

Dell EMC OpenManage Python Software Development Kit

Version 1.1

API Reference Guide

Notes, cautions, and warnings

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

 **CAUTION:** A CAUTION indicates either potential damage to hardware or loss of data and tells you how to avoid the problem.

 **WARNING:** A WARNING indicates a potential for property damage, personal injury, or death.

© Copyright2018 Dell Inc. or its subsidiaries. All rights reserved. Dell, EMC, and other trademarks are trademarks of Dell Inc. or its subsidiaries. Other trademarks may be trademarks of their respective owners. Licensed under the Apache license version 2.0; you may not use this file except in compliance with the license. You may obtain a copy of the license at <https://www.apache.org/licenses/LICENSE-2.0>. Unless applicable law required or agreed to in writing, software distributed under the license is distributed on an AS IS BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the license for the specific language governing permissions and limitations under the license.

Contents

1 Introduction to OpenManage python software development toolkit.....	5
SDK concepts and architectural components.....	5
2 Installation and Uninstallation of OMPSDK.....	6
Installing OMPSDK.....	6
Uninstalling OMPSDK.....	6
3 OpenManage Python SDK API introduction.....	7
OpenManage Python SDK API list.....	7
Monitoring API list.....	7
Administration tasks API list.....	17
Export and Import Tasks API list.....	19
Server iDRAC Settings Configuration API list.....	27
Boot Settings Configuration API list.....	47
RAID Configuration API list.....	52
iDRAC LC Jobs API list.....	56
iDRAC LC Status Check API list.....	57
4 Getting Started.....	59
OMPSDK Infrastructure.....	59
Initialization of OMPSDK Infrastructure.....	59
API to setup share.....	61
Setting up a remote network share.....	61
iDRAC server information.....	61
Administration Tasks.....	61
Server configuration profile overview.....	62
Import export server configuration profiles.....	62
Export server configuration profile.....	62
Import server configuration profile.....	62
Exporting LC Logs.....	63
Server iDRAC settings and configuration.....	63
iDRAC User Configuration settings.....	63
iDRAC Network Configuration.....	63
iDRAC Service Configuration.....	64
iDRAC System Settings.....	64
BIOS Boot settings.....	65
BIOS Boot settings.....	65
RAID Configuration.....	65
Create Virtual Disk.....	65
Delete Virtual Disk.....	65
iDRAC LC jobs.....	65
Job API's.....	65

iDRAC LC status check.....	66
----------------------------	----

Introduction to OpenManage python software development toolkit

DellEMC OpenManage python software development kit (OMPSDK) is a library for programatically managing DellEMC Servers and Chassis. Using this library a user can perform lifecycle operations on DellEMC PowerEdge servers, chassis storage and switches. The OMPSDK leverages the DMTF Redfish, DMTF's Web Services Management standard (WS-Man), and IETF's Simple Network Management Protocol (SNMP) standard for managing and monitoring the DellEMC PowerEdge servers and chassis.

SDK concepts and architectural components

An OMPSDK is a complete set of APIs that allow you to perform most of actions you would need for creating, monitoring and managing the servers or chassis applications. An SDK provides a set of APIs, relevant documentation, code samples, processes, that allows developers to create software applications on a specific platform. Components in OMPSDK are:

- OMPSDK Infrastructure
- Driver
- Credential management
- Protocol Management
 - Protocol preference and options
- Log Manager

Installation and Uninstallation of OMPSDK

Installing OMPSDK

To install OMPSDK, there are some prerequisites which are essential.

Prerequisites:

Ensure that you have the following:

- Python v2.7 or v3.4 and above.
- Package management system (pip).

Run the following command

```
pip install omsdk
```

OMPSDK is installed and the package is available in PyPI (Python package Index).

Uninstalling OMPSDK

To uninstall OMPSDK, use the following command:

```
pip uninstall omsdk
```

OpenManage Python SDK API introduction

OpenManage Python SDK (OMPSDK) API Module allows data center and IT administrators to automate and orchestrate the provisioning, configuration, deployment, and update of Dell EMC PowerEdge Servers (12th generation of PowerEdge servers and later) by using the management automation capabilities in-built into the integrated Dell remote access controller (iDRAC). This product evolves conservatively and values simplicity in design and setup.

The traditional approach for creating a plug-in always starts from scratch, such as building communication module, business logic module, user interface module, and native console data translation module. In most of the plug-ins the native console data translation module varies, but all other modules remain more or less same. As a result, every plug-in maintains their own set of libraries which adds no uniformity in implementation across plug-ins. Indirectly it increases the overhead sustaining cost of the plug-ins.

OMPSDK is targeted to provide

- A uniform implementation of communication module and core business logic module
- Abstraction of multiple communication protocols with data messaging
- Macro services as per use case—Monitoring, configuration, and more
- Optionally automatic code generating capability
- Quick integration to other DevOps.

OpenManage Python SDK API list

This section describes the resource URLs and related operations that are available in the OpenManage Python SDK(OMPSDK) implementation.

Monitoring API list

`sdkinfra.find_driver`

API string/Method: `sdkinfra.find_driver`.

Description: The `sdkinfra.find_driver()` method is used to initialize and load the device drivers.

Table 1. Parameters for `sdkinfra.find_driver`

Parameter Name	Type	Allowed Values
<code>ipaddr</code>	String	NA
Description	The IP address or hostname of the device.	

Table 2. Parameters for the Credentials

Parameter Name	Type	Allowed Values
creds	Dictionary of credentials	Snmpv2Credentials UserCredentials
Description	A bundle of credentials required to communicate to the device driver. Snmpv2Credentials UserCredentials	

Table 3. Parameter for Protocol preference

Parameter Name	Type	Allowed Values
protopref	Enumeration of preferred protocol	ProtoPreference(ProtocolEnum.SNMP) ProtoPreference(ProtocolEnum.WSMAN) ProtoPreference(ProtocolEnum.REDFISH)
Description	The preferred protocol used if the device supports the protocol.	

Table 4. Parameter for protocol specific options

Parameter Name	Type	Allowed Values
pOptions	Dictionary of protocol-specific options	SNMPOptions WsManOptions RedfishOptions
Description	A protocol-specific option for passed, port, timeout and so on. SNMPOptions WsManOptions RedfishOptions	

Return type:

Object**Device driver**: A driver handle to configure or monitor the drivers.

Example:

```
idrac=sd.find_driver(ipaddr,
creds=UserCredentials("root", "calvin"),
protopref=ProtoPreference(ProtocolEnum.WSMAN),
pOptions=WsManOptions(port=443))
```

sdkinfra.get_driver

API string/Method: sdkinfra.get_driver.

Description: The sdkinfra.get_driver() method is used to get a device driver for the given IP address or hostname, and check for a particular device type.

Table 5. Parameters for sdkinfra.get_driver

Parameter Name	Type	Allowed Values
ipaddr	String	NA
Description	The IP address or hostname of the device.	

Table 6. Parameters for device type

Parameter Name	Type	Allowed Values
driver_en	Enumeration of the device type	sdkinfra.driver_enum.iDRAC sdkinfra.driver_enum.CMC sdkinfra.driver_enum.Compellent sdkinfra.driver_enum.EqualLogic sdkinfra.driver_enum.MDArray sdkinfra.driver_enum.F10 sdkinfra.driver_enum.NSeries
Description	The enumeration of the device type.	

Table 7. Parameters for credentials

Parameter Name	Type	Allowed Values
creds	Dictionary of credentials	Snmpv2Credentials UserCredentials
Description	A bundle of credentials to find the device driver. Snmpv2Credentials UserCredentials	

Table 8. Parameters for preferred protocol

Parameter Name	Type	Allowed Values
protopref	Enumeration of preferred protocol	ProtoPreference(ProtocolEnum.SNMP) ProtoPreference(ProtocolEnum.WSMAN) ProtoPreference(ProtocolEnum.REDFISH)
Description	The preferred protocol used if the device supports the protocol.	

