

Dell Lifecycle Controller
Remote Services
Version 1.4
User's Guide



Notes and Cautions



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



CAUTION: A CAUTION indicates potential damage to hardware or loss of data if instructions are not followed.

Information in this document is subject to change without notice.

© 2010 Dell Inc. All rights reserved.

Reproduction of these materials in any manner whatsoever without the written permission of Dell Inc. is strictly forbidden.

Trademarks used in this text: Dell™, the DELL logo, OpenManage™, PowerEdge™, and PowerVault™ are trademarks of Dell Inc. Intel® is a registered trademarks of Intel Corporation in the U.S. and other countries. Microsoft®, Windows® and Windows Server® are either trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries. Novell® and SUSE® are registered trademarks of Novell, Inc. in the United States and other countries. Red Hat® is a registered trademark of Red Hat, Inc. in the United States and other countries. The term Linux® is a registered trademark of Linus Torvalds, the original author of the Linux kernel. Sun and Java are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other trademarks and trade names may be used in this document to refer to either the entities claiming the marks and names or their products. Dell Inc. disclaims any proprietary interest in trademarks and trade names other than its own.

July 2010

Contents

1	Overview	7
	Why Use Remote Services?	8
	Web Services for Management	8
	What's New in Remote Services 1.4.	13
	Remote Services Features and Support Information	13
	Other Documents You May Need	16
2	Getting Started with Remote Services	17
	Prerequisites for Using Remote Services.	17
	Web Services Setup	17
	WinRM Client.	17
	OpenWSMan Client.	18
3	Remote Services Operations.	19
	Managing Auto-Discovery.	19
	Configuring DHCP/DNS.	19
	Auto-Discovery Configuration	20
	Connecting to Provisioning Server for Initial Credential Deployment.	22

Remotely Reinitiating Auto-Discovery in New Environments	24
Managing Certificates	25
Using Custom Certificates	25
Deploying the Operating System.	27
Operating System Deployment Features	27
Remote Operating System Deployment Interface.	27
Operating System Deployment Typical Use Case Scenario	31
Staging and Booting to Operating System Image on vFlash.	32
Using Remote Update	33
Benefits of Remote Update	33
Supported Devices	34
Scheduling Remote Update.	35
Remote Scheduling Types	36
Managing Part Replacement	37
Using Remote Firmware Inventory	40
Instant Firmware Inventory	40
Supported Devices	40
Firmware Inventory Using WS-Management	41
Retrieving Hardware Inventory	42
Exporting Hardware Inventory	43
Viewing and Exporting Hardware Inventory after Resetting Lifecycle Controller.	43
Lifecycle Log	43
Exporting Lifecycle Log	44
Deleting Configuration and Resetting to Defaults	44

Managing NICs	45
Displaying the NIC Inventory	45
Displaying the NIC Attributes.	45
Setting the NIC Attributes	45
Deleting the Pending Values	47
Managing vFlash SD Card	47
Displaying the Inventory of vFlash SD Card	47
Displaying the Partitions on vFlash SD Card	47
Creating and Modifying a Partitions on vFlash SD Card	48
Managing RAID Configuration.	48
Displaying the RAID Controllers	48
Creating a Virtual Disk	49
Managing BIOS and Boot Configuration	50
Displaying the Inventory of BIOS Attributes	50
Setting the BIOS Attributes.	50
One Time Boot	51
Using Job Control	52
Scheduling Separate Jobs for Multiple Actions	52
Running Multiple Target Jobs	53
Specifying the Start time and Until time	53
4 Remote Services Profiles	55
Operating System Deployment Profile	55
Operating System Deployment Methods	55
Lifecycle Controller Management Profile.	56
Auto-Discovery Methods.	57
Lifecycle Log Methods	57

Hardware Inventory Methods	58
Simple NIC Profile	58
Simple NIC Methods	59
BIOS and Boot Management Profile.	59
BIOS and Boot Management Methods.	60
Persistent Storage Profile	61
RAID Profile	63
RAID Methods	64
Hardware Inventory Profiles.	65
Job Control Profile	67
Job Control Methods	67
A Troubleshooting	69
Error Messages	69
Auto-Discovery LCD Messages	74
B Frequently Asked Questions	77
C Schema	83
Lifecycle Log Schema	83

D Easy-to-use System	
Component Names	85
Index	89

Overview

The Dell Lifecycle Controller provides advanced embedded systems management and is delivered as part of iDRAC Express card and embedded Unified Extensible Firmware Interface (UEFI) applications in the 11th generation Dell servers. It includes a 1GB managed and persistent storage that embeds systems management features in addition to the iDRAC features. You can further upgrade to iDRAC Enterprise and the vFlash SD card reader. A vFlash SD card enables hosting of customized and bootable service images.

The Dell Lifecycle Controller Remote Services further enable remote systems management in a one-to-many method. Remote Services is available using Web Service for Management (WS-Management) protocol based web services interface for remote server provisioning and management through the iDRAC. The interface is aimed at simplifying many tasks, some of which include remote operating system (OS) deployment, remote update and inventory, and automating the setup and configuration of new Dell systems remotely.

Remote services are accessible over the network using a secured web services interface and can be programmatically utilized by applications and scripts. Remote services enable management consoles to perform one-to-many bare metal server provisioning. The combination of the Auto-discovery feature to identify and authenticate the attached Dell system to the network and integration with one-to-many management consoles reduces the manual steps required for server provisioning.

Remote services enables the Dell Management Console, the Dell Modular Chassis Management Controller, partner consoles, customer home grown consoles and scripts to **remotely** perform systems management tasks such as:

- Install operating systems and drivers
- Perform BIOS firmware updates
- Perform component firmware updates
- Get hardware inventory information
- Get and set NIC and RAID configuration

- Get and set BIOS configuration
- Export lifecycle log and add user comments
- Export hardware inventory log
- Manage, attach, and boot to vFlash SD card partitions
- Schedule and track the status of the update and configuration jobs

Why Use Remote Services?

Remote services offer the following benefits and features:

- Leverages your existing console for one-to-many server provisioning.
- Does not utilize operating system resources on the managed system.
- Provides a secure communication path for management.
- Reduces manual intervention and improves efficiency while provisioning servers.
- Allows scheduling configuration changes and updates, thereby reducing maintenance shutdown time.
- Enables **PowerShell** and scripting for command line interface (CLI) access.
- Enables integration to consoles through WS-Management interfaces.
- OS-agnostic software update.

Web Services for Management

WS-Management is a Simple Object Access Protocol (SOAP)-based protocol designed for systems management. It is published by the Distributed Management Task Force (DMTF) and provides an interoperable protocol for devices to share and exchange data across networks. The WS-Management implementation complies with the DMTF WS-Management specification version 1.1.0.

Dell Lifecycle Controller - Remote Services uses WS-Management to convey DMTF Common Information Model (CIM)-based management information; the CIM information defines the semantics and information types that can be manipulated in a managed system. Dell utilizes the WS-Management interface to allow remote access to the hardware lifecycle

operations. The Dell-embedded server platform management interfaces are organized into profiles, where each profile defines the specific interfaces for a particular management domain or area of functionality. Additionally, Dell has defined a number of model and profile extensions that provide interfaces for additional capabilities. The data and methods available through WS-Management are provided by the Lifecycle Controller - Remote Services' instrumentation interface mapped to the following DMTF profiles and Dell extension profiles:

Standard DMTF

- **Base Server** — Defines CIM classes for representing the host server.
- **Base Metrics** — Defines CIM classes for providing the ability to model and control metrics captured for managed elements.
- **Host LAN Network Port** — Defines CIM classes for representing a network port that provides a LAN interface to a host system, its associated controller, and network interfaces.
- **Service Processor** — Defines CIM classes for modeling service processors.
- **USB Redirection** — Defines CIM classes for describing information about USB redirections. For keyboard, video, and mouse devices, this profile should be used if the devices are to be managed as USB devices.
- **Physical Asset** — Defines CIM classes for representing the physical aspect of the managed elements.
- **SM CLP Admin Domain** — Defines CIM classes for representing CLP's configuration.
- **Power State Management** — Defines CIM classes for power control operations.
- **Command Line Protocol Service** — Defines CIM classes for representing CLP's configuration.
- **IP Interface** — Defines CIM classes for representing an IP interface of a managed system.
- **DHCP Client** — Defines CIM classes for representing a DHCP client and its associated capabilities and configuration.
- **DNS Client** — Defines CIM classes for representing a DNS client in a managed system.

- **Record Log** — Defines CIM classes for representing different type of logs.
- **Role Based Authorization** — Defines CIM classes for representing roles.
- **SMASH Collections** — Defines CIM classes for representing CLP's configuration.
- **Profile Registration** — Defines CIM classes for advertising the profile implementations.
- **Simple Identity Management** — Defines CIM classes for representing identities.

Dell Extensions

- **Dell Active Directory Client Version 2.0.0** — Defines CIM and Dell extension classes for configuring the Active Directory client and the local privileges for Active Directory groups.
- **Dell Virtual Media** — Defines CIM and Dell extension classes for configuring Virtual Media. Extends the USB Redirection Profile.
- **Dell Ethernet Port** — Defines CIM and Dell extension classes for configuring NIC Side-Band interface for the NIC. Extends the Ethernet Port Profile.
- **Dell Power Utilization Management** — Defines CIM and Dell extension classes for representing the host server's power budget and for configuring/monitoring the host server's power budget.
- **Dell OS Deployment** — Defines CIM and Dell extension classes for representing the configuration of operating system deployment features.
- **Dell Software Update Profile** — Defines CIM and Dell extensions for representing the service class and methods for updating BIOS, component firmware, Lifecycle Controller firmware, Diagnostics, and Driver Pack.
- **Dell Software Inventory Profile** — Defines CIM and Dell Extensions for representing currently installed BIOS, component firmware, Diagnostics, Unified Server Configurator, and Driver Pack versions. Also provides representation of versions of BIOS and firmware update images available in Lifecycle Controller for rollback and re-installation.
- **Dell Job Control Profile** — Defines CIM and Dell extensions for managing jobs generated by update requests. Jobs can be created, deleted, modified and aggregated into job queues to sequence and perform multiple updates in a single reboot.

- **Dell Lifecycle Controller Management Profile** — Defines CIM and Dell extensions for getting and setting attributes for managing Auto-Discovery, Part Replacement, managing Lifecycle Log, and hardware inventory export.
- **Active Directory Client Profile** — Defines the configuration of the Active Directory client service and the groups managed by this service.
- **Power Supply Profile** — Defines the power supplies for manageability and describes the power supplies in a redundant configuration.
- **Power Topology Profile** — Defines a hierarchy of power sources; power supplies and external power domains, and their redundancies.
- **SMASH Collections Profile** — Defines the collections that support Systems Management - Command Line Protocol (SM-CLP) target addressing.
- **Virtual Media Profile** — Provides the capability to manage virtual media sessions and devices that utilize the USB redirection services provided by the iDRAC service processor.
- **Dell RAID Profile** — Describes the classes, properties and methods for the representation and configuration of RAID storage.
- **Dell Simple NIC Profile** — Describes the classes, properties and methods for the representation and configuration of the NIC network controllers.
- **Dell Persistent Storage Profile** — Describes the classes, properties and methods to represent and manage the partitions on the vFlash SD card on Dell platforms.
- **Dell BIOS and Boot Management Profile** — Describes the classes, properties and methods to represent the configuration of the system BIOS setup and to manage the boot order of the system.
- **Dell CPU Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of processors in a managed system.
- **Dell Fan Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of fans in a managed system.
- **Dell iDRAC Card Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of basic properties of iDRAC card.

- **Dell Memory Info Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of memories (DIMMs) in a system.
- **Dell PCI Device Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of PCI devices in a system.
- **Dell Power Supply Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of power supplies in a system.
- **Dell System Info Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of the host system.
- **Dell Video Profile** — Describes the properties and interfaces for executing systems management tasks related to the management of video controllers in a system.

The Lifecycle Controller - Remote Services WS-Management implementation uses SSL on port 443 for transport security, and supports basic authentication. Web services interfaces can be utilized by leveraging client infrastructure such as Windows WinRM and Powershell CLI, open source utilities like WS-MANCLI, and application programming environments like Microsoft .NET.

What's New in Remote Services 1.4

- Auto-Discovery with LCD status
- Certificate management
- Hardware inventory
- NIC configuration
- RAID configuration
- BIOS configuration
- Enhanced job/task control features
- Part replacement with firmware update and configuration recovery
- vFlash SD card management
- Lifecycle log management

Remote Services Features and Support Information

The Remote Services features supported by a Dell server depend on the system configuration. Table 1-1 shows the product classifications for Remote Services. For example, for a Dell system y71x series, y denotes letters such as M, R, or T; and x denotes numbers.

