Address =
    http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
    ReferenceParameters
    ResourceURI =
    http://schemas.dell.com/wbem/wscim/1/cim-schema/2/DCIM_LifecycleJob
    SelectorSet
        Selector: InstanceID = JID_921219018925, __cimnamespace = root/dcim
    ReturnValue = 4096
```