Table 9. Parameters for protocol specific options

Parameter Name	Type	Allowed Values
pOptions	Dictionary of protocol-specific options	SNMPOptions WsManOptions RedfishOptions
Description	A protocol-specific option for passed, port, timeout and so on.	

Parameter Name	Type	Allowed Values
	SNMPOptions WsManOptions RedfishOptions	

Example:

```
idrac=sd.get_driver(sd.driver_enum.iDRAC,
ipaddr,
creds=UserCredentials("root", "calvin"),
protopref=ProtoPreference(ProtocolEnum.REDFISH),
pOptions=RedfishOptions(port=443))
```

Return type:

Object **Device driver**: A driver handle to configure or monitor the drivers.

sdkbase.iBaseDriver.get_entityjson

API string/Method: sdkbase.iBaseDriver.get_entityjson.

Description: The sdkbase.iBaseDriver.get_entityjson() is used to create the **JSON** of the device by fetching the attributes from the device using the protocol. It internally creates the raw **JSON** of the device.

Example:

```
status=idrac.get_entityjson()
st = idrac.get_json_device()
print(status)
```

Returns: Success or Failure.

Return Type: JSON.

sdkbase.iBaseDriver.get_partial_entityjson

API string/Method: sdkbase.iBaseDriver.get_partial_entityjson.

Description: The sdkbase.iBaseDriver.get_partial_entityjson() is used to get the **JSON** with the components which are passed as an argument for the function.

Table 10. Parameters for sdkbase.iBaseDriver.get_partial_entityjson

Parameter Name	Type	Allowed Values	
en	idrac.get_partial_entityjson (idrac.ComponentEnum.CPU)	iDRAC	CMC
		iDRACCompEnum.BIOS iDRACCompEnum.ControllerBattery iDRACCompEnum.Controller iDRACCompEnum.CPU iDRACCompEnum.Enclosure iDRACCompEnum.EnclosureEMM iDRACCompEnum.EnclosurePSU	CMCCompEnum.BladeSlot CMCCompEnum.CMC CMCCompEnum.ComputeModule CMCCompEnum.Controller CMCCompEnum.ControllerBattery CMCCompEnum.Enclosure CMCCompEnum.EnclosureEMM CMCCompEnum.EnclosurePSU

Parameter Name	Type	Allowed Values	
		iDRACCompEnum.EnclosureFanSensor iDRACCompEnum.EnclosureTempSensor iDRACCompEnum.Fan iDRACCompEnum.FC iDRACCompEnum.HostNIC iDRACCompEnum.iDRAC iDRACCompEnum.iDRACNIC iDRACCompEnum.License iDRACCompEnum.LogicalSystem iDRACCompEnum.NIC iDRACCompEnum.PCIDevice iDRACCompEnum.PowerSupply iDRACCompEnum.PresenceAndStatusSensor iDRACCompEnum.PhysicalDisk iDRACCompEnum.Sensors_Amperage iDRACCompEnum.Sensors_Battery iDRACCompEnum.Sensors_Fan iDRACCompEnum.Sensors_Intrusion iDRACCompEnum.Sensors_Temperature iDRACCompEnum.Sensors_Voltage iDRACCompEnum.SystemMetrics iDRACCompEnum.VFlash iDRACCompEnum.Video iDRACCompEnum.VirtualDisk	CMCCCompEnum.Fan CMCCCompEnum.IOModule CMCCCompEnum.KVM CMCCCompEnum.License CMCCCompEnum.PowerSupply CMCCCompEnum.PhysicalDisk CMCCCompEnum.PCIDevice CMCCCompEnum.StorageModule CMCCCompEnum.Slots_Summary CMCCCompEnum.System CMCCCompEnum.VirtualDisk
Description	List of strings or enumeration of the components.		

Example:

```
status=idrac.get_partial_entityjson(idrac.ComponentEnum.EnclosureTempSensor)
print(status)
```

Returns: Success or Failure.

Return Type: JSON.

sdkbase.iBaseDriver.ContainmentTree

API string/Method: sdkbase.iBaseDriver.ContainmentTree.

Description: The sdkbase.iBaseDriver.ContainmentTree() is used to create and return the component tree of the device with keys of the Components organized in a tree structure.

Example:

```
system = idrac.ContainmentTree
print(system)
```

Returns: Success or Failure.

Return Type: JSON.

sdkdevice.iDeviceDriver.get_json_device

API string/Method: sdkdevice.iDeviceDriver.get_json_device.

Description: The `sdkdevice.iDeviceDriver.get_json_device()` is used to apply the monitor filter on the raw `entityJSON` and also filter the category of the attributes and components required for the client.

Table 11. Parameters for sdkdevice.iDeviceDriver.get_json_device

Parameter Name	Type	Allowed Values
monitorfilter	MonitorScope	MonitorScope.BasicInventory MonitorScope.ConfigState MonitorScope.Health MonitorScope.Inventory MonitorScope.Key MonitorScope.MainHealth MonitorScope.Metrics MonitorScope.OtherHealth MonitorScope.OtherInventory
Description	The category of filters to apply.	

Table 12. Parameter for Component

Parameter Name	Type	Allowed Values	
compScope	String or Enum	iDRAC	CMC
		iDRACCompEnum.BIOS iDRACCompEnum.ControllerBattery iDRACCompEnum.Controller iDRACCompEnum.CPU iDRACCompEnum.Enclosure iDRACCompEnum.EnclosureEMM iDRACCompEnum.EnclosurePSU iDRACCompEnum.EnclosureFanSensor iDRACCompEnum.EnclosureTempSensor iDRACCompEnum.Fan iDRACCompEnum.FC iDRACCompEnum.HostNIC iDRACCompEnum.iDRAC iDRACCompEnum.iDRACNIC iDRACCompEnum.License iDRACCompEnum.LogicalSystem iDRACCompEnum.NIC iDRACCompEnum.PCIDevice iDRACCompEnum.PowerSupply iDRACCompEnum.PresenceAndStatusSensor	CMCCompEnum.BladeSlot CMCCompEnum.CMC CMCCompEnum.ComputeModule CMCCompEnum.Controller CMCCompEnum.ControllerBattery CMCCompEnum.Enclosure CMCCompEnum.EnclosureEMM CMCCompEnum.EnclosurePSU CMCCompEnum.Fan CMCCompEnum.IOModule CMCCompEnum.KVM CMCCompEnum.License CMCCompEnum.PowerSupply CMCCompEnum.PhysicalDisk CMCCompEnum.PCIDevice CMCCompEnum.StorageModule CMCCompEnum.Slots_Summary CMCCompEnum.System CMCCompEnum.VirtualDisk

Parameter Name	Type	Allowed Values
		iDRACCompEnum.PhysicalDisk iDRACCompEnum.Sensors_Amperage iDRACCompEnum.Sensors_Battery iDRACCompEnum.Sensors_Fan iDRACCompEnum.Sensors_Intrusion iDRACCompEnum.Sensors_Temperature iDRACCompEnum.Sensors_Voltage iDRACCompEnum.SystemMetrics iDRACCompEnum.VFlash iDRACCompEnum.Video iDRACCompEnum.VirtualDisk
Description	The required components.	

Return type:

Table 13. Parameter for return type

Returns	Return type
JSON	The formatted JSON of the device with all the applied process attributes—mapping, units conversion.

 **NOTE:** Before utilizing the APIs, ensure to apply the `sdkinfra.find_driver` or `sdkinfra.get_driver`.

sdkcreds.Snmpv2Credentials

API string/Method: `sdkcreds.Snmpv2Credentials`.

Description: Credentials for SNMP version 1 and version 2.

Table 14. Parameters for sdkcreds.Snmpv2Credentials

Parameter Name	Type
community	String
Description	Community string for the SNMP device.