Table 1-1. Product Classification for Lifecycle Controller Remote Services

Dell System Series	Options	Available Systems Management Device	Available Remote Services Features
y11x	No Options	Embedded BMC	NA

Table 1-1. Product Classification for Lifecycle Controller Remote Services

Dell System Series	Options	Available Systems Management Device	Available Remote Services Features
y21x to y51x	Standard	Embedded BMC	NA
	Optional	Embedded BMC + iDRAC6 Express Card	Platform Update, Hardware Configuration, Driver Repository, Remote OS Deployment, Remote Update, Remote Configuration, View and Export Hardware Inventory, Auto-Discovery, View and Export Lifecycle Log, and Add a Comment to Lifecycle Log.
		Embedded BMC + iDRAC6 Express card + iDRAC6 Enterprise card	<p>iDRAC6 Express - adds Platform Update, Hardware Configuration, Driver Repository, Remote OS Deployment, Remote Update, Remote Configuration, View and Export Hardware Inventory, Auto-Discovery, View and Export Lifecycle log, and Add a Comment to Lifecycle Log.</p> <p>iDRAC6 Enterprise - adds Full Remote Management, Dedicated NIC port, Virtual KVM, Part Replacement, and vFlash SD Card Management.</p>

Table 1-1. Product Classification for Lifecycle Controller Remote Services

Dell System Series	Options	Available Systems Management Device	Available Remote Services Features
	Standard	Embedded BMC with iDRAC6 Express card	Hardware Diagnostics, Platform Update, Hardware Configuration, Driver Repository, Remote OS Deployment, Remote Update, Remote Configuration, View and Export Hardware Inventory, Auto-Discovery, View and Export Lifecycle Log, and Add a Comment to Lifecycle Log.
y6lx to y9lx	Optional ¹	Embedded BMC with iDRAC6 Express card + iDRAC6 Enterprise card	<p>iDRAC6 Express - Hardware Diagnostics, Platform Update, Hardware Configuration, Driver Repository, Remote OS Deployment, Remote Update, Remote Configuration, View and Export Hardware Inventory, Auto-Discovery, View and Export Lifecycle Log, and Add a Comment to Lifecycle Log.</p> <p>iDRAC6 Enterprise - adds Full Remote Management, Dedicated NIC port, Virtual KVM, Part Replacement, and vFlash SD Card Management.</p>

1. For Dell modular systems — BMC, iDRAC6 Express card, and iDRAC6 Enterprise card are included as standard configurations.

For information on the supported systems and operating systems, see the *Dell Systems Software Support Matrix* at support.dell.com/manuals.

See the *Glossary* at support.dell.com/manuals for terms used in this document.

Other Documents You May Need

In addition to this guide, you can access the following guides available at support.dell.com/manuals. On the **Manuals** page, click **Software** → **Systems Management**. Click on the appropriate product link on the right-side to access the documents.

- The *Integrated Dell Remote Access Controller 6 (iDRAC6) Enterprise for Blade Servers User Guide* provides information about configuring and using an iDRAC6 for blade servers to remotely manage and monitor your system and its shared resources through a network.
- The *Integrated Dell Remote Access Controller 6 (iDRAC6) User Guide* provides complete information about configuring and using an iDRAC6 for rack and tower servers to remotely manage and monitor your system and its shared resources through a network.
- The *Glossary* provides information about the terms used in this document.

There are additional implementation guides, white papers, profile specifications, class definition (.mof) files, and code samples you may require apart from this user's guide. See:

- Lifecycle Controller page on Dell TechCenter — delltechcenter.com/page/Lifecycle+Controller
- Lifecycle Controller WS-Management Script Center — delltechcenter.com/page/Scripting+the+Dell+Lifecycle+Controller
- MOFs and Profiles — delltechcenter.com/page/DCIM.Library
- DTMF Web site — dmtf.org/standards/profiles/
- *Dell Lifecycle Controller Remote Services WS-Management Release Notes*

Getting Started with Remote Services

This section describes some of the prerequisites that will help you get started with the Remote Services functionality and use the new features effectively, for better results.

Prerequisites for Using Remote Services

Web Services Setup

Ensure that the following conditions are met while setting the system:

- Use the following tools to access Remote Services:
 - Windows-based client WinRM that is already installed in the operating system, else you can download it from support.microsoft.com/kb/968930.
 - Linux-based clients like the open-source OpenWSMan based CLI. For more information, see openwsman.org.
 - Java-based client such as open-source project **Wiseman**. For more information, see wiseman.dev.java.net.
- Ensure that you know the IP address of the systems on your network. You will also need to be able to connect to iDRAC. See the iDRAC documentation at support.dell.com/manuals for more information.
- Ensure the proper network configuration for client and managed server. Verify the connectivity with the ping utility. Then ensure that the client and network allows HTTP and SSL protocols.

WinRM Client

You will need to install the WinRM Client on your console to be able to use the Remote Services functionality. Microsoft Windows 7, Microsoft Windows Vista, and Microsoft Windows Server 2008 contain a standard component called WS-Management. This component contains the WinRM client. For Microsoft Windows XP and Microsoft Server 2003, you can download and install this component from support.microsoft.com/kb/968929. You need local administrator privileges for installation.

You must configure the client for the connection. For more information, see the *Lifecycle Controller 1.4 Web Services Interface Guide*.

OpenWSMan Client

The OpenWSMan client is the WS-Management CLI that is part of the open-source project Openwsman. To download, build, install, and use the WS-Management CLI and OpenWSMan packages from sourceforge.net, see openwsman.org for download links.



NOTE: You must configure the client for the connection. For configuration details, see the *Lifecycle Controller 1.4 Web Services Interface Guide*.

Remote Services Operations

This section describes the Remote Services features with high-level descriptions and sample tasks. For more information on the tasks, see the Use Cases section in the individual profile documents at delltechcenter.com/page/DCIM.Library.

Managing Auto-Discovery

The Auto-Discovery feature allows newly installed servers to automatically discover the remote management console that hosts the Provisioning Server. The Provisioning Server provides custom administrative user credentials to the iDRAC so that the unprovisioned server can be discovered and managed by the management console.

When Auto-Discovery is enabled, the iDRAC6 requests an IP address from DHCP and either acquires the name of the Provisioning Server host and/or subsequently resolves the address through DNS. After acquiring the Provisioning Server host address, the iDRAC6 securely handshakes with the Provisioning Server before acquiring custom administrative account credentials. The iDRAC can now be managed through its newly acquired credentials to perform operations, such as remote operating system deployment.

If you ordered a Dell system with the Auto-Discovery feature **Enabled** (factory default setting is **Disabled**), then the iDRAC will be delivered with DHCP-enabled and no enabled user accounts. If the auto-discovery feature is set to **Disabled**, you can manually enable this feature and disable the default administrative account from the **iDRAC6 Configuration Utility** when booting your system.

For more information on auto-discovery, see the "Lifecycle Controller Management Profile" on page 56.

Configuring DHCP/DNS

Before adding your Dell system to the network and utilizing the Auto-Discovery feature, ensure that Dynamic Host Configuration Protocol (DHCP) server/Domain Name System (DNS) are configured with added

support for Auto-Discovery. There are several options for enabling the network environment to support discovery of the Provisioning Server host by unprovisioned servers.

One of the following prerequisites must be met for the Auto-Discovery feature to work properly:

- The DHCP server provides a comma separated list of Provisioning Server locations using a vendor scope option of class LifecycleController option 1. These locations can be a hostname or IP address and optionally include a port. The iDRAC will resolve the hostname of the management console to an IP address with a DNS lookup.
- The DNS server specifies a service option `_dcimprovsrv._tcp` that will resolve to an IP address.
- The DNS server specifies an IP address for a server with the known name `DCIMCredentialServer`.

For more information on configuring DHCP and DNS, see *Lifecycle Controller Auto Discovery Network Setup Specification* on the Dell Enterprise Technology Center at delltechcenter.com/page/Lifecycle+Controller.

Auto-Discovery Configuration

To manually enable the Auto-Discovery feature:

- 1** Press `<Ctrl><e>` when prompted within 5 seconds during system start-up.

The **iDRAC6 Configuration Utility** page appears.

- 2** Enable **NIC** (for modular system only.)
- 3** Enable **DHCP**.
- 4** Navigate to **LAN Parameters**.
- 5** Select **Domain Name** from DHCP and select **On**.
- 6** Select **DNS Server** from DHCP select **On**.
- 7** Navigate to **LAN user configuration**.
 - a** Select **Account Access** and select **Disabled**.
This disables the default administrative account.
 - b** Select **Auto-Discovery**.

- c Select **Enable** to enable the Auto-Discovery feature.



NOTE: Auto-Discovery feature will not run if the administrator accounts are enabled.

- 8 Save and exit iDRAC6 Configuration Utility.
- 9 Restart your system.

Auto-Discovery Workflow

This is the Auto-Discovery workflow once it is configured and enabled:

- 1 Plug in your new Dell system to your network.
- 2 Plug-in the power cables to turn on the system.
- 3 iDRAC starts, acquires the Provisioning Server IP addresses/hostnames from DHCP/DNS and announces itself to the Provisioning Server.
- 4 The Provisioning Server validates and accepts the secure handshake session from the iDRAC.
- 5 The Provisioning Server provides custom user credentials with administrator privileges to iDRAC.
- 6 iDRAC receives and completes the secure handshake.

With enhancements to the Auto-Discovery process you can:

- Configure the provisioning server host address through the iDRAC Configuration utility, Unified Server Configurator (USC), or using WinRM commands instead of using DHCP or DNS.
- Remotely reinitiate Auto-Discovery in new environments.
- Upload custom client and server certificates using WS-Management.

Viewing the Discovery Status on the System

You can view the status of the Discovery and Handshake on the LCD (running, stopped, suspended, or complete.)

After the system is connected to the network:

Use the Auto-Discovery setup on iDRAC Option ROM (CTRL+E) to set the Auto-Discovery status, save and exit. The LCD displays the status as running.

If the discovery process is running, you can view its progress code that corresponds to how far the last attempt reached (i.e. whether Discovery and Handshake is blocked because the NIC is disabled, or an administrator account is enabled, and so on). You can also view the time left before timeout. For example, a menu item could be added for Auto-Discovery at the same level as iDRAC network setting.

Connecting to Provisioning Server for Initial Credential Deployment

This feature allows you to directly connect to a specified Provisioning Server host for handshake and registration of the new server on the network. You can manually configure the provisioning server IP address or host name through the USC console, or through a web services request using WS-Management, or iDRAC6 configuration utility, or preset at the factory.

Set Provisioning Server Using a WS-Management Request

The Provisioning Server IP address property is set by invoking the `SetAttribute()` method on the `DCIM_LCService` class through WS-Management. See the profile specific chapters in this user guide for command line examples of Microsoft WinRM `SetAttribute()` invocations or in the *Lifecycle Controller 1.4 Interface Guide* on the Dell TechCenter wiki at delltechcenter.com/page/Lifecycle+Controller.

The following conditions apply to using a command to set the provisioning server IP address/hostname:

- When issuing the `racadm racresetcf` or updating iDRAC6, ensure to enable the Preserve Configuration option while resetting the iDRAC6 to defaults. If this option is disabled, the provisioning server IP/hostname is erased.

- Auto-Discovery feature does not use the newly set provisioning server IP address/hostname for any handshakes in progress, but is used only during the next handshake process.
- Auto-Discovery feature supports setting multiple IP addresses and/or hostnames using the following format:
 - The string is a list of IP addresses and/or hostnames and ports separated by comma.
 - Hostname can be fully qualified.
 - IPv4 address – starts with ‘(‘ and ends with ‘)’ when specified at the same time with a hostname.
 - Each IP address or hostname can be optionally followed by a ‘:’ and a port number.
 - Examples of valid strings are - hostname, hostname.domain.com.

Setting Provisioning Server using the USC Console

- 1** Press <F10> System Services when prompted within 5 seconds during system startup.
The **Unified Server Configurator Lifecycle Controller Enabled** screen appears.
- 2** Navigate to **Hardware Configuration -> Configuration Wizard -> iDRAC6 Configuration**.
- 3** Use the **Next** button to navigate to the **LAN User Configuration** screen.
- 4** Navigate to the **Provisioning Server Addresses** screen.
- 5** Enter the IP/hostname string of the Provisioning Server host.
- 6** Click **Next** and then click **Apply**.
- 7** Click **Finish**.
- 8** Click **Exit and Reboot**. Confirm exit.

Set Provisioning Server using iDRAC6 Configuration Utility

- 1** Press <Ctrl+e> when prompted within 5 seconds during system start-up.
The **iDRAC6 Configuration Utility** screen appears.

- 2 Navigate to the **LAN User Configuration** screen and select the **Provisioning Server**.
- 3 Type the IP/hostname string of the Provisioning Server host and click **Enter**.
- 4 Save and Exit the iDRAC6 Configuration Utility.

Remotely Reinitiating Auto-Discovery in New Environments

This feature allows you to reinitiate Auto-Discovery through WS-Management, even though Auto-Discovery may have taken place earlier. Use this feature to move a server from one data center to another. The Auto-Discovery settings are persisted along with the credentials used for discovery. When the server is powered on in the new data center, Auto-Discovery will run according to the settings, and will download the new user credentials for the new data center.



NOTE: The Auto-Discovery uses WS-Management, so the iDRAC administrator or iDRAC user with Execute Server Command privilege is required.

The supported WS-Management interface to reinitiate Auto-Discovery includes these options:

- Whether Auto-Discovery will run immediately or at the next AC power cycle. This is a required input.
- Provisioning Server IP address/hostname. This is optional.

