Cisco IMC REST API Overview

This chapter includes the following sections:

Introduction

Representational state transfer (REST) or RESTful web services allow you to provide interoperability between computer systems on the Internet. Using the REST-compliant web services you can request systems to access and manipulate textual representations of web resources using a uniform and predefined set of stateless operations. Cisco has now built capabilities of using RESTful APIs to configure the UCS C-series servers using the Redfish™ technology.

Redfish™ is an open industry standard specification and schema that specifies a RESTful interface and utilizes JSON and OData to help customers integrate solutions within their existing tool chains. It utilizes a range of scalable IT technologies that are widely used, and by using these accepted technologies, it makes the use of Redfish™ easier. Redfish™ is sponsored and controlled by the Distributed Management Task Force, Inc. (DMTF), a peer-review standards body recognized throughout the industry.

Note To determine which Cisco UCS rack-mount servers are supported by this firmware release, see the associated *Release Notes*. The release notes are available at the following URL: http://www.cisco.com/en/US/products/ps10739/prod_release_notes_list.html

For more information on DMTF and Redfish™ standards, see DMTF and Redfish™

Beginning with release 4.2(2a), you can use REST with Cisco UCS S-Series servers also.

New and Modified APIs

New and Modified APIs in Release 4.2(3d)

New APIs:

- Data Sanitization - Beginning with release 4.2(3d), Cisco IMC supports data sanitization feature. Using the data sanitization process, Cisco IMC erases all sensitive data, thus making extraction or recovery of customer data impossible. You can check the status and progress of the data sanitization process for each individual device erase from the status report and rectify any issues, if required.
  - You must perform data sanitization on the components that contain customer data.
  - This feature is supported on the following servers:
    - Cisco UCS C220 M5, C240 M5, C480 M5, C125 M5 servers
    - Cisco UCS C220 M6, C240 M6, C225 M6, C245 M6 servers

New and Modified APIs in Release 4.2(3b)

New APIs:

- Password Change
- Configuring LDAP Server with NULL Address
- DDNS and Domain Name Properties Support Under NIC
- Enabling SMTP Service with Allowable Port from 1 to 65535
- Setting Session Timeout for SSH Protocol
- Setting COM Port for Serial Over LAN Policy

Note This API is available in both C-series and S-series servers.

- Setting Privilege and Encryption Key

Note This API is available in both C-series and S-series servers.

New and Modified APIs in Release 4.2(2a)

- Beginning with release 4.2(2a), you can use REST with Cisco UCS S-Series servers. For Cisco UCS S-Series server examples, see Cisco IMC REST API Examples for Supported S-Series Servers in Release 4.2.

- Configuring TACACS+ and Priority for various authentication methods—As per Redfish schema, the value for **Priority** starts from **0** in Redfish API, whereas for other Cisco IMC interfaces, the priority starts from **1**.
- Following are deprecated in release 4.2(2a):
  - **EncryptionStatus** under Oem/Cisco in `/redfish/v1/Managers/CIMC` URI
  - **VideoEncryption** under Oem/Cisco in `/redfish/v1/Managers/CIMC/NetworkProtocol` URI
  - **SyslogConnectionInfo** under Oem/Cisco in `/redfish/v1/Managers/CIMC/LogServices/CIMC` URI
- New APIs:
  - Cisco IMC Syslog Configuration
  - FlexMMC Configurations

New and Modified APIs in Release 4.2(1a)

New APIs:

- Configuring SNMP Users
- Configuring MCTP Fault Alert Setting
- Adding SPDM Authority Certificate
- Viewing Endpoint SPDM Certificate

Redfish™ Architecture

The Redfish™ API comprises a folder structure that starts with the Redfish root at "/redfish/". In case of a C-Series server, the root is accessed through the URI https://<Cisco IMC IP>/redfish/v1/ - the "v1" at the end of the URI denotes the version of the API.

The URI is the primary unique identifier of resources. Redfish™ URIs consist of three parts as described in RFC3986: Part one defines the scheme and authority of the URI, part two specifies the root service and version, and part three defines a unique resource identifier.

For example, in the following URI: https://mgmt.vendor.com/redfish/v1/Systems/SvrID:

- https://mgmt.vendor.com is the scheme and authority
- /redfish/v1 is the root and version
- /Systems/SvrID is the resource identifier

Redfish™ Tree Structure

The Redfish tree structure comprises a top-level root from where the RESTful interface branches out to cover a number of "Collections" that subsequently include multiple levels within, creating a tree-like structure. You can navigate down to this structure to find information and settings.