Table 15. Parameter for write community

Parameter Name	Type	Allowed Values
writeCommunity	String	NA
Description	Write community string for the SNMP device.	

Example:

```
idrac=sd.find_driver(ipaddr,
creds=Snmpv2Credentials("nagios"),
protopref=ProtoPreference(ProtocolEnum.SNMP),
pOptions=SNMPOptions(port=163))
```

sdkcreds.UserCredentials

API string/Method: `sdkcreds.UserCredentials` .

Description: Credentials username and password for WSMAN communication with the device.

Table 16. Parameters for `sdkcreds.UserCredentials`

Parameter Name	Type
username	String
Description	User name for WSMAN communication.

Table 17. Parameter for password

Parameter Name	Type	Allowed Values
password	String	NA
Description	Password for WSMAN communication.	

Example:

```
idrac=sd.find_driver(ipaddr,creds=UserCredentials("root", "calvin"))
```

sdkwsmanbase.WsManOptions

API string/Method: `sdkwsmanbase.WsManOptions` .

Description: Options to establish WSMAN communication.

Table 18. Parameters for `sdkwsmanbase.WsManOptions`

Parameter Name	Type	Allowed Values
authentication	AuthenticationType	NA
Description	HTTP authentication type Basic , Digest .	

Table 19. Parameter for port

Parameter Name	Type	Allowed Values
port	Integer	NA
Description	HTTPS port number for WSMAN communication.	

Table 20. Parameter for connection time-out

Parameter Name	Type	Allowed Values
connection_timeout	Integer	NA
Description	Time in seconds to wait for the server to connect before giving up.	

Table 21. Parameter for read time-out

Parameter Name	Type	Allowed Values
read_timeout	Integer	NA
Description	Time in seconds to wait for the server to read data before giving up.	

Table 22. Parameter for max retries

Parameter Name	Type	Allowed Values
max_retries	Integer	NA
Description	HTTP connection retries in case of failures.	

Table 23. Parameter for verifying SSL certificate

Parameter Name	Type	Allowed Values
verify_ssl	Boolean	NA
Description	SSL certificate verification.	

sdcredfishbase.RedfishOptions

API string/Method: `sdcredfishbase.RedfishOptions`.

Description: Options to establish REDFISH communication.

Table 24. Parameters for sdcredfishbase.RedfishOptions

Parameter Name	Type
authentication	AuthenticationType
Description	HTTP authentication type Basic , Digest .

Table 25. Parameter for port

Parameter Name	Type	Allowed Values
port	Integer	NA
Description	HTTPS port number for WSMAN communication.	

Table 26. Parameter for connection time-out

Parameter Name	Type	Allowed Values
connection_timeout	Integer	NA
Description	Time in seconds to wait for the server to connect before giving up.	

Table 27. Parameter for read time-out

Parameter Name	Type	Allowed Values
read_timeout	Integer	NA
Description	Time in seconds to wait for the server to read data before giving up.	

Table 28. Parameter for max retries

Parameter Name	Type	Allowed Values
max_retries	Integer	NA
Description	HTTP connection retries in case of failures.	

Table 29. Parameter for verify ssl

Parameter Name	Type	Allowed Values
verify_ssl	Boolean	NA
Description	SSL certificate verification.	

MonitorScopefilter.setdefaultMap

API string/Method: MonitorScopefilter.setdefaultMap .

Description: MonitorScopeFilter.setdefaultMap(MonitorScope.MainHealth, "Unknown") is used to change the default value of the monitor scope attributes that gives out null values or if they are not available in the particular device.

NOTE: To execute this API, load the driver using find_driver or get_driver API . Also get_entityjson API is mandatory for this execution.

Table 30. Parameters for MonitorScopefilter.setdefaultMap

Parameter Name	Type	Allowed Values
mScope	Enum MonitorScope	MonitorScope.BasicInventory MonitorScope.ConfigState MonitorScope.Health MonitorScope.Inventory MonitorScope.MainHealth MonitorScope.Metrics

Parameter Name	Type	Allowed Values
		MonitorScope.OtherHealth MonitorScope.OtherInventory
Description	Defines the category of the filter. The default value of MonitorScope attributes is set to Not Available except for MonitorScope.Health and MonitorScope.Metrics .	

Example:

```
device.get_entityjson()
    monitorfilter = MonitorScopeFilter(MonitorScope.Metrics)
    monitorfilter.setdefaultMap(MonitorScope.Metrics, "NOT AVAL")
    idrac.get_json_device(monitorfilter)
```

Table 31. Parameter for ndefval

Parameter Name	Type	Allowed Values
ndefval	String	User defined string value.
Description	To set the default value for the MonitorScope described above.	

Administration tasks API list

This section lists the available Administration tasks APIs.

idrac.config_mgr.power_boot

API string/Method: idrac.config_mgr.power_boot.

Protocol Support: WSMAN.

Description: The idrac.config_mgr.power_boot allows you to power On/Off the Server.

Table 32. Parameters for idrac.config_mgr.power_boot(PowerBootEnum.Enabled)

Parameter Name	Type	Allowed Values
power_state	Enum PowerBootEnum	PowerBootEnum.Enabled PowerBootEnum.Disabled PowerBootEnum.Reset
Description	Allows you to power On/Off the Server. Reset Enabled Disabled	

Example:

```
status= idrac.config_mgr.power_boot(power_state=PowerBootEnum.Reset)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.reset_idrac

API string/Method: idrac.config_mgr.reset_idrac.

Protocol Support: Redfish.

Description: This method is used to reset the iDRAC.

Parameters:

Table 33. Parameters for idrac.config_mgr.reset_idrac

Parameter Name	Type	Allowed Values
force	ResetForceEnum	ResetForceEnum.Graceful ResetForceEnum.Force
Description	Description: This method is used to reset the iDRAC. Graceful - Reset is performed after the device is properly restarted. Force - Reset is performed without a proper shutdown of the device.	

Example:

```
status=idrac.config_mgr.reset_idrac(force=ResetForceEnum.Force)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.reset_to_factory

API string/Method: idrac.config_mgr.reset_to_factory.

Protocol Support: WSMAN, Redfish.

Description: This method is used to reset the iDRAC to factory default.

Parameters:

Table 34. Parameters for idrac.config_mgr.reset_to_factory

Parameter Name	Type	Allowed Values
Force	ResetForceEnum	ResetForceEnum.Graceful ResetForceEnum.Force
Description	Description: This method is used to reset the iDRAC. ResetForceEnum.Graceful - Reset is performed after the device is properly restarted. ResetForceEnum.Force - Reset is performed without a proper shutdown of the device.	

Table 35. Parameters for idrac.config_mgr.reset_to_factory

Parameter Name	Type	Allowed Values
preserver_config	ResetToFactoryPreserveEnum	ResetToFactoryPreserveEnum.ResetExceptNICAndUsers ResetToFactoryPreserveEnum.ResetAll ResetToFactoryPreserveEnum.ResetAllExceptDefaultUser
Description	Allows you to preserve the configuration 0 - ResetToFactoryPreserveEnum.ResetExceptNICAndUsers Allows you to reset the device without resetting the NIC and User settings. 1 - ResetToFactoryPreserveEnum.ResetAll Allows you to reset the device along with the NIC and the User settings. 2 - ResetToFactoryPreserveEnum.ResetAllExceptDefaultUser Allows you to reset without resetting the default user.	

Export and Import Tasks API list

This section lists the available Export and Import Tasks APIs.

idrac.config_mgr.scp_export

API string/Method: idrac.config_mgr.scp_export.

Protocol Support: WSMAN, Redfish.

Description: This method is used to export the system configuration profile from the Lifecycle Controller to a local or a remote share location.