Regardless of the options you specify, the following operations are performed as part of the Auto-Discovery initiation:

- Enable NIC (modular servers)
- Enable IPv4
- DHCP enable
- Disable all administrator accounts
- Disable Active Directory
- Get DNS server address from DHCP
- Get DNS domain name from DHCP

The described interfaces are specified in the Dell Lifecycle Controller Management Profile available at delltechcenter.com/page/DCIM+Extensions+Library. Managed Object Format (MOF) files for related class and method definitions are also available in the Dell TechCenter DCIM Extensions Library area. The interfaces are:

ReinitiateDHS(ProvisioningServer, ResetToFactoryDefaults, PerformAutoDiscovery)

- **ProvisioningServer**: optional parameter to indicate the Provisioning Server information. This could be an IP address or a hostname.
- **ResetToFactoryDefaults**: required parameter (**TRUE** or **FALSE**) to indicate whether the current configuration data needs to be deleted prior to the next cycle of Auto-Discovery. Only **TRUE** will be accepted; specifying **FALSE** will cause an error message indicating the parameter value is not supported. **TRUE** will reset iDRAC to the default values and then set iDRAC for Auto-Discovery. iDRAC will not be available until the Auto-Discovery provisioning process is complete and the iDRAC receives the new credentials.
- **PerformAutoDiscovery**: required parameter to indicate when the next Auto-Discovery cycle should be performed: immediately or at the next boot. Select **Now** to run the Auto-Discovery cycle immediately; select **Next** to run it the next time you boot your system.

SetAttribute(ProvisioningServer)

- **ProvisioningServer**: parameter to indicate the Provisioning Server IP address/host name.
- **ClearProvisioningServer()**: Method to clear the Provisioning Server property. No input parameters are required.

Managing Certificates

Using Custom Certificates

You can now transfer custom-defined certificates to the iDRAC6, and create a unique certificate based on the service tag of your system to ensure enhanced security. You can also have the factory preset the system with the certificate of your choice using the Custom Factory Install (CFI) process available from Dell.

Creating Custom Trusted Root Client Certificates for the Provisioning Server

The `DownloadClientCerts()` method on the `DCIM_LCService` class can be called to generate a custom signed Auto-Discovery client certificate. The method takes as input a Certificate Authority generated key certificate and related hash and password parameters. The key certificate provided is used to sign a certificate containing the system service tag as the Common Name(CN). The method returns a job ID that can be used to check the success of the download, generation, and installation of the Auto-Discovery client certificate. For examples of command line invocations using WinRM and WSMANCLI see the *Lifecycle Controller 1.4 Web Services Interface Guide*.

Providing Custom Server Certificates using WS-Management

The `DownloadServerPublicKey()` method on the `DCIM_LCService` class can be called to transfer a Provisioning Server public key certificate. The Provisioning Server public key can be used as part of mutual authentication between the Auto-Discovery client and the provisioning server. The method takes as input a Provisioning Server public key certificate and related hash and hash type parameters. The method returns a job ID that can be used to check the success of the processing and installation of the Provisioning Server public key. For examples of command line invocations using see the *Lifecycle Controller 1.4 Web Services Interface Guide*. DCIM Profile specification and related MOF files are available at Dell TechCenter wiki in the DCIM Extension Library area (delltechcenter.com/page/DCIM.Library.)

Deleting the Custom Certificates Using WS-Management

You can delete the custom certificate that is part of the managed server supplied from the factory. Using this feature, you can wipe all the custom signed certificates from the server, whenever required.



NOTE: This feature does not delete the factory certificates.

Custom Server Public Key Deletion using WS-Management

Use the `DeleteAutoDiscoveryServerPublicKey()` method on the `DCIM_LCService` class to delete the CA certificate that is used to validate or authenticate server certificates.

Custom Client Certificate Deletion using WS-Management

Use the `DeleteAutoDiscoveryClientCerts()` method on the `DCIM_LCService` class to delete a client certificate and private key.

Changing the Web Server/WS-Management Encryption Certificate and Private Key from PKCS #12

- 1 Generate a CSR and private key. The CSR needs to be signed by a CA.
- 2 Combine the certificate with the private key then encrypt it into a PKCS#12 file.
- 3 BASE64 encode the PKCS#12 file to convert it from binary to text so you can pass it as a WS-Management parameter.
- 4 Copy the contents of the active certificate to a XML file.

Deploying the Operating System

The operating system deployment capabilities enable deployment of an operating system remotely using WS-Management web services protocols and CIFS and NFS network file sharing protocols.

Operating System Deployment Features

These are the capabilities of remote operating system deployment:

- Remote activation of local exposure of embedded drivers as a USB device.
- Remote acquisition of embedded drivers per selected operating system.
- Boot to an ISO image located on a network share.
- Download ISO to vFlash SD card and boot from the card.

For more information on operating system deployment profile, see the "Operating System Deployment Profile" on page 55.

Remote Operating System Deployment Interface

Dell Operating System Deployment web services interface provides the capability to support operating system deployment using the features provided by the iDRAC service processor. Detailed interface specifications and class definition (.mof) files can be found at the Lifecycle Controller area on the Dell Enterprise Technology Center at delltechcenter.com. Using CIM and Dell extension classes using the web services protocols WS-Management, Dell Operating System Deployment feature provides the following capabilities:

- Get the driver pack (a package of all supported operating system drivers for all supported operating systems for the platform) version:

Remote management consoles, applications, and scripts request driver pack version and list of supported operating systems from iDRAC through WS-Management.

The `GetDriverPackInfo()` method on the `DCIM_OSDeploymentService` class returns the driver pack version and the list of operating systems supported by the driver pack.

- After determining which operating system the drivers support, one of the following methods can be invoked through WS-Management to unpack the appropriate drivers and expose them locally or acquire them remotely.
 - a** The `UnpackAndAttach()` method on the `DCIM_OSDeploymentService` class extracts the drivers for the requested operating system and places them on an internal USB device labeled `OEMDRV`. The `OEMDRV` appears as a locally attached USB device to the system. The method takes the operating system name and an expose duration time as input parameters and returns a job identification that can be subsequently checked for the status of the unpack and attach activity.
 - b** The `UnpackAndShare()` method on the `DCIM_OSDeploymentService` class extracts the drivers for the requested operating system and copies them to a network share. The method takes the operating system name and network share information as input parameters and returns a job identification that can be subsequently checked for the status of the unpack and share activity. Network share information includes the IP address of the share, the share name, share type, and username, password and workgroup data for secure shares.

Important

- The drivers unpacked and attached are removed after the time specified in `ExposeDuration` parameter or if no time is specified in the method invocation then by default the `OEMDRV` USB device will be removed after 18 hours.
- Ensure that network based ISO images attached during the process are detached before you use Unified Extensible Firmware Interface (UEFI) System Service.

- When installing Red Hat Linux 5.3 using remote enablement commands, the installation will fail whenever there is an OEM drive (for driver source) attached. To avoid failure, do not attach the OEM drive when using remote enablement commands to install Red Hat Enterprise Linux 5.3.
- The following methods can be used to boot the system from an ISO image on a network share or to initiate PXE boot mechanisms:
 - a The **BootToNetworkISO()** method on the **DCIM_OSDeploymentService** class will boot the system using an ISO image that has been made available on a CIFS or NFS network share. The method takes the ISO image name, network share information, and exposure duration as input parameters and returns a job identification that can be subsequently checked for the status of the unpack and share activity. Network share information includes the IP address of the share, the share name, share type, and username, password and workgroup data for secure shares. For additional security a hash value can be calculated using well known hash algorithms and this value along with the type of the hash used can be provided as input parameters.
 - b The **BootToPXE()** method on the **DCIM_OSDeploymentService** class initiates a Pre-Boot Execution Environment (PXE) boot of the system. The method requires no input parameters.

Important

- The drivers unpacked and attached are removed after the time specified in **ExposeDuration** parameter. If no time is specified in the method invocation, then by default the OEMDRV USB device will be removed after 18 hours.
- Ensure that network based ISO images attached during the process are detached before you use UEFI System Service.
- The following methods are used to directly detach the local OEMDRV device or the network ISO image. These can be used before the previously set exposure durations time out:
 - a The **DetachDrivers()** method on the **DCIM_OSDeploymentService** class detaches and removes the **OEMDRV** device that had been previously attached by an invocation of the **UnpackAndAttach()** method.

- b** The `DetachISOImage()` method on the `DCIM_OSDeploymentService` class detaches and removes the network share based ISO image that had been previously attached by an invocation of the `BootToNetworkISO()` method.
- Several methods described in this document return job identifiers as output parameters. The jobs provide a means of keeping track of a requested action that cannot be performed immediately and, because of underlying technology constraints, will take longer than standard web service request response timeouts. The returned job identifier can subsequently be used in WS-MAN Enumerate or Get requests to retrieve job object instances. Job object instances contain a job status property that can be checked to see what state the job is in and whether it completed successfully or encountered a problem and failed. If a job failure occurs, the job instance also contains an error message property that provides detailed information on the nature of the failure. Other properties contain other error identification information that can be used to localize the error message to the supported languages and get more detailed error descriptions and recommended response action descriptions.
- The `GetHostMACInfo()` method on the `DCIM_OSDeploymentService` class returns an array of physical network port MAC addresses representing all the LAN on Motherboard (LOM) ports in the system. The method requires no input parameters.
- All the `DCIM_OSDeploymentService` methods described in this document return error codes indicating whether the method successfully executed, an error occurred, or a job was created. Job creation occurs if the action being performed in the method cannot be completed immediately. Additionally, if an error occurs, the methods will also return output parameters that include an error message (in English) and other error identifiers that can be used to localize the error to the supported languages. The error identifiers can be used to index into and process Dell Message Registry XML files. The Dell Message Registry files are available in the six supported languages, one file per language. In addition to translated error messages, the Message Registry files contain additional detailed error descriptions and recommended response actions for each error returned by the Lifecycle Controller Remote Services web service interface. To download the Dell Message Registry XML files, see delltechcenter.com/page/Lifecycle+Controller.

Operating System Deployment Typical Use Case Scenario

This section contains a typical scenario for deploying an operating system remotely.

Prerequisites and Dependencies

The following are the prerequisites and dependencies for remotely deploying the operating system:

- Boot disk is available to install operating system, or the operating system ISO image on the network share.
- It is recommended that the latest driver pack is installed so that they are available for newer operating systems.
- Provisioning console, application or appropriate scripts that are capable of sending WS-Management Web services requests and method invocations.

Workflow

The following is a typical workflow for remote operating system deployment:

- Create the custom pre-operating system/operating system image and share it on the network, or create the required operating system media ISO image.
- Get the list of supported operating system and driver pack version information.
- Stage the operating system drivers by unpacking and attaching drivers for operating system deployment. These drivers will be installed during the operating system deployment process.
- Remotely boot to the custom pre-operating system/operating system image to initiate the operating system deployment process.
- Run detach commands to detach the ISO media and driver device.

For more information on the Lifecycle Controller Remote Operating Systems Deployment feature including the Lifecycle Controller 1.4 Web Services Interface Guideline, white papers, the Dell OS Deployment Profile data model specification, class definition (.mof) files, sample code and scripts, see the Lifecycle Controller area on the Dell Enterprise Technology Center at delltechcenter.com.

Staging and Booting to Operating System Image on vFlash

This feature allows you to download an ISO image to the vFlash SD card on the target system and booting the system to this ISO image.

Prerequisite

This feature is available only if you have Dell-licensed vFlash present on your system.

WS-Management Methods

Important

If the supported SD card is installed and not formatted, executing the download ISO command will first format the SD card and then download to ISO image.

The WS-Management methods under the operating system deployment profile for vFlash are:

- **DownloadISOTOVFlash** - Downloads the image to the vFlash. Support is available for CIFS, TFTP and NFS.
- **BootToISOFromVFlash** - Boots to the ISO image that has been staged on the vFlash. You cannot perform this action if you are using the iDRAC GUI or RACADM commands to communicate with the vFlash. This command will also reboot or power on your system if it is in an **Off** state once executed.
- **DetachISOFromVFlash** - Detaches the partition so that the console cannot access it anymore.
- **DeleteISOFromVFlash** - Deletes the ISO image from the vFlash partition. This command will execute only if the ISO is detached.

You will need to perform the following steps to complete the process:

- 1 Download the ISO image to the vFlash.
- 2 Get the concrete job ID and poll for the completion of this job.
- 3 Run the `BootToISOFromVFlash` command. This will attach the image as a CD ROM, boot to the attached image and then continue with the operating system installation.
- 4 Get the concrete job ID and poll for the completion of this job.

- 5 Detach the partition on the vFlash SD card.
- 6 Delete the ISO image from the partition.

Using Remote Update

Remote update, also known as out-of-band update or operating system-independent platform update, allows you to update the system independent of the state of the operating system. You can initiate the firmware update regardless of the system power on or off state.

Benefits of Remote Update

With Operating System independent platform update, an operating system need not be running on the system. Multiple updates can be scheduled together along with a graceful or power-cycle reboot into UEFI system services to perform the updates. Although the updates may involve intermediate BIOS restarts, Lifecycle Controller will automatically handle them until the updates are complete.

This feature supports two methods to perform updates:

- **Install from Uniform Resource Identifier (URI)** — This method allows a WS-Management request to install or update software on a host platform using a URI. The URI consists of a string of characters used to identify or name a resource on the network. The URI is used to specify the location of the Dell Update Package image on the network that can be downloaded to the Lifecycle Controller and then installed.
- **Install from Software Identity** — This method allows update or rollback to a version that is already available on the Lifecycle Controller.