For example, accessing the Redfish™ structure for the controller on a C-Series server would be navigated by using the following path: **https://10.10.10.10/redfish/v1/Systems/FCH2005V1EN/SimpleStorage/SLOT-HBA**

Note Some portions of an API path could vary depending on the hardware configuration. For example, "SLOT-HBA" may be different when another type of RAID controller is installed in the managed server.

Redfish™ Operations

Redfish™ uses the HTTPS method to perform operations of a RESTful API. You can specify the type of request being made. It adheres to a standard CRUD (Create, Retrieve, Update, and Delete) format. Depending on the desired result, you can issue the following types of commands:

- **GET**: View data
- **POST**: Create resources or use actions
- **PATCH**: Change one or more properties on a resource
- **DELETE**: Remove a resource

Note Currently, HEAD and PUT operations are not supported for Redfish™ URIs.

Table 1: Redfish Schema and Specification

Release	Redfish Schema	Redfish Specification
Release 4.2(1a)	https://www.dmtf.org/sites/default/files/standards/documents/DSP8010_2020.3.zip	https://www.dmtf.org/sites/default/files/standards/documents/DSP0266_1.7.0.pdf
Release 4.2(2a)	https://www.dmtf.org/sites/default/files/standards/documents/DSP8010_2021.1.zip	https://www.dmtf.org/sites/default/files/standards/documents/DSP0266_1.13.0.pdf
Release 4.2(3b)	https://www.dmtf.org/sites/default/files/standards/documents/DSP8010_2021.1.zip	https://www.dmtf.org/sites/default/files/standards/documents/DSP0266_1.13.0.pdf

Management Standard

IT solution models have evolved over the years and given way to several Out-of-Band (OOB) systems management standards, or lights-out management (LOM) systems that work within emerging programming standards and can be implemented in the embedded systems. While this has worked fairly well, there was still a need for a single management standard that could handle the various demands of IT solutions robustly. Expanded scale, higher security, and multi-vendor openness call for equally diverse DevOps tools and processes.

Keeping these requirements in mind, the DMTF took on the responsibility of creating a new management interface standard, which resulted in Redfish™ version 1.0, which was formally launched in July, 2015.

Key features of the Redfish™ management standard include:

- Simple to use and highly secure
- Encrypted connections and generally heightened security
- Simple programmatic interface that can be easily managed using scripts
- Meets Open Compute Project's Remote Machine Management requirements
- Based on widely-used standards for web APIs and data formats

Redfish™ can support an entire range of server architectures, right from monolithic servers to converged infrastructure and hyper-scale architecture. The Redfish™ data model is vendor neutral, and defines its own structure and format of data that comprises server status, inventory and existing operational functions. You, as an administrator can then automate management scripts to manage any Redfish™ compliant server, resulting in the efficient operation of a heterogeneous server fleet.

In terms of security, Redfish™ offers a highly secure and reliable communication opportunity with its use of HTTPS encryption as opposed to conventional management protocols. You can convey all Redfish™ network traffic, including event notifications across the network in an encrypted packet, reducing threats significantly.

Key Technologies

HTTPS Communications

The Hypertext Transfer Protocol or HTTP is an application protocol for distributed, collaborative, hypermedia information systems and forms the foundation of data communication for the World Wide Web. Secure HTTP or HTTPS is a secure version of HTTP that enables secure communications by operating HTTP within a network connection encrypted by TLS or SSL. By utilizing HTTPS, Redfish™ significantly enhances the security of server management especially in comparison to legacy server management protocols.

RESTful Application Programming Interface

Representational State Transfer (REST) or RESTful API is a programming interface that uses the HTTP request to retrieve information with the help of GET, POST, and DELETE data. Many IT companies use the RESTful architecture. Leveraging this standardized approach, Redfish™ implements a RESTful API for accessing management information and for issuing commands to change the configuration or operational state of a server.

Operational Model

Redfish™ operations are initiated by a client using HTTPS for GET, POST, PATCH and DELETE operations and are capable of interpreting JSON responses from the managed server. The responses provide the requested information and indications of success or failure of the requested operation.

Redfish™ Client

RESTful API goes by the principle "Everything is a Resource". This means that every Uniform Resource Identifier or URI represents a resource of a specific type - a service, a collection or an individual entity. Within the Redfish™ context however, a resource can be thought of as the content of the HTTPS message returned

when accessing a URI. A variety of REST Clients can be used for gaining access to Redfish™ resources such as:

- Applications such as the “Advanced REST Client” and “Postman” from the Google Chrome web store.
- “REST Easy” and “RESTClient” plug-ins for the Firefox browser.
- cURL, Python, and other scripting or programming languages that provide support for dealing with URIs and for parsing JSON payloads.