Parameters:

Table 36. Parameters for idrac.config_mgr.scp_export

Parameter Name	Type	Allowed Values
export_format	Enum	ExportFormatEnum.XML ExportFormatEnum.JSON
Description	Allows you to export the configuration detail as XML or JSON file. 0 - ExportFormatEnum.XML 1 - ExportFormatEnum.JSON Default - XML	

Table 37. Parameters for idrac.config_mgr.scp_export

Parameter Name	Type	Allowed Values
export_use	Enum	ExportUseEnum.Default ExportUseEnum.Clone ExportUseEnum.Replace
Description	The output file formats for export_use are: ExportUseEnum.Default - By default, none of the options are selected. ExportUseEnum.Clone - This option is used with Redfish protocol to generate an SCP that is ready for cloning. ExportUseEnum.Replace - This option is used to retire a server from the data center and replace it with another.	

Table 38. Parameters for idrac.config_mgr.scp_export

Parameter Name	Type	Allowed Values
include_in_export	Enum	IncludeInExportEnum.Default IncludeInExportEnum.Include_Read_Only IncludeInExportEnum.Include_Password_Hash_Values IncludeInExportEnum.Include_Both
Description	This API allows you to include more while exporting the file. The additional information added with include_in_export are: 1 - IncludeInExportEnum.Include_Read_Only - Includes read only values. 2 - IncludeInExportEnum.Include_Password_Hash_Values - Includes password hash values. 3 - IncludeInExportEnum.Include_Both - Includes both the values. IncludeInExportEnum.Default - 0	

Table 39. Parameters for idrac.config_mgr.scp_export

Parameter Name	Type	Allowed Values
job_wait	Boolean	True/False
Description	True - Wait for the performed export Server Configuration Profile jobs to complete. False - Returns only the JobID.	

Table 40. Parameters for idrac.config_mgr.scp_export

Parameter Name	Type
share_path	FileOnShare (for CIFS & NFS Share) LocalFile (For Local Share)
Description	SCP exports can be directed to local file systems and network shares. NFS - Network File System

Parameter Name	Type
	IPAddress - Name of the NFS share server. Sharename - Name of the shared file. Mountpoint - To mount the shared file. UserCredentials - Username and password for accessing the shared file. CIFS - Common Internet File System IPAddress - Name of the CIFS share server. Sharename - Name of the shared file. Sharepath - The share path where file needs to be exported. UserCredentials - Username and password for accessing the shared file.

Table 41. Parameters for idrac.config_mgr.scp_export

Parameter Name	Type	Allowed Values
target	Enum	SCPTargetEnum.ALL SCPTargetEnum.IDRAC SCPTargetEnum.BIOS SCPTargetEnum.NIC SCPTargetEnum.RAID
Description	To identify the component for export. It identifies one or more FQDDs. Selective list of FQDDs should be given in comma-separated format. SCPTargetEnum.IDRAC - The module exports only the iDRAC component in SCP file. SCPTargetEnum.BIOS - The module exports BIOS configuration in SCP file. SCPTargetEnum.NIC - The module exports NIC configuration in SCP file. SCPTargetEnum.RAID - The module exports RAID configuration in SCP file. This module will import the configuration component from the exported SCP Default - SCPTargetEnum.ALL - This module exports the complete system configuration from the Lifecycle Controller to a file on local or remote share location.	

Table 42. Parameters for idrac.config_mgr.scp_export

Parameter Name	Type
Workgroup	String
Description	The applicable workgroup.

Example:

NOTE: Use the Export LC Logs as required along with the [FileOnShare](#) API.

NOTE: Localpath is supported on 14G and later generations.

```
# scp_export - Local Share
localpath = LocalFile(local='/folder/filename.xml')
status=idrac.config_mgr.scp_import(end_host_power_state=EndHostPowerStateEnum.On,
job_wait=True,
share_path=localpath,
target = SCPTargetEnum.ALL)

# scp_export - NW Share
```

```

myshare= FileOnShare(remote = '<public_IP>:/Share/file.xml', mount_point = '/mnt/Share',
isFolder = True,
creds==UserCredentials('username', 'password'))
idrac.config_mgr.set_liason_share(myshare)
status=idrac.config_mgr.scp_import(end_host_power_state=EndHostPowerStateEnum.On,
job_wait=True,
share_path=localpath,
target = SCPTargetEnum.ALL)

```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.scp_import

Prerequisites:

API string/Method: idrac.config_mgr.scp_import

Protocol Support: WSMAN, Redfish.

Description: This method is used to import the system configuration.

Parameters:

Table 43. Parameters for idrac.config_mgr.scp_import

Parameter Name	Type	Allowed Values
end_host_power_state	Enum	EndHostPowerStateEnum.Off EndHostPowerStateEnum.On
Description	The desired host power state after the import operation is complete. 0 - Off 1 - On Default value is 0	

Table 44. Parameters for idrac.config_mgr.scp_import

Parameter Name	Type	Allowed Values
job_wait	Boolean	True/False
Description	This API raises a flag to wait for the job to complete. A false value will return the Job ID.	

Table 45. Parameters for idrac.config_mgr.scp_import

Parameter Name	Type
share_path	FileOnShare (for CIFS & NFS Share) LocalFile (For Local Share)
Description	SCP can be import from local file system or network shares. NFS - Network File System

Parameter Name	Type
	IPAddress - Name of the NFS share server. Sharename - The NFS share name. Mountpoint - To mount the shared file. UserCredentials - Username and password for accessing the shared file. CIFS - Common Internet File System IPAddress - Name of the CIFS share server. Sharename - The CIFS share name. Sharepath - The share path where file needs to be imported. UserCredentials - Username and password for accessing the shared file.

Table 46. Parameters for idrac.config_mgr.scp_import

Parameter Name	Type	Allowed Values
target	Enum	SCPTargetEnum.ALL SCPTargetEnum.IDRAC SCPTargetEnum.BIOS SCPTargetEnum.NIC SCPTargetEnum.RAID
Description	To identify the component for Import. It identifies one or more FQDDs. Selective list of FQDDs should be given in comma-separated format. SCPTargetEnum.IDRAC - The module imports iDRAC configuration in SCP file. SCPTargetEnum.BIOS - The module imports BIOS configuration in SCP file. SCPTargetEnum.NIC - The module imports NIC configuration in SCP file. SCPTargetEnum.RAID - The module imports RAID configuration in SCP file. This module will import the configuration component from the imported SCP Default - SCPTargetEnum.ALL - This module imports the complete system configuration from the Lifecycle Controller to a file on local or remote share location.	

Example:

NOTE: Use the Export LC Logs as required along with the [FileOnShare API](#).

NOTE: Localpath is supported on 14G and later generations.

```
# scp_export - Local Share
localpath = LocalFile(local='/folder/file.xml')
status=idrac.config_mgr.scp_import(end_host_power_state=EndHostPowerStateEnum.On,
job_wait=True,
share_path=localpath,
target = SCPTargetEnum.ALL)

# scp_export - NW Share
myshare= FileOnShare(remote = '<public_IP>:/Share/file.xml', mount_point = '/mnt/Share',
isFolder = True,
creds==UserCredentials('username', 'password'))
idrac.config_mgr.set_liason_share(myshare)
status=idrac.config_mgr.scp_import(end_host_power_state=EndHostPowerStateEnum.On,
job_wait=True,
```

```
share_path=localpath,
target = SCPTargetEnum.ALL)
```

```
status=idrac.config_mgr.reset_idrac(force=ResetForceEnum.Force)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.log_mgr.lclog_export

API string/Method: idrac.log_mgr.lclog_export

Protocol Support: WSMAN.

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

Parameters:

Table 47. Parameters for idrac.log_mgr.lclog_export

Parameter Name	Type	Allowed Values
job_wait	Boolean	True/False
Description	This API raises a flag to wait for the job to complete. A false value will return the Job ID.	

Table 48. Parameters for idrac.log_mgr.lclog_export

Parameter Name	Type
share_path	FileOnShare (for CIFS & NFS Share) LocalFile (For Local Share)
Description	LC log exports can be directed to local file systems and network shares NFS - Network File System IPAddress - Name of the NFS share server. Sharename - The NFS share name. Mountpoint - To mount the shared file. UserCredentials - Username and password for accessing the shared file. CIFS - Common Internet File System IPAddress - Name of the CIFS share server. Sharename - The CIFS share name. Sharepath - The share path where file needs to be exported. UserCredentials - Username and password for accessing the shared file.

Example:

NOTE: Use the Export LC Logs as required along with the FileOnShare API.

NOTE: The export format is filename.gz.