You can use a WS-Management capable application, script or command line utility to perform a remote update. The application or script performs WS-Management invoke method request using one of the remote update interface methods. The iDRAC then downloads the firmware from the network share (local network share, CIFS, NFS, FTP, TFTP, http) URI and stages the updates to be performed at the specified time and utilizing the specified graceful, power cycle or none system reboot types.

Important

When you perform a remote update on the Driver Pack for the system it will replace the current driver pack. The replaced driver pack will no longer be available.

Supported Devices

Remote Update is supported for the following devices and components:

- iDRAC6
- RAID Series 6 and 7
- NICs and LOMs (Broadcom)
- Power supplies
- BIOS
- OS Driver Pack
- USC
- Diagnostics

Workflow for Remote Update from URI

- 1** Use the appropriate WS-Management client to send a method invocation request to the iDRAC IP address. The WS-Management command includes the **InstallFromURI()** method on the `DCIM_SoftwareInstallationService`, and the location from where iDRAC should download the Dell Update Package (DUP). The download protocols that are supported are FTP, HTTP, CIFS, NFS, and TFTP.
- 2** When the WS-Management command is invoked successfully, a Job ID will be returned back.
- 3** Additional **InstallFromURI()** method invocation requests can be sent using WS-Management to create other update jobs.
- 4** A reboot job can be created by invoking the **CreateRebootJob()** method on the `DCIM_SoftwareInstallationService` and specifying the desired reboot type. The reboot type can be graceful, power cycle or graceful with power cycle after 10 minutes.

- 5 Using the update and reboot Job IDs, you can use the Dell Job Control Profile profile to schedule these jobs to run immediately or at future date and time. You can also use the Job ID to query the status of a job or to cancel a job.
- 6 All jobs will be marked successful or, if an error occurred during downloading or updating, failed. For failed jobs, the error message and error message ID for the failure are available in the job information.

Important

- After successfully downloading the DUP and extracting it, the downloader will update the status of the job as "Downloaded" and the job can then be scheduled. If the signature is invalid or if download/extraction fails then the Job status is set to "Failed" with an appropriate error code.
- Updated firmware can be viewed by requesting firmware inventory after firmware update jobs have completed.

Scheduling Remote Update

The remote update scheduling capability provides the ability to schedule or stage firmware updates now or in the future. Updates for Diagnostics and USC can be performed directly and do not require any staging. These updates will be applied as soon as they are downloaded and do not need the Job Scheduler. All other remote updates are staged updates, and require scheduling, using different scheduling options. The DUPs are downloaded to the Lifecycle Controller and staged, and the actual update is performed by rebooting the system into UEFI System Services.

There are multiple options for scheduling updates:

- Run updates on the desired components at a desired time.
- Run the reboot command to get a reboot job ID.
- Check on the status of any of the jobs by enumerating `DCIM_SoftUpdateConcreteJob` instances and checking the `JobStatus` property value.
- Schedule the job using the `SetupJobQueue()` method on the `DCIM_JobService`.



NOTE: For Remote Services version 1.3 remote updates, you can only use the `SetupJobQueue()` method.

- Delete existing jobs using the `DeleteJobQueue()` method on the `DCIM_JobService`.

Important

USC, Diagnostics and Driver Pack updates cannot be rolled back.

Rolling Back to Previous Versions

Use the `InstallFromSoftwareIdentity()` method to reinstall from previous versions of firmware for a component that are stored in the Lifecycle Controller. Instead of downloading the DUP, the `InstallFromSoftwareIdentity()` creates a job and returns the job ID.

Remote Scheduling Types

Immediate Update

To immediately update component firmware, schedule the update and reboot jobs with start time as `TIME_NOW`. Scheduling a reboot or update is not required for updates to the Lifecycle Controller components like USC and Diagnostics. The updates are immediate for these components.

Scheduled Update

Specifying a scheduled start time for one or more jobs using the `SetupJobQueue()` method involves specifying a date and time value for the `StartTimeInterval` parameter. Optionally, a date and time value can be also be specified for the `UntilTime` parameter.

Specifying an `UntilTime` defines a maintenance window to run the updates within a time-bound slot. If the time window expires and the updates have not completed, any update jobs that are currently running will complete, but any unprocessed jobs whose scheduled start time has begun will be failed.

Setting the Scheduling Reboot Behavior

The `DCIM_SoftwareInstallationService.CreateRebootJob()` method takes one of the following reboot types as an input parameter and a reboot job ID is returned as an output parameter. The reboot Job ID is used as the first Job ID in the `JobArray` parameter of the `DCIM_JobService.SetupJobQueue()` method along with other update Job IDs.

- **Reboot 1 - Power cycle** — Performs the PowerCycle of the managed server that will power down the system and power it back up. This is not a graceful reboot. The system will power off the system without sending a shutdown request to an operating system running on the system. Only reboot type 1 will power on the system if the system is in an Off state, but A/C power is still applied.
- **Reboot 2 - Graceful reboot without forced shutdown** — Performs the Graceful Shutdown of the managed server and if the system is powered off within the PowerCycle Wait Time, it powers the system back up and marks the reboot job as **Reboot Completed**. If the system is not powered off within the PowerCycle WaitTime, the reboot job is marked as failed.
- **Reboot 3 - Graceful reboot with forced shutdown** — Performs the Graceful Shutdown of the managed server and if the system is powered off within the PowerCycle Wait Time, it powers the system back up and marks the reboot job as **Reboot Completed**. If the system is not powered off within the PowerCycle WaitTime, the system is Power Cycled.

Managing Part Replacement

The Part Replacement feature provides an automatic update of firmware, or configuration, or both of a newly replaced component, such as a PowerEdge RAID controller, NIC or power supply, to match that of the original part. This feature is disabled by default and may be enabled if required. It is a licensed feature and requires the Dell vFlash SD card. When a component is replaced and the Part Replacement feature is enabled, the actions taken by the Lifecycle Controller are displayed locally on the system monitor.

The presence of the vFlash SD card and configuration of Part Replacement related properties can be accomplished remotely through the Web services interface using the WS-Management protocol. For examples of command line invocations using WinRM see the *Lifecycle Controller 1.3 Web Services Interface Guide*. DCIM Profile specification and related MOF files are available at Dell TechCenter wiki in the DCIM Extension Library area (delltechcenter.com).

Important

- For a SAS card, only firmware update is supported. Configuration update is not supported because the attributes are not configurable on a SAS card.

- Part replacement is supported on modular systems with the following Broadcom and Intel devices:
 - Broadcom NetXExtreme II 5709 Quad Port Ethernet Mezzanine Card for M-Series
 - Broadcom NetXtreme II 57711 Dual Port 10 Gb Ethernet Mezzanine Card with TOE and iSCSI Offload for M-Series
 - Broadcom 57710 10 Gb Ethernet card
 - Intel Ethernet X520 10 GBE Dual Port KX4-KR Mezz

For more information on the supported cards, see *Dell Lifecycle Controller USC/USC-LCE 1.4 User's Guide*.

Validating vFlash presence Using WS-Management

To ensure that the system is equipped with a Dell-licensed vFlash card follow these steps:

- 1 Using an application, script or command line shell that can process WS-Management based web services requests, send a get instance request for the DCIM_LCEnumeration class instance with the InstanceID of "DCIM_LCEnumeration:CCR1".
- 2 If the vFlash is present, the output will have the following attribute values:
 - AttributeName = Licensed
 - CurrentValue = Yes
- 3 If the vFlash is not present on the system, or if it is not Dell-licensed, the output will have the following attribute values:
 - AttributeName = Licensed
 - CurrentValue = No

Using WS-Management to get/set Part Firmware and Configuration Update Attributes

To get the current **Part Firmware Update** and **Collect System Inventory On Restart** property values using WS-Management, an enumerate command request may be sent to get instances of the class DCIM_LCEnumeration. A DCIM_LCEnumeration instance object is returned per attribute where the AttributeName string property on the object will contain the name of the Part Replacement related property, such as **Part Firmware Update**. The

CurrentValue property contains the current setting of the property. See the Dell Lifecycle Controller Management Profile specification for specific attribute names and values. Some of them are:

- AttributeName = Part Configuration Update
- PossibleValues = Disabled, Apply always, Apply only if firmware match
- AttributeName = Part Firmware Update
- PossibleValues = Disable, Allow version upgrade only, Match firmware of replaced part

To configure a Part Replacement related property value, set and apply actions are requested using the WS-Management Web services protocol.

The set action is performed by invoking the **SetAttribute()** method on the `DCIM_LCService` class. The **SetAttribute()** method takes as input parameters the property names and values. Table 3-1 lists the values of the part firmware and configuration update:

Table 3-1. Part Firmware and Configuration Updates

Options	Values
Part Firmware Update	
Allow version upgrade only	If the input for the CurrentValue is Allow version upgrade only, firmware update on replaced parts will be performed if the firmware version of the new part is lower than the original part.
Match firmware of replaced part	If the input for the CurrentValue is Match firmware of replaced part, firmware on the new part will be updated to the version of the original part.
Disable	If the input is Disable, the firmware upgrade actions will not occur.
Part Configuration Update	
Apply always	The current configuration is applied if a part is replaced.
Apply only if firmware match	The current configuration is applied only if the current firmware matches with the firmware of a replaced part.
Disabled	The current configuration is not applied if a part is replaced.

The apply action is performed by invoking the `CreateConfigJob()` method on the `DCIM_LCService` class. The `CreateConfigJob()` method takes as parameters the scheduled start time (which can be `TIME_NOW`) and a reboot if required flag. A job ID is returned as a parameter and can be used to check on the job completion status.

Using Remote Firmware Inventory

Remote firmware inventory enables a WS-Management client to use the Web services interface provided by iDRAC to instantly retrieve the firmware and embedded software inventory of the system.

The firmware inventory feature will return an inventory of the installed firmware on devices on the system and the inventory of available BIOS/firmware on the iDRAC6 express card Lifecycle Controller. It also returns the inventory of both the currently installed version of BIOS /Firmware on the iDRAC6 Express card and the versions available for rollback (N and N-1 versions) that can be installed using the remote update Web services interface.

Instant Firmware Inventory

Instant firmware inventory allows you to run an inventory independent of whether the system is turned on or off. Traditionally, the system firmware inventory was performed by downloading an inventory collector onto the operating system, executing it locally, and then gathering the results. Instant firmware inventory allows you to inventory the host platform remotely from a WS-Management client, even if the host is not running an operating system. iDRAC user credentials used for the WS-Management request authentication requires Execute Server Command privileges to request firmware and embedded software inventory; it is not restricted to administrators. You can get a list of firmware for devices that are installed, and also the firmware that is available for rollback and reinstallation.

Supported Devices

Remote instant firmware inventory is supported for these devices and components:

- iDRAC6
- Storage controllers (RAID Series 6 and 7)

- Broadcom NICs and LOMs
- Power supplies
- BIOS
- OS Driver Pack
- USC
- Diagnostics

The instant firmware inventory class provides firmware inventory information on:

- The firmware installed on the supported devices
- The firmware versions available for installation for each device

Firmware Inventory Using WS-Management

The Dell Software Inventory profile defines the Dell CIM data model extensions that represent installed and available to be installed versions of firmware and embedded software on the server. The firmware inventory can be accessed using the WS-Management web services protocol.

To request for firmware inventory using Windows WS-Management:

- 1 Request inventory of the system using the WS-Management enumeration command for class `DCIM_SoftwareIdentity`.
- 2 Users that have administrator or Execute Server Command privileges can retrieve the firmware and embedded software inventory of the system.
- 3 Inventory instances are pulled up from the system in both system-off and system-on conditions.
- 4 The enumeration request will generate a WS-Management error when the UEFI system services are set to **Disabled**.
- 5 Requested inventories are collected as "Installed" and "Available" CIM instances.
- 6 The software currently installed on the component is listed as the "Installed Software Instance". The key property value of this instance, InstanceID represented as `DCIM: INSTALLED :< COMPONENTTYPE> :< COMPONENTID> :< Version>` and the status value of this instance is represented as "Installed"

- 7 The available software in the persistent storage is listed as the Available Software Instance. The key property value of the instance, InstanceID represented as `DCIM: AVAILABLE :< COMPONENTTYPE> :< COMPONENTID> :< Version>` and the status value of this instance is represented as "Available". Current installed software instances are also represented as available software instances.
- 8 Inventory instances provide input values for the update and rollback operations. To perform the update operation, pick the InstanceID value from the Installed Instance, `DCIM: INSTALLED :< comptype> :< compid> :< version>`. For the rollback operation pick the InstanceID Value from the Available instance, `DCIM:AVAILABLE:<comptype>:<compid>:<version>`. You will not be able to edit InstanceID values.



NOTE: If the "version string" property value of "Available Software Instance" is equal to the "Installed Software Instance," then the InstanceID value of that Available Software Instance should not be used for the rollback operation.

Important

- If Unified Server Configurator (USC) is being run on the system during the inventory operation, only "Installed Instances" are returned.
- There may be `DCIM_SoftwareIdentity` instances for hardware that was previously installed and then removed still listed in the inventory as "available."

Retrieving Hardware Inventory

Remote hardware configuration and inventory enables a WS-Management client to use the Web services interface provided by iDRAC to instantly retrieve the hardware inventory of a system. The inventory feature provides an inventory of the installed hardware devices on the system. The Inventory and configuration includes BIOS and UEFI attributes.

In addition, you can perform several hardware inventory tasks. Hardware related information is cached on the Lifecycle Controller persistent storage and is available to iDRAC and UEFI applications.

Enumerate the view classes of different system hardware like fans, power supplies, iDRAC, video controllers, CPU, DIMMs and PCI/PCIe to view their properties.

For more information on different hardware profiles, see the "Hardware Inventory Profiles" on page 65.

For more information on the easy-to-use names of the hardware components, see Table C-1.

Exporting Hardware Inventory

- To export the hardware inventory to an XML file, invoke the `ExportHWInventory()` method on the `DCIM_LCService` class.
- To store a copy of the factory defaults of a managed node, invoke the `ExportFactoryConfiguration()` method on the `DCIM_LCService` class. For more information on the schema, see the "Lifecycle Log Schema" on page 83.



NOTE: Store the XML file on an USB device or network share, or both the locations.

Viewing and Exporting Hardware Inventory after Resetting Lifecycle Controller

Incorrect inventory data is displayed or exported (into an XML file) after performing Delete Configuration & Reset Defaults. To view or export the correct hardware inventory data after resetting the Lifecycle Controller:



NOTE: After performing **Delete Configuration & Reset Defaults**, manually shut down the system.

- 1 Power on the system and wait for a couple of minutes for iDRAC to start functioning.
- 2 Since CSIOR is not enabled upon reset, press <F10> to launch USC so that the system inventory is collected. After USC launches, exit the wizard and wait for the system to reboot.
- 3 Disconnect the power cord and wait for 30 seconds. Reconnect the power cord and boot the system and invoke the `ExportHWInventory()` method on `DCIM_LCService` class.

Lifecycle Log

Lifecycle log shows the following information:

- Firmware update history based on device, version, and date.

- BIOS and NIC configuration changes.
- RAID configuration changes.
- Error message IDs. For more information, see error message registry at support.dell.com/manuals.
- Events (update and configuration only) based on severity, category, and date.



NOTE: The details of the configuration changes are not shown.

- Customer comments based on date.



NOTE: Lifecycle log is available even if the OS is not installed on the system and is independent of the power state of the system.

Exporting Lifecycle Log

Use this feature to export the Lifecycle Log information to an XML file. Store the XML file on an USB Device or network share, or both the locations.

To export the lifecycle log, invoke the `ExportLifecycleLog()` method on the `DCIM_LCService` class. For more information on the schema, see "Schema" on page 83.

Deleting Configuration and Resetting to Defaults

Use this feature to delete any sensitive data and configuration related information when you need to retire a managed node, reuse a managed node for a different application, or move a managed node to a non-secure location.



CAUTION: This feature resets the iDRAC to factory defaults, and deletes all iDRAC user credentials and IP address configuration settings. It also deletes lifecycle logs that contain the history of all the change events, firmware upgrades, and user comments, certificates, `ExportFactoryConfiguration` information, and firmware rollback files. It is recommended that you export the Lifecycle Log in a safe location before using this feature. After the operation, manually shut down and power on the system.



NOTE: Backup the lifecycle log and the `ExportedFactoryConfiguration` before deleting the configuration.

To delete configuration and reset to factory default, invoke the `LCWipe()` method on the `DCIM_LCService` class.

Managing NICs

Use this feature to get a detailed list of all the NICs embedded in the system and set the different attributes of a specific NIC.

For more information on the **Simple NIC** profile, see the "Simple NIC Profile" on page 58.

Displaying the NIC Inventory

- Perform the Enumerate operation on the **DCIM_NICView** class to display the instance properties of all (Broadcom and Intel) the NICs embedded in the system.
- Perform the Get operation on the class using the correct instance IDs of the required NIC to display the related properties.

Displaying the NIC Attributes

- Perform the Enumerate operation on one of the **DCIM_NICAttribute** classes (**DCIM_NICEnumeration**, **DCIM_NICInteger**, and **DCIM_NICString**) to display all available attributes and possible values of all the NICs embedded in the system.
- Perform the Get operation on the one of the **DCIM_NICAttribute** classes to display the NIC attributes. For specific sub-class attribute information, use the correct instance ID along with the attribute name listed in the sub-class.

Setting the NIC Attributes

To set the attributes:

- 1** Identify the applicable instance ID.
- 2** Confirm the **IsReadOnly** field is set to false.
- 3** Before invoking the **SetAttribute()** or **SetAttributes()** method, note the instance information that you got in step 1 and prepare the input parameters.
- 4** Invoke the **SetAttribute()** or **SetAttributes()** method.
- 5** Run the Get command on the attribute to see updated value in the pending field.

- 6 Before invoking the `CreateTargetedConfigJob()` method, construct the input parameters (for example, `Target`, `RebootType`, `ScheduledStartTime`, `UntilTime`, and so on) and use the correct Fully Qualified Device Descriptor (FQDD) of the NIC for `Target`.



NOTE: See the Simple NIC Profile document at delltechcenter.com/page/DCIM.Library to see the list of all the supported input parameters.

- 7 Invoke the `CreateTargetedConfigJob()` method to apply the pending values. If this method is successful, the system must return a job ID for the configuration task you created.




NOTE: The system must reboot to execute the task of setting the attribute or attributes.

- 8 You can query the status of the jobID output using the job control profile methods.
- 9 Repeat step 1 to confirm successful execution of the method.

Deleting the Pending Values

To delete the pending values:

- 1 Before invoking the `DeletePendingConfiguration()` method in `DCIM_JobService` class, construct input parameters and use the correct Fully Qualified Device Descriptor (FQDD) of the NIC.
 -  **NOTE:** You can only delete pending data before creating a target job. After the target job is created, you cannot run this method. If required, you can invoke the `DeleteJobQueue()` method to delete the job and clear the pending values.
- 2 Invoke the `DeletePendingConfiguration()` method.
- 3 You can confirm the deletion based on the method return code value that is returned.

Managing vFlash SD Card

vFlash is the Non-volatile Random Access Memory (NVRAM) flash located on a SD card that is inserted into the SD card reader controlled by the iDRAC service processor. The card is used as a feature enabling license key for several Lifecycle Controller features including Part Replacement. Additionally, the vFlash SD card is the storage location for partitions that you can define and configure to be available to the system as a USB device. You can create a bootable USB device that is displayed as an option under the BIOS boot menu.

For more information on the vFlash SD card, see the "Persistent Storage Profile" on page 61.

Displaying the Inventory of vFlash SD Card

Perform the Enumerate operation on the `DCIM_VFlashView` class to display all the properties of the vFlash SD card; such as Available size, Capacity, Licensed, and Health, Enable/Disable state, Initialized state, and Write protected state.

Displaying the Partitions on vFlash SD Card

Perform the Enumerate operation on the `DCIM_OpaqueManagementData` class to display all the partitions and their properties; such as partition ID, its size and data format.

Creating and Modifying a Partitions on vFlash SD Card

- 1 Perform the enumerate operation on the `DCIM_OpaqueManagementData` class to get the list of current partitions.
- 2 Before you invoke the `CreatePartition()` method in `DCIM_PersistentStorageService` class, construct the input parameters.
- 3 Invoke the `CreatePartition()` method. For example, if a job is created successfully, code 4096 is returned.
- 4 Invoke the `CreatePartition()` method to form a bootable image. This creates a bootable partition from image stored on server shares like NFS, CIFS, and FTP.
- 5 Query the status of the jobID output using the job control profile methods.
- 6 Repeat step 1 to confirm successful execution of the method.
- 7 Set the created bootable partition as an option under the BIOS boot menu and boot to the image stored on the partition.
- 8 Invoke the `AttachPartition()` method, to view and modify the contents of partitions.
- 9 Invoke the `Accesstype()` and `FormatType()` methods, to change the access type and the format type of the created partitions.

Managing RAID Configuration

Use the RAID configuration feature to get the properties of the RAID controller, physical disks, and the enclosures attached to the system. You can configure different attributes of the physical and virtual disks using the available methods.

For more information on the RAID profile, see the "RAID Profile" on page 63.


Displaying the RAID Controllers

- Perform the Enumerate operation on the `DCIM_ControllerView` class to display the instance properties of all the RAID controllers attached to the system.

- Perform the Get operation on the `DCIM_ControllerView` class using the correct instance ID of the required RAID controller to display the related properties.

Creating a Virtual Disk

To create the virtual disk:

- 1 Find out the RAID configurations in the system using the `GetRAIDLevels()` method in `DCIM_RAIDService` class.
- 2 Select the physical disk(s) on which you need to create the virtual disk based on the IDs gathered using the `GetAvailableDisks()` method in `DCIM_RAIDService` class.
- 3 Check the available sizes and default virtual disk parameters for the required RAID level and physical disk using the `CheckVDValues()` method in `DCIM_RAIDService` class.
 **NOTE:** The `CheckVDValues()` method does not show the Span details correctly for RAID-10.
- 4 Construct the input parameters before you invoke the `CreateVirtualDisk()` method.
- 5 Invoke the `CreateVirtualDisk()` method.
- 6 Check the output parameters (return code values) for the selected method. The InstanceID of the pending virtual disk is an output parameter and the return code value is returned if the method is successful. For example, if the method is successful, code 0 is returned.
- 7 Before invoking the `CreateTargetedConfigJob()` method, construct the input parameters and use the correct Fully Qualified Device Descriptor (FQDD) for the controller.
- 8 Invoke the `CreateTargetedConfigJob()` method to apply the pending values.
- 9 Query the status of the jobID output using the job control profile methods.
- 10 The system is rebooted based on the time specified.
- 11 Enumerate the `DCIM_VirtualDiskView` class to view the virtual disk created earlier.

Managing BIOS and Boot Configuration

Use the BIOS and boot configuration feature to configure BIOS properties and to perform operations such as changing the boot source and boot order. For more information, see the "BIOS and Boot Management Profile" on page 59.

Displaying the Inventory of BIOS Attributes

Perform the Enumerate operation on the DCIM_BIOSEnumeration class to view all available instances of the BIOS attributes in a system.

Setting the BIOS Attributes

To set the attributes:

- 1 Identify the applicable instance ID.
- 2 Confirm the IsReadOnly field is set to false.
- 3 Before invoking the `SetAttribute()` or `SetAttributes()` method, note the instance information that you got in step 1 and prepare the input parameters.
- 4 Invoke the `SetAttribute()` or `SetAttributes()` method.
- 5 Examine output parameters.
- 6 Before invoking the `CreateTargetedConfigJob()` method, prepare input parameters (for example, `RebootJobType`, `ScheduledStartTime`, `UntilTime`, `Job`, and so on) and use the correct BIOS FQDD.
- 7 Invoke the `CreateTargetedConfigJob()` method.




NOTE: The system must reboot to execute the task of setting the attribute or attributes.

- 8 Query the status of the jobID output using the job control profile methods.
- 9 Repeat step 1 to confirm successful execution of the method.

One Time Boot

Use the boot management methods to perform one time boot to a BIOS boot device. If you try to one time boot to a vFlash partition that is not attached, Remote Services automatically attaches it and returns a job ID. You can query the job using this ID.

To set one time boot:

- 1 Perform the enumerate operation on the `DCIM_BootConfigSetting` class and identify the `ElementName` field containing `BootSeq` and corresponding `InstanceID`.
- 2 Perform the `Enumerate` operation on the `DCIM_BootSourceSetting` class and identify the boot source `InstanceID`. The `CurrentEnabledStatus` attribute of each instance identifies whether it is enabled or disabled.
- 3 Before invoking the `ChangeBootOrderByInstanceID()` method, note the instance information you got in step 1 and step 2 and prepare the input parameters.
- 4 Invoke the `ChangeBootOrderByInstanceID()` method.
- 5 Examine output parameters.
- 6 Before invoking the `CreateTargetedConfigJob()` method, prepare input parameters (for example, `RebootJobType`, `ScheduledStartTime`, `UntilTime`, `Job`, and so on) and use the correct BIOS `FQDD`.
- 7 Invoke the `CreateTargetedConfigJob()` method.
 **NOTE:** The system must reboot to execute the task of setting the attribute or attributes.
- 8 Query the status of the `jobID` output using the job control profile methods.
- 9 Repeat step 2 to confirm successful execution of the method.

Using Job Control

Use this feature to do the following:


- Reporting all Jobs - Enumerate the **DCIM_ConcreteJob** class to report all the jobs.
- Reporting scheduled Jobs - Enumerate the **DCIM_ConcreteJob** class with a selection filter of `JobStatus=Scheduled` to generate a report of all the scheduled jobs.
- Scheduling Jobs and Job Queues - You can run multiple jobs in a single reboot of the system using the **SetupJobQueue()** method on the **DCIM_JobService** class. If you create a job using the **CreateTargetedConfigJob()** method without setting the start time, use the **SetupJobQueue()** method to set the schedule and order of execution. If the start time was set in the **CreateTargetedConfigJob()** method, it cannot be bundled with the other jobs, and the job is setup for execution at the time that was specified.
- Deleting Jobs - Delete a specified existing job using the **DeleteJobQueue()** method on the **DCIM_JobService** class.

For more information on job control, see the "Job Control Profile" on page 67.

Scheduling Separate Jobs for Multiple Actions

To schedule separate jobs for multiple actions (in the following example, BIOS and NIC update and NIC configuration):

- 1 Invoke the **InstallFromURI()** method for the BIOS and NIC firmware update packages.
The method downloads the BIOS and NIC updates and creates a job ID for each device update job.
- 2 Set the NIC attributes for a NIC (for example, Embedded NIC 1) and create a targeted job for this set. The method returns a job ID.
- 3 Take these job IDs and use the **SetupJobQueue()** method to schedule these jobs so that they are executed in the order specified at the specified start time.