```
#localpath = LocalFile(local='/newfolder/completelclogs.gz')
status=idrac.log_mgr.complete_lclog_export(job_wait=True,share_path=localpath)

# Export LC Logs - NFS Share
```

```

nfs_share = FileOnShare(remote=<IP OR HOSTNAME>:/<NFS-SHARE-PATH>,
    mount_point=<MOUNT-DRIVE>:\>, isFolder=<True/False>,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))

lclog_file = nfs_share.new_file(<FILE-NAME>)

msg = idrac.log_mgr.lclog_export(lclog_file)

# Export LC Logs - CIFS Share
cifs_share = FileOnShare(remote=\\<IP OR HOSTNAME>\<CIFS-SHARE-PATH>, isFolder=<True/False>,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))

lclog_file = cifs_share.new_file(<FILE-NAME>)

msg = idrac.log_mgr.lclog_export(lclog_file)

# Export LC Logs - Local Share
local_share = LocalFile(local=os.path.join(, "path", "to", "lc-logs-file.xml"))
export_lclog_streaming = idrac.log_mgr.lclog_export(share_path=local_share)

```

Returns: Success or Failure

Return type: JSON

idrac.log_mgr.complete_lclog_export

API string/Method: idrac.log_mgr.complete_lclog_export

Protocol Support: WSMAN.

Description: The complete_lclog_export() method is used to export the full log from the Lifecycle Controller to a remote share.

Parameters:

Table 49. Parameters for idrac.log_mgr.complete_lclog_export

Parameter Name	Type	Allowed Values
job_wait	Boolean	True/False
Description	True - It will wait for the export the full LC logs job to complete and return the job completion status. False - It will return immediately with a JOB ID after queuing the export LC logs job in LC job	

Table 50. Parameters for idrac.log_mgr.complete_lclog_export

Parameter Name	Type
share_path	FileOnShare (for CIFS & NFS Share) LocalFile (For Local Share)
Description	LC log exports can be directed to local file systems and network shares NFS - Network File System IPAddress - Name of the NFS share server. Sharename - The NFS share name. Mountpoint - To mount the shared file. UserCredentials - Username and password for accessing the shared file.

Parameter Name	Type
	CIFS - Common Internet File System IPAddress - Name of the CIFS share server Sharename - The CIFS share name. Sharepath - The share path where file needs to be exported. UserCredentials - Username and password for accessing the shared file.

Example:

```
# Export LC Full Logs - NFS Share
nfs_share = FileOnShare(remote=<IP OR HOSTNAME>:</NFS-SHARE-PATH>,
    mount_point=<MOUNT-DRIVE>:\>, isFolder=<True/False>,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))

tsr_file = nfs_share.new_file(<FILE-NAME>)

idrac.log_mgr.complete_lclog_export(tsr_file)

# Export LC Full Logs - CIFS Share
cifs_share = FileOnShare(remote=\\<IP OR HOSTNAME>\<CIFS-SHARE-PATH>, isFolder=<True/False>,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))

tsr_file = cifs_share.new_file(<FILE-NAME>)

idrac.log_mgr.complete_lclog_export(tsr_file)

# Export LC Full Logs - Local Share
local_share = LocalFile(local=os.path.join("path", "to", <FILE_NAME>))
idrac.log_mgr.complete_lclog_export(local_share)
```

Returns: Success or Failure.

Return type: JSON

idrac.config_mgr.export_tsr

API string/Method: idrac.config_mgr.export_tsr

Protocol Support: WSMAN.

Description: The idrac.config_mgr.export_tsr is used to collect the TSR i.e. hardware, OS and App data, then compresses the .zip file and saves it into the respective remote share path—cifs/nfs.

Parameters:

Table 51. Parameters for idrac.config_mgr.export_tsr

Parameter Name	Type	Allowed Values
data_selector_array_in	DataSelectorArrayInEnum	DataSelectorArrayInEnum.HW_Data DataSelectorArrayInEnum.OSApp_Data DataSelectorArrayInEnum.TTY_Logs DataSelectorArrayInEnum.Debug_Logs
Description	The DataSelectorArrayIn allows you to select one of the data DataSelectorArrayInEnum.HW_Data - Technical support report for Hardware Data. DataSelectorArrayInEnum.OSApp_Data - Technical support report for OSApp Data Without PII.	

Parameter Name	Type	Allowed Values
		DataSelectorArrayInEnum.TTY_Logs - Technical support report for TTY Logs. DataSelectorArrayInEnum.Debug_Logs - Technical support report for OSApp Data.

Table 52. Parameters for idrac.config_mgr.export_tsr

Parameter Name	Type	Allowed Values
job_wait	Boolean	True/False
Description	This API raises a flag to wait for the job to complete. A false value will return the Job ID.	

Table 53. Parameters for idrac.config_mgr.export_tsr

Parameter Name	Type	Allowed Values
tsr_store_path	FileOnShare—for CIFS and NFS Share	
Description	This method collects the TSR i.e hardware, OS and App data, then compresses and saves the .zip file to remote share path. The FileOnShare API allows you to view the share path where file needs to be exported.	

Example:

```
#Export TSR - NFS Share
nfs_share = FileOnShare(remote=<IP OR HOSTNAME>:/<NFS-SHARE-PATH>,
    mount_point=<MOUNT-DRIVE>:\>, isFolder=<True/False>,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))

tsr_file = nfs_share.new_file(<FILE-NAME>)

idrac.config_mgr.export_tsr(tsr_file, data_selector_array_in = DataSelectorArrayInEnum.HW_Data)

# Export TSR - CIFS Share
cifs_share = FileOnShare(remote=\\<IP OR HOSTNAME>\<CIFS-SHARE-PATH>, isFolder=<True/False>,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))

tsr_file = cifs_share.new_file(<FILE-NAME>)