 **NOTE:** To have the iDRAC reboot the system automatically at the scheduled time, create a reboot job (specifying type of reboot, graceful or power cycle) and include the reboot job ID in the list of jobs specified in the **SetupJobQueue()** method invocation. If a reboot job is not included in the Job Queue setup, the jobs are ready to run at the scheduled start time but rely on an external actor to restart the system and get the job execution started.

Running Multiple Target Jobs

To run multiple target jobs (for example, setting NIC attributes on multiple NICs) at one time:

- 1 Configuring Embedded NIC 1:
 - a Set the NIC attributes for Embedded NIC 1.
 - b Create a targeted config job for Embedded NIC 1 with a scheduled start time of `TIME_NOW`, but ensure not to schedule a reboot.
- 2 Configuring Embedded NIC 2:
 - a Set NIC attributes for Embedded NIC 2
 - b Create a targeted config job for Embedded NIC 2 with a scheduled start time of `TIME_NOW`, but ensure not to schedule a reboot.
- 3 Set NIC attributes for Embedded NIC 3, create targeted job for Embedded NIC 3 with a scheduled start time of `TIME_NOW` and also specify a reboot type.

The iDRAC restarts the system according to the method defined by the reboot type, and all the jobs are executed at one time.

Specifying the Start time and Until time

The **CreateTargetedConfigJob()** and **SetupJobQueue()** methods accept the start times **ScheduledStartTime** and **StartTimeInterval** and until parameter. The parameter data type is CIM date-time. If the **StartTime** parameter is null, the action is not started. The date-time data type is defined in the format as follows:

YYYYMMDDhhmmss

Where:

- YYYY is the year
- MM is the month

- DD is the day
- hh is the hour
- mm is the minute
- ss is the second

For example, 20090930112030 — You must type the date and time in this format for all the Lifecycle Controller updates, set attributes and **CreateTargetedConfigJob()** methods on different service classes. TIME_NOW is a special value that represents "running the tasks immediately".

Remote Services Profiles

This section provides high-level information on the individual profiles.

For more information on the profiles and the related MOFs, see delltechcenter.com/page/DCIM.Library

For examples of WinRM and WS-Management command line invocations, see:

- delltechcenter.com/page/Lifecycle+Controller
- *The Lifecycle Controller 1.4 Web Services Interface Guide*

Operating System Deployment Profile

Table 4-1 lists the classes, functions, operations, and methods under the Operating System Deployment profile.

Table 4-1. Operating System Deployment Profile

Class Name	Operations	Methods
DCIM_OSDeploymentService	Get Enumerate Invoke	See "Operating System Deployment Methods" on page 55
CIM_ConcreteJob	Get Enumerate	NA

Operating System Deployment Methods

- The **GetDriverPackInfo()** method returns the list of operating systems that you can install on the server using the embedded device drivers available in the Dell Lifecycle Controller.
- The **UnpackAndAttach()** method extracts the drivers for the selected operating system to a USB device that is attached locally to the server for the specified time interval.
- The **DetachDrivers()** method detaches the USB device containing the drivers from the host server.

- The **UnpackAndShare()** method extracts the drivers for the selected operating system, and copies them to the specified network share.
- The **BootToNetworkISO()** method is used to boot the system to an ISO image located on a CIFS or NFS network share.
- The **DetachISOImage()** method detaches the ISO Image from the host server.
- The **BootToPXE()** method is used to boot the server using the Preboot Execution Environment (PXE) mechanism.
- The **DownloadISOToVFlash()** method is used to download the pre-OS ISO Image to the vFlash SD card.
- The **BootToISOFromVFlash()** method is used to boot to the vFlash pre-OS image that was already downloaded.
- The **DetachISOFromVFlash()** detaches the ISO Image from the host server.
- The **DeleteISOFromVFlash()** method deletes the ISO Image from vFlash SD card.

Lifecycle Controller Management Profile

Table 4-2 lists the classes, functions, operations, and methods under the Lifecycle Controller management profile.

Table 4-2. Lifecycle Controller Management Profile

Class Name	Operations	Methods
DCIM_LCService	Get Enumerate Invoke	See "Auto-Discovery Methods" on page 57, "Lifecycle Log Methods" on page 57, and "Hardware Inventory Methods" on page 58
DCIM_LCString	Get Enumerate	NA
DCIM_LCEnumeration	Get Enumerate	NA

Auto-Discovery Methods

- The **SetAttribute()** method is used to set the value of a single attribute.
- The **SetAttributes()** method is used to set the values of multiple attributes.
- The **CreateConfigJob()** method is used to apply the pending values set by the **SetAttribute()** and **SetAttributes()** methods.
- The **ReInitiateDHS()** method is used to reinitiate the provisioning server discovery and handshake.
- The **ClearProvisioningServer()** method is used to clear the provisioning server values.
- The **DownloadServerPublicKey()** method is used to download the server public key to the Lifecycle Controller (LC).
- The **DownloadClientCerts()** method is used to download the client private certificate, password, and root certificate to LC.
- The **DeleteAutoDiscoveryClientCerts()** method is used to delete the auto-discovery client certificates and private keys previously downloaded.
- The **SetCertificateAndPrivateKey()** method is used to update iDRAC certificate and private key pairs using the contents of a PKCS#12 file.
- The **SetPublicCertificate()** method is used to update a public SSL Certificate on the iDRAC.
- The **DeleteAutoDiscoveryServerPublicKey()** method is used to delete the auto-discovery server public keys previously downloaded.

Lifecycle Log Methods

- The **LCWipe()** method is used to wipe all configurations from the Lifecycle controller before the system is retired.
- The **ExportLifecycleLog()** method is used to export the log from the Lifecycle Controller to a file on a remote share.
- The **InsertCommentInLCLog()** method is used to insert additional user comments into the Lifecycle Controller log.

Hardware Inventory Methods

- The `ExportHWInventory()` method is used to export the hardware inventory from the Lifecycle Controller to a file on a remote share.
- The `ExportFactoryConfiguration()` method is used to export the factory configuration from the Lifecycle Controller to a file on a remote share.

Simple NIC Profile

Table 4-3 lists the classes, functions, operations, and methods under the Simple NIC profile.

Table 4-3. Simple NIC Profile

Class Name	Functions	Operations	Methods
DCIM_NICService	This is the central class. It is called to modify the NIC attributes.	Get Enumerate Invoke	See "Simple NIC Methods"
DCIM_NICView	Use this class to display the instanceIDs and other properties of the LOMs and add-in NICs in the system.	Get Enumerate	NA
DCIM_NICAttribute	— This class displays the output for the following BIOS sub-classes:		
• DCIM_NICEnumeration	Use this sub-class to display the properties of NIC enumeration instances.	Get Enumerate	SetAttribute() SetAttributes()
• DCIM_NICInteger	Use this sub-class to display the properties of NIC integer instances.	Get Enumerate	SetAttribute() SetAttributes()
• DCIM_NICString	Use this sub-class to display the properties of NIC string instances.	Get Enumerate	SetAttribute() SetAttributes()

Simple NIC Methods

These methods are used to apply attributes to LAN on motherboards and add-in NICs in the system. Each of the methods have their own set of input and output parameters. The methods have specific return code values. There are four different methods under the NIC service class:

- The **SetAttribute()** method is used to set or change the value of a NIC attribute.
- The **SetAttributes()** method is used to set or change the values of a group of attributes.
- The **CreateTargetedConfigJob()** method is used to apply the pending values created by the **SetAttribute** and **SetAttributes** methods. The successful execution of this method creates a job for application of pending attribute values.



NOTE: Subsequent calls to the **CreateTargetedConfigJob()** method after the first **CreateTargetedConfigJob()** method results in an error until the first job is completed. If you invoke the **CreateTargetedConfigJob()** method multiple times, older requests are overwritten or lost.

- The **DeletePendingConfiguration()** method cancels the pending configuration (created using the **SetAttribute** and **SetAttributes** methods) changes made before the configuration job is created with **CreateTargetedConfigJob()**.

BIOS and Boot Management Profile

Table 4-4 lists the classes, functions, operations, and methods under the BIOS and Boot Management profile.

Table 4-4. BIOS and Boot Management Profile

Class Name	Functions	Operations	Methods
BIOS Management			
DCIM_BIOSService	Use this central class to modify the BIOS attributes.	Get Enumerate Invoke	See "BIOS and Boot Management Methods" on page 60

Table 4-4. BIOS and Boot Management Profile (continued)

Class Name	Functions	Operations	Methods
DCIM_BIOSEnumeration	Use this sub-class to display the properties of BIOS enumeration instances.	Get Enumerate	SetAttribute() SetAttributes()
DCIM_BIOSInteger	Use this sub-class to display the properties of BIOS string instances.	Get Enumerate	SetAttribute() SetAttributes()
DCIM_BIOSString	Use this sub-class to display the properties of BIOS integer instances.	Get Enumerate	SetAttribute() SetAttributes()
Boot Management			
DCIM_BootConfigSetting	This class has the following boot list instances: <ul style="list-style-type: none"> • IPL • BCV • UEFI • vFlash • OneTime 	Get Enumerate Invoke	ChangeBootSourceState() ChangeBootOrderByInstanceID()
DCIM_BootSourceSetting	Use this class to change the boot source and the boot order of the related devices.	Get Enumerate	NA

BIOS and Boot Management Methods

The methods are used to apply attributes and change the boot configurations in the system. Each of the methods have their own set of input and output parameters. The methods have specific return code values. The following methods are used under BIOS and boot management:

- The **SetAttribute()** method is used to set or change the value of a BIOS attribute.
- The **SetAttributes()** method is used to set or change the values of a group of attributes.

- The **ChangeBootSourceState()** method is used to change the EnabledState of a boot source from either disable to enable and enable to disable.
- The **ChangeBootOrderByInstanceID()** method is used to change the boot order of the boot sources from the boot list instances (IPL, BCV, UEFI). This method expects boot source instances from one list only, so to change the boot order of multiple instances, call this method multiple times with instances from different boot lists.
- The **CreateTargetedConfigJob()** method is used to apply the pending values created by the **SetAttribute()** and **SetAttributes()** methods. The successful execution of this method creates a job for application of pending attribute values. This method is also used to set the boot order, source state, and one time boot device.
 - ✍ **NOTE:** Subsequent calls to the **CreateTargetedConfigJob()** method after the first **CreateTargetedConfigJob()** method results in an error until the first job is completed. However, the you can delete the current job and create a new job using **CreateTargetedConfigJob()**.
- The **DeletePendingConfiguration()** method cancels the pending configuration (created using the SetAttribute and SetAttributes methods) changes made before the configuration job is created with **CreateTargetedConfigJob()**.

Persistent Storage Profile

Table 4-5 lists the classes, functions, operations, and methods under the persistent storage profile.

Table 4-5. Persistent Storage Profile

Class Name	Functions	Operations	Methods
DCIM_PersistentStorageService	Use this central class to define the extrinsic methods.	Get Enumerate Invoke	See "vFlash SD Card Methods"
DCIM_vFlashView	Use this class to display the different instance IDs and related properties of all the vFlash SD cards attached to a system.	Get Enumerate	NA

Table 4-5. Persistent Storage Profile

Class Name	Functions	Operations	Methods
DCIM_OpaqueManagementData	Use this sub-class to display the available partitions on a specific vFlash SD card.	Get Enumerate	NA

vFlash SD Card Methods

- The **InitializeMedia()** method is used to format the vFlash SD card.
- The **VFlashStateChange()** method is used to enable or disable the vFlash SD card.
- The **CreatePartition()** method is used to create a new partition on a vFlash SD card.
- The **CreatePartitionUsingImage()** method is used to create a new partition using an image file (available in the .img or .iso format.)
- The **DeletePartition()** method is used to delete a vFlash SD card partition.
- The **FormatPartition()** method is used to format the selected vFlash SD card partition.
- The **ModifyPartition()** method is used to modify the partitions on the vFlash. This depends on the type of partition - Floppy, Hard Disk, or CD.
- The **AttachPartition()** method is used to attach one or more partitions as a virtual USB mass storage device.
- The **DetachPartition()** method is used to detach one or more partitions that are being used a virtual USB mass storage device.
- The **ExportDataFromPartition()** method is used to copy or export the contents of a vFlash SD card partition to a local or remote location as an image file in the .img or .iso format.

RAID Profile

Table 4-6 lists the classes, functions, operations, and methods under the RAID profile.

Table 4-6. RAID Profile

Class Name	Functions	Operations	Methods
DCIM_RAIDService	This is the central class. It defines the extrinsic methods.	Get Enumerate Invoke	See "RAID Methods"
DCIM_ControllerView	Use this class to display the different instance IDs and related properties of the controllers attached to a system.	Get Enumerate	NA
DCIM_PhysicalDiskView	Use this class to display the different instance IDs and related properties of the physical disks attached to a system.	Get Enumerate	NA
DCIM_VirtualDiskView	Use this class to display the different instance IDs and related properties of the virtual disks created.	Get Enumerate	NA
DCIM_EnclosureView	Use this class to display the different instance IDs and related properties of the enclosures attached to a system.	Get Enumerate	NA
DCIM_Attribute			
• DCIM_EnumAttribute	Use this sub-class to display the properties of RAID enumeration instances.	Get Enumerate	NA
• DCIM_IntegerAttribute	Use this sub-class to display the properties of RAID integer instances.	Get Enumerate	NA
• DCIM_StringAttribute	Use this sub-class to display the properties of RAID string instances.	Get Enumerate	NA

RAID Methods

The RAID methods are used to apply attributes to different RAID components. Each of the methods have their own set of input and output parameters. The methods have specific return code values. The different methods under the RAID service class are:

- The **AssignSpare()** method is used to assign a physical disk as a dedicated hot spare for a virtual disk, or as a global hot spare.
- The **ResetConfig()** method is used to delete all virtual disks and un-assign all hot spare physical disks. All data on the existing virtual disks are lost.