idrac.config_mgr.export_tsr(tsr_file, data_selector_array_in = DataSelectorArrayInEnum.HW_Data)
```

Returns: Success or Failure.

Return type: JSON

Server iDRAC Settings Configuration API list

This section lists the available Server iDRAC Settings Configuration APIs.

iDRAC User Configuration

idrac.user_mgr.Users.new

idrac.user_mgr.Users.new

API string/Method: idrac.user_mgr.Users.new

Protocol Support: Redfish.

Description: The `idrac.user_mgr.Users.new()` returns a JSON to create the user. For more details about the properties and values, refer to iDRAC Attribute Registry Guide.

Table 54. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
UserName_Users	String	User defined string
Description	Allows you to create a user name.	

Table 55. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
Password_Users	String	User defined string
Description	Allows you to create a password.	

Table 56. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
Privilege_Users	ENUM	Privilege_UsersTypes.Administrator Privilege_UsersTypes.NoAccess Privilege_UsersTypes.Readonly Privilege_UsersTypes.Operator
Description	Allows you to create a privileged user,	

Table 57. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
IpmlanPrivilege_Users	ENUM	IpmlanPrivilege_UsersTypes.Administrator IpmlanPrivilege_UsersTypes.No_Access IpmlanPrivilege_UsersTypes.Operator IpmlanPrivilege_UsersTypes.User
Description	Allows you to set IPMI LAN privilege for users.	

Table 58. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
<code>IpmiSerialPrivilege_Users</code>	ENUM	<code>IpmiSerialPrivilege_UsersTypes.Administrator</code> <code>IpmiSerialPrivilege_UsersTypes.No_Access</code> <code>IpmiSerialPrivilege_UsersTypes.Operator</code> <code>IpmiSerialPrivilege_UsersTypes.User</code>
Description	Allows you to set IPMI serial privilege for users.	

Table 59. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
<code>Enable_Users</code>	ENUM	<code>Enable_UsersTypes.Enabled</code> <code>Enable_UsersTypes.Disabled</code>
Description	Allows you to enable or disable a user.	

Table 60. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
<code>SolEnable_Users</code>	ENUM	<code>SolEnable_UsersTypes.Enabled</code> <code>SolEnable_UsersTypes.Disabled</code>
Description	Allows you to enable or disable a SOL user.	

Table 61. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
<code>ProtocolEnable_Users</code>	ENUM	<code>ProtocolEnable_UsersTypes.Enabled</code> <code>ProtocolEnable_UsersTypes.Disabled</code>
Description	Allows you to enable or disable a protocol for a user.	

Table 62. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
<code>AuthenticationProtocol_Users</code>	ENUM	<code>AuthenticationProtocol_UsersTypes.SHA</code> <code>AuthenticationProtocol_UsersTypes.MD5</code> <code>AuthenticationProtocol_UsersTypes.T_None</code>
Description	Allows you to set the authentication protocol.	

Table 63. Parameters for idrac.user_mgr.Users.new

Parameter Name	Type	Allowed Values
PrivacyProtocol_Users	ENUM	PrivacyProtocol_UsersTypes.AES PrivacyProtocol_UsersTypes.DES PrivacyProtocol_UsersTypes.T_None
Description	Allows you to set the privacy protocol for a user.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

    idrac.user_mgr.Users.new
    (
        UserName_Users="Abc123",
        Password_Users="Xyz123",
        Privilege_Users=Privilege_UsersTypes.Administrator,
        IpmiLanPrivilege_Users=IpmiLanPrivilege_UsersTypes.Administrator,
        IpmiSerialPrivilege_Users=IpmiSerialPrivilege_UsersTypes.Administrator,
        Enable_Users=Enable_UsersTypes.Enabled,
        SolEnable_Users=SolEnable_UsersTypes.Enabled,
        ProtocolEnable_Users=ProtocolEnable_UsersTypes.Disabled,
        AuthenticationProtocol_Users=AuthenticationProtocol_UsersTypes.SHA,
        PrivacyProtocol_Users=PrivacyProtocol_UsersTypes.AES
    )
    apply_status = idrac.config_mgr.apply_changes(reboot=False)
```

Returns: JSON Returns a json indicating whether the user was created successfully or not.

Return Type: JSON.

idrac.user_mgr.Users.remove

idrac.user_mgr.Users.remove

API string/Method: idrac.user_mgr.Users.remove

Protocol Support: Redfish.

Description: The idrac.user_mgr.Users.remove() returns a JSON to remove the user. For more details about the properties and values, refer to iDRAC Attribute Registry Guide.

Table 64. Parameters for idrac.user_mgr.Users.remove

Parameter Name	Type	Allowed Values
UserName_Users	String	User Defined string
Description	Allows you to remove a user.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
```

```

        creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

        idrac.user_mgr.Users.remove(Username_Users="XYZ123")
        apply_status = idrac.config_mgr.apply_changes(reboot=False)

```

Returns: JSON Returns a json indicating whether the user was deleted successfully or not.

Return Type: JSON.

iDRAC Network Settings Configuration

idrac.config_mgr.configure_dns

API string/Method: idrac.config_mgr.configure_dns

Protocol Support: WSMAN, Redfish.

Description: The idrac.config_mgr.configure_dns allows you to configure the DNS server.

Table 65. Parameters for idrac.config_mgr.configure_dns

Parameter Name	Type	Allowed Values
register_idrac_on_dns	Enum	DNSRegister_NICTypes.Enabled DNSRegister_NICTypes.Disabled
Description	Allows you to register the iDRAC on DNS.	

Table 66. Parameters for idrac.config_mgr.configure_dns

Parameter Name	Type	Allowed Values
dns_idrac_name	String	None
Description	Allows you to configure the DNS iDRAC name.	

Table 67. Parameters for idrac.config_mgr.configure_dns

Parameter Name	Type	Allowed Values
auto_config	Enum	DNSDomainNameFromDHCP_NICTypes.Enabled DNSDomainNameFromDHCP_NICTypes.Disabled
Description	Allows you to enable or disable the DNS name from the DHCP	

Table 68. Parameters for idrac.config_mgr.configure_dns

Parameter Name	Type	Allowed Values
static_dns	String	User defined values
Description	Configure the DNS server.	

Example:

```

#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",

```

```

        mount_point='Z:', isFolder=True,
        creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

#Configure iDRAC DNS Configuration
msg = idrac.config_mgr.configure_dns(register_idrac_on_dns = DNSRegister_NICTypes.Enabled,
dns_idrac_name = None,
    auto_config = DNSDomainNameFromDHCP_NICTypes.Disabled, static_dns = None)
apply_status = idrac.config_mgr.apply_changes(reboot=False)

```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_ipv4

API string/Method: idrac.config_mgr.configure_ipv4

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_ipv4() allows you to configure the IPv4 Settings.

Table 69. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
ip_address	String	IP Address
Description	Enter an IP (IPv4) address or a hostname of the shared folder. The maximum character limit is 64 .	

Table 70. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
enable_dhcp	Enum	DHCPEnable_IPv4Types.Enabled DHCPEnable_IPv4Types.Disabled
Description	Indicates whether the DHCP server is Enabled or Disabled .	

Table 71. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
dns_1	String	User defined values
Description	Indicates the static DNS server IPv4 address if DHCP is not used to get the DNS address.	

Table 72. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
dns_2	String	User defined values
Description	Indicates the static alternate DNS server IPv4 address if DHCP is not used to get the DNS address.	

Table 73. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
dns_from_dhcp	Enum	DNSFromDHCP_IPv4Types.Enabled DNSFromDHCP_IPv4Types.Disabled
Description	Indicates whether DHCP is used to obtain primary and secondary DNS server addresses from	

Table 74. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
enable_ipv4	Enum	Enable_IPv4Types.Enabled Enable_IPv4Types.Disabled
Description	Indicates whether the IPv4 protocol is Enabled or Disabled .	

Table 75. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
gateway	String	User defined values
Description	Configure the Default Gateway IP address.	

Table 76. Parameters for idrac.config_mgr.configure_ipv4

Parameter Name	Type	Allowed Values
net_mask	String	User defined values
Description	Configure the iDRAC subnet mask.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

    idrac.config_mgr.configure_ipv4
    (
        ip_address = "1.1.1.1", enable_dhcp = DHCPEnable_IPv4Types.Enabled,
        dns_1 = None, dns_2 = None, dns_from_dhcp = DNSFromDHCP_IPv4Types.Enabled,
        enable_ipv4 = Enable_IPv4Types.Enabled, gateway = None, net_mask = None
    )
    apply_status = idrac.config_mgr.apply_changes(reboot=False)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_static_ipv4

API string/Method: idrac.config_mgr.configure_static_ipv4.

Protocol Support: Redfish.

Description: The `idrac.config_mgr.configure_static_ipv4()` allows you to configure the Static IPv4 Settings.

Table 77. Parameters for `idrac.config_mgr.configure_static_ipv4`

Parameter Name	Type	Allowed Values
<code>ip_address</code>	String	IP Address
Description	Displays the iDRAC NIC IPv4 address.	

Table 78. Parameters for `idrac.config_mgr.configure_static_ipv4`

Parameter Name	Type	Allowed Values
<code>dns_1</code>	String	User defined values
Description	Indicates the static DNS server IPv4 address if DHCP is not used to get the DNS address.	

Table 79. Parameters for `idrac.config_mgr.configure_static_ipv4`

Parameter Name	Type	Allowed Values
<code>dns_2</code>	String	User defined values
Description	Indicates the static alternate DNS server IPv4 address if DHCP is not used to get the DNS address.	

Table 80. Parameters for `idrac.config_mgr.configure_static_ipv4`

Parameter Name	Type	Allowed Values
<code>dns_from_dhcp</code>	Enum	<code>DNSFromDHCP_IPv4Types.Enabled</code> <code>DNSFromDHCP_IPv4Types.Disabled</code>
Description	Indicates whether DHCP is used to obtain primary and secondary DNS server addresses from the DHCPv4 server.	

Table 81. Parameters for `idrac.config_mgr.configure_static_ipv4`

Parameter Name	Type	Allowed Values
<code>gateway</code>	String	User defined values
Description	Configure the iDRAC subnet mask.	

Table 82. Parameters for `idrac.config_mgr.configure_static_ipv4`

Parameter Name	Type	Allowed Values
<code>net_mask</code>	String	User defined values
Description	Configure the iDRAC subnet mask.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
```

```

        mount_point='Z:', isFolder=True,
        creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

#Perform iDRAC Static IPv4 configuration
    idrac.config_mgr.configure_static_ipv4
    (
        ip_address = "1.1.1.1", dns_1 = None, dns_2 = None,
        dns_from_dhcp = DNSFromDHCP_IPv4StaticTypes.Enabled, gateway=None,
        net_mask=None)
    apply_status = idrac.config_mgr.apply_changes(reboot=False)

```

Returns: Success of Failure.

Return Type: JSON.

idrac.config_mgr.configure_timezone

API string/Method: idrac.config_mgr.configure_timezone

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_timezone() allows you to configure the time zone Settings.

Table 83. Parameters for idrac.config_mgr.configure_timezone

Parameter Name	Type	Allowed Values
timezone	String	Timezone
Description	Configures the time zone.	

Example:

```

#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:</NFS-SHARE-PATH>/<FILE-NAME>",
    mount_point='Z:', isFolder=False,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

# Configure Timezone
msg = idrac.config_mgr.configure_timezone('US/Pacific')
apply_status = idrac.config_mgr.apply_changes(reboot=False)

```

Returns: Success or Failure.

Return Type: JSON.

iDRAC Services Configuration API list

idrac.config_mgr.configure_web_server

API string/Method: idrac.config_mgr.configure_web_server

Protocol Support: WSMAN, Redfish.

Description: The idrac.config_mgr.configure_web_server() allows you to configure web server Settings.

Table 84. Parameters for idrac.config_mgr.configure_web_server

Parameter Name	Type	Allowed Values
enable_web_server	Enum	Enable_WebServerTypes.Enabled Enable_WebServerTypes.Disabled
Description	Select Enabled to enable the iDRAC web server. This setting is enabled by default .  NOTE: Disabling the web server disables the current web interface session and you cannot log in to iDRAC web interface or use remote RACADM.	

Table 85. Parameters for idrac.config_mgr.configure_web_server

Parameter Name	Type	Allowed Values
http_port	Number	HTTP Port number
Description	Enter the port number that iDRAC uses to listen for a browser connection. The default value is 80 . The value must be 1–65535.	

Table 86. Parameters for idrac.config_mgr.configure_web_server

Parameter Name	Type	Allowed Values
https_port	Number	HTTPS Port number
Description	Enter the port number that iDRAC uses to listen for a secure browser connection. The default value is 443 . The value must be 1–65535.	

Table 87. Parameters for idrac.config_mgr.configure_web_server

Parameter Name	Type	Allowed Values
timeout	Number	Timeout seconds
Description	Enter the time (in seconds) for which a connection is allowed to remain Idle. The session is canceled when the time-out is reached. Changes to the timeout setting do not affect the current session. When you change this time, you must logout and log in again for the new setting to take effect. Timeout range is 60–10,800 seconds. The default value is 1800 seconds .	

Table 88. Parameters for idrac.config_mgr.configure_web_server

Parameter Name	Type	Allowed Values
ssl_encryption	Enum	SSLEncryptionBitLength_WebServerTypes.T.Auto_Negotiate SSLEncryptionBitLength_WebServerTypes.T.128-bit or higher SSLEncryptionBitLength_WebServerTypes.T.168-bit or higher

Parameter Name	Type	Allowed Values
		SSLEncryptionBitLength_WebServerTypes.T_256_Bit_or_higher
Description	To specify the level of SSL encryption for providing authenticated and encrypted communication between clients and servers, select one of the following options: Auto Negotiate—Enables auto-negotiation of the SSL encryption between client and the server by using all industry standard encryption algorithms. NOTE: This option includes weaker SSL encryption algorithm, which reduces the security. 128-bit or higher—Enables SSL encryption between client and server using the industry standard 128-bit or higher. 168-bit or higher—Enables SSL encryption between client and server using 168-bit or higher. 256-bit or higher—Enables SSL encryption between client and server using 256-bit or higher. If you select 256-bit or higher and are using Java, the following warning message is displayed: Selecting a higher SSL encryption enhances the security. However, the cryptography settings for your virtual machine environment (JVM, IcedTea) may require installing the Unlimited Strength Java TM Cryptography Extension Policy Files to permit usage of iDRAC plugins such as vConsole with this higher level of encryption. NOTE: Using this level of encryption may have import or export implications. For more guidance, contact your legal department.	

Table 89. Parameters for idrac.config_mgr.configure_web_server

Parameter Name	Type	Allowed Values
tls_protocol	Enum	TLSProtocol_WebServerTypes.TLS_1_0_and_Higher TLSProtocol_WebServerTypes.TLS_1_1_and_Higher TLSProtocol_WebServerTypes.TLS_1_2_and_Higher
Description	To specify the minimum supported level of TLS protocol, select one of the following options: TLS 1.0 and Higher TLS 1.1 and Higher TLS 1.2 Only NOTE: TLS 1.1 and Higher is the default option. For better security, Dell recommends to use TLS 1.2.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
```

```

        creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

#Configure iDRAC WebServer
    idrac.config_mgr.configure_web_server
    (
        enable_web_server = Enable_WebServerTypes.Enabled,
        http_port = 80,
        https_port = 443,
        timeout = 1800,
        ssl_encryption = SSLEncryptionBitLength_WebServerTypes.T.T_128_Bit_or_higher,
        tls_protocol = TLSProtocol_WebServerTypes.TLS_1_1_and_Higher
    )
    apply_status = idrac.config_mgr.apply_changes(reboot=False)

```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_snmp

API string/Method: idrac.config_mgr.configure_snmp

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_snmp() allows you to configure SNMP Settings.

Table 90. Parameters for idrac.config_mgr.configure_snmp

Parameter Name	Type	Allowed Values
snmp_enable	Enum	AgentEnable_SNMPTypes.Disabled AgentEnable_SNMPTypes.Enabled
Description	Allows you to enable to disable the SNMP server.	

Table 91. Parameters for idrac.config_mgr.configure_snmp

Parameter Name	Type	Allowed Values
community_name	String	User provided input
Description	Enter the SNMP community name. The name can have up to 31 nonblank characters. The default name is public . iDRAC uses it to validate SNMP queries and gets (received from remote systems requesting SNMP data Access from iDRAC).	

Table 92. Parameters for idrac.config_mgr.configure_snmp

Parameter Name	Type	Allowed Values
alert_port	Number	Alert port number
Description	Enter the SNMP port number that must be used for SNMP traps. The default value is 162 . Range is 1-65535.	

Table 93. Parameters for idrac.config_mgr.configure_snmp

Parameter Name	Type	Allowed Values
discovery_port	Number	Discovery port number
Description	Allows the server to discover the available port.	

Table 94. Parameters for idrac.config_mgr.configure_snmp

Parameter Name	Type	Allowed Values
snmp_protocol	Enum	SNMPProtocol_SNMPTypes.All SNMPProtocol_SNMPTypes.SNMPv3
Description	Allows you to select one of the SNMP formats.	

Table 95. Parameters for idrac.config_mgr.configure_snmp

Parameter Name	Type	Allowed Values
trap_format	Enum	TrapFormat_SNMPTypes.SNMPv1 TrapFormat_SNMPTypes.SNMPv2 TrapFormat_SNMPTypes.SNMPv3
Description	Allows you to select one of the SNMP trap formats.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

# Configure SNMP Settings
msg = idrac.config_mgr.configure_snmp
(
    snmp_enable = AgentEnable_SNMPTypes.Disabled,
    community_name = "test",
    snmp_protocol = SNMPProtocol_SNMPTypes.SNMPv3,
    alert_port = 161,
    discovery_port = 162,
    trap_format = TrapFormat_SNMPTypes.SNMPv3
)
apply_status = idrac.config_mgr.apply_changes(reboot=False)
```

Returns: Success or Failure.