NOTE: The virtual disks that are not imported on the foreign physical disks, are not deleted.

- The **ClearForeignConfig()** method is used to prepare any foreign physical disks for inclusion in the local configuration.



NOTE: All the data on the foreign physical disks are lost.

- 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 **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 **GetDHSDisks()** method is used to find out the possible choice of drives to be a dedicated hot-spare for the identified virtual disk.
- The **GetRAIDLevels()** method is used to find out the possible choice of RAID Levels to create virtual disks. If the list of physical disks is not provided, this method operates on all connected disks.
- The **GetAvailableDisks()** method is used to find out the possible choice of drives to create virtual disks.
- The **CheckVDValues()** method is used to find out the size of the virtual disks and the default settings for a given RAID level and set of disks.
- The **SetControllerKey()** method sets the key on controllers that support encryption of the drives.
- The **LockVirtualDisk()** method encrypts the identified virtual disk. The virtual disk must reside on physical disks that support encryption while the encryption is enabled on them.

- The `CreateTargetedConfigJob()` method is used to apply the pending values created by other methods. The successful execution of this method creates a job for application of pending attribute values.



NOTE: Subsequent calls to the `CreateTargetedConfigJob()` method after the first `CreateTargetedConfigJob()` method results in an error until the first job is completed.

- The `DeletePendingConfiguration()` method cancels the pending configuration (created using the other methods) changes made before the configuration job is created with `CreateTargetedConfigJob()`.

Hardware Inventory Profiles

Table 4-7 lists the classes, functions, operations, and methods for different hardware on the managed node.

Table 4-7. Hardware Inventory Profiles

Class Name	Functions	Operations	Methods
CPU Profile			
DCIM_CPUView	Use this class to get the instance information of all the CPUs and associated cache available in the system.	Get Enumerate	NA
Fan Profile			
DCIM_FanView	Use this class to get the instance information of all the fans available in the system.	Get Enumerate	NA
iDRAC Profile			
DCIM_IDRACCardView	Use this class to get the instance information of all iDRAC cards available in the system.	Get Enumerate	NA
Memory Profile			
DCIM_MemoryView	Use this class to get the instance information of all memory modules available in the system.	Get Enumerate	NA
PCI Profile			

Table 4-7. Hardware Inventory Profiles (continued)

Class Name	Functions	Operations	Methods
DCIM_PCIDeviceView	Use this class to get the instance information of all PCI devices available in the system.	Get Enumerate	NA
Video Profile			
DCIM_VideoView	Use this class to get the instance information of all the video controllers available in the system.	Get Enumerate	NA
Power Supply Profile			
DCIM_PowerSupplyView	Use this class to get the instance information of all the power supply units available in the system.	Get Enumerate	NA
System View Profile			
DCIM_SystemView	Use this class to get the general details about the system like System Manufacturer, Model, Service Tag, Total Memory, BIOS Version, System ID, Asset Tag, Power State, and so on.	Get Enumerate	NA

Job Control Profile

Table 4-8 lists the classes, functions, operations, and methods under the **Job Control Profile**.

Table 4-8. Job Control Profile

Class Name	Operations	Methods
DCIM_JobControlService	Get Enumerate	See "Job Control Methods"
DCIM_ConcreteJob	Get Enumerate	NA

Job Control Methods

The methods are used to setup job queue and deleting the jobs from the job queue.

- The **SetupJobQueue()** method is used for creating a job queue containing one or more jobs that are executed in a specific order within the queue.
- The **DeleteJobQueue()** method is used for deleting jobs from the job queue.

Troubleshooting

This section describes some of the error messages commonly generated by Remote Services, and provides suggestions for resolving the errors.

For more information on the error message IDs and the recommended actions, see *Dell Lifecycle Controller Remote Services Error Messages and Troubleshooting List* on support.dell.com/manuals. To view the error message and related information, select the error message ID from the **Error Message ID** drop-down list. Additionally, you can download the detailed error message registry from delltechcenter.com/page/Lifecycle+Controller.

Error Messages

Table 5-1 describes the error messages commonly generated by Remote Services, and provides suggestions for resolving the errors.

Table 5-1. Lifecycle Controller - Remote Services Error Messages and Resolutions

Error Message	Resolution
General failure	An error has occurred. No other details are available at this time. <ol style="list-style-type: none"> 1 Run the command again. 2 Reset iDRAC and run the command.
Lifecycle Controller is being used by another process	Lifecycle Controller is currently locked by another process. Ensure that the process is completed before attempting to run another command. <ol style="list-style-type: none"> 1 Run the command again after sometime. 2 Ensure that USC or DUP is not running. 3 Reset iDRAC and run the command

Table 5-1. Lifecycle Controller - Remote Services Error Messages and Resolutions

Error Message	Resolution
Cannot access Driver Pack partition in Lifecycle Controller.	Driver Pack partition in Lifecycle Controller is not accessible. The Lifecycle Controller might be corrupted. Reset iDRAC and run the command.
Driver Pack not found in Lifecycle Controller	No Driver Pack in Lifecycle Controller. Update the Driver Pack manually using USC or DUP and then run the command again.
Cannot allocate memory	Unable to dynamically allocate memory to perform the task. Reset iDRAC and run the command.
Driver Pack does not have drivers for the selected operating system.	Lifecycle Controller does not have any drivers for the selected operating system. The installation will have to use the native drivers present on the operating system media.
Cannot create USB device to copy drivers for the selected operating system.	Unable to create USB device to copy drivers for selected operating system. iDRAC may not be operating normally Reset iDRAC and run the command again.
Cannot mount USB device to copy drivers for the selected operating system.	Unable to access the newly created USB device to copy drivers for selected operating system. iDRAC may not be operating normally Reset iDRAC and run the command again
Unable to expose USB device containing operating system drivers to host system.	Unable to expose the newly created USB device (with drivers for selected operating system) to the host server. iDRAC may not be operating normally. Reset iDRAC and run the command again.
Mount network share failed - incorrect username or password.	Unable to mount the network share using the credentials specified in the command. Either username or password is incorrect. Run the command again with correct username and password.

Table 5-1. Lifecycle Controller - Remote Services Error Messages and Resolutions

Error Message	Resolution
Mount network share failed - incorrect IP address or share name.	Unable to mount the network share using the credentials specified in the command. Either IP address or share name is incorrect. Run the command again with correct IP address and share name.
Exposing ISO image as internal device to the host system failed.	Unable to expose the ISO image as internal CD device to the host system. The ISO file is no longer present, network errors are preventing access to the ISO file, or iDRAC may not be operating normally. Reset iDRAC and run the command again.
Unable to locate the ISO image on the network share point.	Unable to find the ISO file specified in the network share. Ensure that you have specified the correct path to the ISO file in the command and all other user credentials are correct. Run the command again with correct path to ISO file.
The fork() command for a child process to do the task failed	Failed to execute fork() system call to perform the task in a child process. iDRAC may not be operating normally. Reset iDRAC and run the command
Unable to get size or label from Driver Pack for selected operating system.	Unable to get the size or label for selected operating system from the Driver Pack present in Lifecycle Controller. The driver pack may be corrupt. Update the driver pack using USC or DUP and run the command again
Unable to boot to ISO image	Bootting to ISO has failed. Either BIOS was unable to boot to the ISO image or provider did not get a response in 5 minutes from BIOS on successful boot to ISO image. 1 Ensure there is no POST error that resulted in user interaction (Press F1 to continue or F2 to run setup). 2 Reset iDRAC and run the command
Unable to detach ISO image from the host	Unable to detach ISO image from the host. Either the image may have already detached or iDRAC may not be operating normally. Reset iDRAC to automatically detach the ISO image.

Table 5-1. Lifecycle Controller - Remote Services Error Messages and Resolutions

Error Message	Resolution
Unable to continue with DetachISOImage - another command is in the process of exposing ISO Image and booting to it.	Cannot continue with DetachISOImage because another command is in the process of exposing ISO image and booting to it. See ConcreteJob status to ensure that the current running process is complete and then run DetachISOImage.
Unable to continue with DetachDrivers - UnPackAndAttach is in progress	Wait until UnpackAndAttach finishes and then run DetachDrivers.
Unable to detach USB device containing operating system drivers.	Detaching the USB device (that contains drivers for the operating system installation) from the host has failed. The device may have been detached already or IDRAC may not be operating normally. Reset iDRAC to detach this device automatically.
Unable to continue with BootToPXE - another command is running.	Unable to continue with BootToPXE command because another process is using Lifecycle Controller. See ConcreteJob status to ensure that the current running process is complete and then run BootToPXE.
Copying drivers for selected operating system failed.	Copying drivers for selected operating system failed. The Driver Pack may be corrupt. Update the Driver Pack either remotely using WS-Management or locally using USC or DUP and then run the command again.
Hash verification on the ISO image failed.	Hash verification on the ISO image has failed. The hash value specified in the command is either not correct or the ISO image has been changed. 1 Verify that the hash value specified in the command is correct. 2 Ensure that the ISO has not been changed - replace the ISO image on the share and run the command again.
Driver Pack config file not found in Lifecycle Controller. Driver Pack might be corrupt.	Update the Driver Pack either remotely using WS-Management or locally using USC or DUP and then run the command again.

Table 5-1. Lifecycle Controller - Remote Services Error Messages and Resolutions

Error Message	Resolution
Invalid value for ExposeDuration - must be 60-65535 seconds	<p>The value specified for ExposeDuration is out of range. It must be 60-65535 seconds</p> <p>Run the command again with ExposeDuration value 60 to 65535 seconds.</p>
Copying operating system drivers to network share failed	<p>The share may be read-only or the driver pack present in Lifecycle Controller may be corrupt.</p> <p>Ensure that the network share has write permission, or update the Driver Pack either remotely using WS-Management or locally using USC or DUP and then run the command again.</p>
Unable to detach ISO image from the system	<p>Cannot continue with DetachISOImage because system does not have attached ISO image.</p> <p>Do not run DetachISOImage command.</p>
Installed BIOS version does not support this method.	<p>The system has an older version of BIOS that does not support this method. Install the latest version of BIOS to use this method.</p> <p>Update the BIOS to version 1.2 or later and then run the command again.</p>
Unable to continue with BootToPXE - ISO image is attached to the system.	<p>Unable to continue with BootToPXE command because system has an ISO image attached. Detach the ISO image before continuing with BootToPXE.</p> <p>Run DetachISOImage command and then run BootToPXE.</p>
Lifecycle Controller is disabled	<p>Lifecycle Controller is disabled on the system, so none of the remote enablement OSD commands will work. Ensure Lifecycle Controller is enabled before running any command</p> <p>Reboot the system and enable System Services using CTRL+E option in the POST</p>

Table 5-1. Lifecycle Controller - Remote Services Error Messages and Resolutions

Error Message	Resolution
Boot to ISO Image has been cancelled by user using CTRL+E option on the server	User has cancelled system services by using CTRL+E option during POST. This has effectively cancelled the WSMAN request to boot to ISO Do not cancel system services using CTRL+E during POST when system is rebooting to the ISO

Auto-Discovery LCD Messages

Table 5-2 lists the LCD messages that are displayed while performing Auto-Discovery operations.

Table 5-2. Auto-Discovery Messages

Message 1	Message 2
Stopped	NA
Running	see Table 5-3
Suspended	see Table 5-3
Complete	NA

Table 5-3 lists the LCD messages and resolutions. These messages are shown in combination with the messages listed in Table 5-2. For example, when a Auto-Discovery operation is running and an administrative account is enabled, the messages Running and Blocked and Admin Account Enabled are shown on the LCD screen.

Table 5-3. Auto-Discovery Messages

Message 2	Resolutions
Stopped (default)	N/A
Started	N/A
Auto Discovery disabled	Enable auto-discovery.
Blocked Admin Account Enabled	Disable all the administrative accounts.

Table 5-3. Auto-Discovery Messages

Message 2	Resolutions
Blocked Active Directory Enabled	Disable the active directory.
Blocked IPv6 Enabled	Disable IPv6.
Blocked No IP on NIC	Enable the NIC.
No Provisioning Server Found	Check the value of psinfo in the BIOS. If the psinfo is not configured in the BIOS, check if the DHCP option is enabled and/or DNS server configuration is valid.
Blocked Provisioning Server Unreachable/Invalid address	Check the value of psinfo in the BIOS.
No Service Tag	Boot the server. If the problem persists, contact technical support.
SSL connection failed no service at IP/port	Check the value of psinfo in the BIOS, or vendor option on the DHCP server.
SSL Connection refused	Check the value of psinfo in the BIOS, or vendor option on the DHCP server.
SSL connection failed (server authentication)	The server certificate is invalid or not signed by the trusted server CA certificate installed on iDRAC. Either replace the provisioning server certificate or upload a new server certificated on the iDRAC.
SSL connection failed (client authentication)	iDRAC client certificate was not signed by a CA trusted by the provisioning server. Either add the iDRAC CA to the trusted list or generate a new certificate on the iDRAC.
SSL connection failed other	Enable a root account through the BIOS to retrieve the iDRAC tracelog. If the problem persists, contact technical support.
SOAP failure	The provisioning server does not support the <code>getCredentials()</code> SOAP call. Verify that the provisioning server supports auto-discovery and the provisioning server information is set correctly in the DHCP vendor option, DNS SRV record, or BIOS.

Table 5-3. Auto-Discovery Messages

Message 2	Resolutions
No credentials returned	Check that the service tag is in the list of known servers on the provisioning server.
Failed to create account	Ensure that all the 16 iDRAC accounts are not already used.

Frequently Asked Questions

This section answers questions that are frequently asked by Remote Services users.

1 What is lifecycle controller?

Lifecycle Controller (LC) is an embedded systems management solution to help customers perform diagnostics, operating system (OS) Deployment, firmware Update, and Configurations.

2 What is Unified Server Configurator?

Unified Server Configurator (USC) is an essential component of Lifecycle Controller to deploy, update, and configure systems under the Unified Extensible Firmware Interface (UEFI) environment. One major advantage of UEFI is that it is OS-Agnostic.

3 What tools does the LC replace?

The Lifecycle Controller is intended to replace the use of the *Dell Systems Build and Update Utility* DVD (software, drivers, BIOS, and other updates). Lifecycle Controller also provides Remote Services, a web services based network accessible interface for managing system hardware.

4 What is Remote Services or Remote Enablement?

Remote Services is a general term that refers to the capability of enabling users to remotely connect to the target servers and perform systems management operations.

5 How to set the network configuration to use Remote Services?

Use the ping utility to verify the connection between the client and managed server. Ensure that the client and network allows HTTP and SSL protocols.

6 What are the firewall ports that need be to enabled to ensure proper communication?

Use port 443 for HTTPS communication.

7 What is Part Replacement and how does it work?

Part Replacement is a feature that allows the system to automatically update the firmware, or configuration, or both for a hardware component that is installed or replaced.

8 What is CSIOR and when to enable it?

CSIOR stands for Collect System Inventory on Reboot. It enables automatic firmware and hardware inventory refresh during system startup. The system is shipped from the factory with CSIOR disabled. Ensure that CSIOR is enabled before using any of the features like part replacement or setting attributes.

9 How do I keep the System Inventory Information up-to-date when local changes are made to any HII attribute?

Either manually press <F10> during system startup or set the CSIOR attribute to enabled, to collect the system inventory and configuration attribute information on every system startup.

Enumerate the DCIM_SystemView class to view the value under **LastUpdateTime** property that gives the time of update for a specific component.

10 How to update the managed node using USC or Remote Services?

For USC, press <F10> during startup. Select 'Platform Update' and select 'devices to update'. For more information on Remote Services, see the *Lifecycle Controller 1.4 Web Services Interface Guide*.

11 What do I do when a fatal error occurs followed by a red screen?

Perform a cold reboot of the system when the red screen appears.

12 Do I need to install an operating system (OS) to access USC or Remote Services?

OS is not required to access USC or remote service.

13 Which UEFI version is supported? 32 bit or 64 bit?

UEFI supports 64 bit.

14 Why is the NIC inventory not returning anything even though the system is using Broadcom or INTEL NICs?

The NICs that are installed on the system are not supported by Dell.

15 Can I remotely reboot the system using WS-Management functions?

Yes, the system can be rebooted using the **RequestStateChange()** method on the ComputerbSystem class. A reboot can be scheduled by creating a reboot job using the **CreateRebootJob()** method on the SoftwarebInstallationbService class and then scheduling the reboot job using the **SetupJobQueue()** method on the Job control Service.

16 How do I cancel a system service when in use?

Use the iDRAC configuration utility (CTRL+E option during startup) or remove the power cable to reset the iDRAC.

17 How do I reset the system to factory defaults?

Use the iDRAC configuration utility (CTRL+E option during boot), 'Reset to Default' -> 'yes' to continue.

18 What are the Dell licensed features that require a Dell vFlash SD card?

The part replacement feature is a licensed feature that requires the presence of the Dell vFlash SD card. All vFlash SD Card management capabilities require a Dell vFlash branded SD Card.

19 Why does the "LastUpdateTime" not change when I replace a DIMM?

If a DIMM is removed and reinstalled in the same slot then "LastUpdateTime" does not change in the view.

20 Are there ways to improve response time for getting PCIDeviceView using WinRM?

Yes. Setting the WinRM configuration by executing the following command reduces the time taken by PCIDeviceView enumeration.

```
#winrm set winrm/config @{MaxBatchItems="100"}
```

21 How to clear jobs?

- a** Enumerate **DCIM_LifecycleJobs** to list all the jobs in Lifecycle Controller.
- b** Use **DeleteJobqueue()** method to delete particular jobs.

22 What happens when the `DeleteJobQueue()` method is invoked with a `JobID` of `JID_CLEARALL` from the WSMAN client?

All jobs are cleared. Some services and processes on the iDARC are restarted and there is a delay of one to three minutes before Remote Enablement WSMAN commands are available again.

23 When do we see the changes reflected through the WS-Management if the changes are made locally in HII?

After exiting from USC, the WS-Management interface updates the available information in approximately 2 minutes.

24 What should be state of the system for the `CreateTargetedConfigJob()` method invocation to be successful?

The System must either be powered off, or past BIOS POST (for example, BIOS or UEFI boot manager), or must have booted into the OS for the `CreateTargetedConfigJob()` method to be successful.

25 What is different about the `ProcCore` setting for Quad core processors?

For quad port processors, setting the attribute `ProcCore` value to 4 sets the current value to "All".

26 Why are the `NIC Blink LED` attributes always set to `NULL` after the job is completed?

A blink LED NIC attribute is a one time setting that you are able to set, but once SSIB task is complete, it will set the current value back to null. The purpose of this attribute is to blink the NIC LEDs for a certain amount of time (seconds).

27 How many attributes can I set through the `SetAttribute()` method.

You can set only one attribute through the `SetAttribute()` method. To set two or more attributes in one method invocation, use the `SetAttributes()` method on the services for the component being configured.

28 Why do I see some other attributes being set when a different attribute is set?

There are few attributes in BIOS and NIC that have dependencies. When you set a specific attribute, all the dependent attributes are modified based on their dependency. This is an expected behavior.

BIOS Dependencies — TPM, Power Management, AC power recovery, and Embedded NIC.

NIC Dependencies — VLAN Mode and WakeONLAN attributes.

29 Can I set VLanMode and VLanID in the same Task?

You cannot set the VLanMode and VLanID attributes involving dependencies in the same task. You must set the parent attribute (VLanMode) as the first set operation, the child attribute (VLanID) as a second set operation and then commit the job.

Schema

This section displays a typical schema for lifecycle log.

Lifecycle Log Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:dm="
"http://www.w3.org/2001/XMLSchema" targetNamespace="
"http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <xs:element name="Description" type="xs:string"/>
  <xs:element name="MessageID" type="xs:string"/>
  <xs:element name="Arg" type="xs:string"/>
  <xs:element name="MessageArguments">
    <xs:complexType>
      <xs:sequence minOccurs="0">
        <xs:element ref="dm:Arg" minOccurs="0"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Event">
    <xs:complexType>
      <xs:sequence minOccurs="0">
        <xs:element ref="dm:Description" minOccurs="0"/>
        <xs:element ref="dm:MessageID" minOccurs="0"/>
        <xs:element ref="dm:MessageArguments" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="TimeStamp" type="xs:string" use="
required"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```

        <xs:attribute name="AgentID" type="xs:integer" use=
"required"/>
        <xs:attribute name="Severity" type="xs:integer" use="required"/>
        <xs:attribute name="s" type="xs:string" use="required"/>
    </xs:complexType>
</xs:element>
<xs:element name="Events">
    <xs:complexType>
        <xs:sequence minOccurs="0">
            <xs:element ref="dm:Event" minOccurs="0" maxOccurs=
"unbounded"/>
        </xs:sequence>
        <xs:attribute name="lang" type="xs:string" use="optional"/>
        <xs:attribute name="schemaVersion" type="xs:string" use=
"optional"/>
        <xs:attribute name="timeStamp" type="xs:dateTime" use=
"optional"/>
    </xs:complexType>
</xs:element>
</xs:schema>

```

Easy-to-use System Component Names

Table C-1. Easy-to-use Names of System Components

System Component Name	Easy-to-use Name
RAID.Integrated.1	Integrated RAID Controller 1 Integrated RAID Controller 2
RAID.Slot.1-1	RAID Controller in Slot 1
NIC.Mezzanine.1B-1	
NIC.Mezzanine.1C-1	NIC in Mezzanine 1 (Fabric B)
NIC.Mezzanine.1C-2	
NIC.Mezzanine.3C-2	
NonRAID.Integrated.1-1	Integrated Storage Controller 1 Integrated Storage Controller 2
NonRAID.Slot.1-1	Storage Controller in Slot 1
NonRAID.Mezzanine.2C-1	Storage Controller in Mezzanine 1 (Fabric C)
NIC.Embedded.1	Embedded NIC 1
NIC.Embedded.2	Embedded NIC 2
NIC.Embedded.1-1	Embedded NIC 1 Port 1
NIC.Embedded.1-1-1	Embedded NIC 1 Port 1 Partition 1
NIC.Slot.1-1	NIC in Slot 1 Port 1
NIC.Slot.1-2	NIC in Slot 1 Port 2
Video.Embedded.1-1	Embedded Video Controller
HostBridge.Embedded.1-1	Embedded Host Bridge 1
ISABridge.Embedded.1-1	Embedded ISA Bridge 2
P2PBridge.Embedded.1-1	Embedded P2P Bridge 3

Table C-1. Easy-to-use Names of System Components (continued)

System Component Name	Easy-to-use Name
P2PBridge.Mezzanine.2B-1	Embedded Host Bridge in Mezzanine 1 (Fabric B)
USBUHCI.Embedded.1-1	Embedded USB UHCI 1
USBOHCI.Embedded.1-1	Embedded USB OHCI 1
USBEHCI.Embedded.1-1	Embedded USB EHCI 1
Disk.SATAEmbedded.A-1	Disk on Embedded SATA Port A
Optical.SATAEmbedded.B-1	Optical Drive on Embedded SATA Port B
TBU.SATAExternal.C-1	Tape Back-up on External SATA Port C
Disk.USBFront.1-1	Disk connected to front USB 1
Floppy.USBBack.2-1	Floppy-drive connected to back USB 2
Optical.USBFront.1-1	Optical drive connected to front USB 1
Disk.USBInternal.1	Disk connected to Internal USB 1
Optical.iDRACVirtual.1-1	Virtually connected optical drive
Floppy.iDRACVirtual.1-1	Virtually connected floppy drive
Disk.iDRACVirtual.1-1	Virtually connected disk
Floppy.vFlash.<string>	vFlash SD Card Partition 2
Disk.vFlash.<string>	vFlash SD Card Partition 3
iDRAC.Embedded.1-1	iDRAC
System.Embedded.1-1	System
HardDisk.List.1-1	Hard Drive C:
BIOS.Embedded.1-1	System BIOS
BIOS.Setup.1-1	System BIOS Setup
PSU.Slot.1	Power Supply 1
Fan.Embedded.1	Fan 1 Fan 2
System.Chassis.1	Blade Chassis
LCD.Chassis.1	LCD

Table C-1. Easy-to-use Names of System Components *(continued)*

System Component Name	Easy-to-use Name
Fan.Slot. 1	Fan 1
Fan.Slot. 2	Fan 2
...	...
Fan.Slot. 9	Fan 9
MC.Chassis.1	Chassis Management Controller 1
MC.Chassis.2	Chassis Management Controller 2
KVM.Chassis.1	KVM
IOM.Slot.1	IO Module 1
...	...
IOM.Slot.6	IO Module 6
PSU.Slot.1	Power Supply 1
...	...
PSU.Slot.6	Power Supply 6
CPU.Socket.1	CPU 1
System.Modular.2	Blade 2
DIMM.Socket.A1	DIMM A1

Index

A

auto-discovery
 enable, 20

C

Certificates
 managing, 25

D

Deleting Configuration, 44
deployment interfaces, 12
DHCP/DNS
 configure, 19

J

Job Control, 52

L

Log
 export hardware inventory, 43
 export lifecycle log, 44

P

Profile
 BIOS and boot, 59
 LC management, 56
 NIC configuration, 58
 OS deployment, 55
 other Hardware, 65
 persistent storage, 61
 RAID configuration, 63

R

RAID configuration, 48
Remote Operating System
 Deployment, 27
remote operating system
 deployment, 27
 deployment interface, 27
 main features, 27
 prerequisites and
 dependencies, 31
 use case, 31
 workflow, 31
remote services, 7
Remote Update, 33
Remotely reinitiating discovery
 and handshake, 24

S

Scheduling Remote Update, 35

Staging and Booting to OS image
on vFlash, 32

T

troubleshooting, 69

Types of remote scheduling, 36

V

vFlash SD Card, 47

W

web services for management, 8

WS-MAN, 8