Return Type: JSON.

iDRAC System Settings API list

This section list the available iDRAC system setting APIs.

Syslog

idrac.config_mgr.enable_syslog

API string/Method: idrac.config_mgr.enable_syslog

Protocol Support: WSMAN.

Description: The idrac.config_mgr.enable_syslog() enables System Log configuration.

Allows you to enable the System Log configuration.

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
                      mount_point='Z:\', isFolder=True,
                      creds=UserCredentials(<USERNAME>, <PASSWORD>))

liason_share_status = idrac.config_mgr.set_liason_share(myshare)

# Enable Syslog
msg = idrac.config_mgr.enable_syslog()
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.disable_syslog

API string/Method: idrac.config_mgr.disable_syslog.

Protocol Support: WSMAN.

Description: The idrac.config_mgr.disable_syslog() enables System Log configuration.

Allows you to disable the System Log configuration.

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
                      mount_point='Z:\', isFolder=True,
                      creds=UserCredentials(<USERNAME>, <PASSWORD>))

liason_share_status = idrac.config_mgr.set_liason_share(myshare)

# Disable Syslog
msg = idrac.config_mgr.disable_syslog()
```

Returns: Success or Failure.

Return Type: JSON.

Lockdown

idrac.config_mgr.enable_system_lockdown

API string/Method: idrac.config_mgr.enable_system_lockdown

Protocol Support: WSMAN, Redfish.

Description: The `idrac.config_mgr.enable_system_lockdown()` method allows you to enable the Lockdown operation.

NOTE: Lockdown is supported only on 14th Generation of PowerEdge Servers

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
                      mount_point='Z:', isFolder=True,
                      creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)
status=idrac.config_mgr.enable_csior()
apply_status = idrac.config_mgr.apply_changes(reboot=False)
print(apply_status)

# Enable System Lockdown
msg = idrac.config_mgr.enable_system_lockdown()
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.disable_system_lockdown

API string/Method: `idrac.config_mgr.disable_system_lockdown`.

Protocol Support: WSMAN, Redfish.

Description: The `idrac.config_mgr.disable_system_lockdown()` performs a Lockdown disable operation.

NOTE: Lockdown is supported only on 14th Generation of PowerEdge Servers.

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
                      mount_point='Z:', isFolder=True,
                      creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)
status=idrac.config_mgr.disable_csior()
apply_status = idrac.config_mgr.apply_changes(reboot=False)
print(apply_status)

# Enable System Lockdown
msg = idrac.config_mgr.disable_system_lockdown()
```

Returns: Success or Failure.

Return Type: JSON.

CSIOR

idrac.config_mgr.enable_csior

API string/Method: `idrac.config_mgr.enable_csior`

Protocol Support: WSMAN.

Description: The `idrac.config_mgr.enable_csior()` method allows you to enable the CSIOR option in iDRAC which enables the collection of system inventory on reboot.

CSIOR is an option in iDRAC which enables collecting of system inventory on reboot. It allows you the CSIOR to perform configuration.

Example:

NOTE: Use the Export LC Logs as required along with the [FileOnShare API](#).

```
# Export LC Logs - NFS Share
nfs_share = FileOnShare(remote=<IP OR HOSTNAME>:/<NFS-SHARE-PATH>,
                        mount_point=<MOUNT-DRIVE>:\>, isFolder=<True/False>,
                        creds=UserCredentials(<USERNAME>, <PASSWORD>))

lclog_file = nfs_share.new_file(<FILE-NAME>)

msg = idrac.log_mgr.lclog_export(lclog_file)

# Export LC Logs - CIFS Share
cifs_share = FileOnShare(remote=\\<IP OR HOSTNAME>\<CIFS-SHARE-PATH>, isFolder=<True/False>,
                        creds=UserCredentials(<USERNAME>, <PASSWORD>))

liason_share_status = idrac.config_mgr.set_liason_share(myshare)
status=idrac.config_mgr.enable_csior()
apply_status = idrac.config_mgr.apply_changes(reboot=False)
print(apply_status)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.disable_csior

API string/Method: idrac.config_mgr.disable_csior

Protocol Support: WSMAN.

Description: The idrac.config_mgr.disable_csior() method allows you to disable the CSIOR option in iDRAC which disables the collection of system inventory on reboot.

Example:

NOTE: Use the Export LC Logs as required along with the [FileOnShare API](#).

```
# Export LC Logs - NFS Share
nfs_share = FileOnShare(remote=<IP OR HOSTNAME>:/<NFS-SHARE-PATH>,
                        mount_point=<MOUNT-DRIVE>:\>, isFolder=<True/False>,
                        creds=UserCredentials(<USERNAME>, <PASSWORD>))

lclog_file = nfs_share.new_file(<FILE-NAME>)

msg = idrac.log_mgr.lclog_export(lclog_file)

# Export LC Logs - CIFS Share
cifs_share = FileOnShare(remote=\\<IP OR HOSTNAME>\<CIFS-SHARE-PATH>, isFolder=<True/False>,
                        creds=UserCredentials(<USERNAME>, <PASSWORD>))

liason_share_status = idrac.config_mgr.set_liason_share(myshare)
status=idrac.config_mgr.disable_csior()
apply_status = idrac.config_mgr.apply_changes(reboot=False)
print(apply_status)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_idrac_alerts

API string/Method: idrac.config_mgr.configure_idrac_alerts

Protocol Support: Redfish.

Description: The `idrac.config_mgr.configure_idrac_alerts()` allows you to enable or disable the iDRAC alerts Settings.

Table 96. Parameters for `idrac.config_mgr.configure_idrac_alerts`

Parameter Name	Type	Allowed Values
enable_alerts	Enum	AlertEnable_IPMILanTypes.Enabled AlertEnable_IPMILanTypes.Disabled
Description	Configure the iDRAC alert settings.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:\', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

#Alert Configuration
    idrac.config_mgr.configure_idrac_alerts
        (
            enable_alerts=AlertEnable_IPMILanTypes.Disabled
        )
    apply_status = idrac.config_mgr.apply_changes(reboot=False)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_snmp_trap_destination

API string/Method: `idrac.config_mgr.configure_snmp_trap_destination`

Protocol Support: Redfish.

Description: The `idrac.config_mgr.configure_snmp_trap_destination()` allows you to configure the SNMP Trap Destination Settings.

Table 97. Parameters for `idrac.config_mgr.configure_snmp_trap_destination`

Parameter Name	Type	Allowed Values
destination_number	Number	Destination number
Description	Displays up to 4 IPv4 and IPv6 addresses.	

Table 98. Parameters for `idrac.config_mgr.configure_snmp_trap_destination`

Parameter Name	Type	Allowed Values
destination	String	Destination Address
Description	To receive trap alerts, enter the address (IPv4, IPv6, or FQDN). If all destination addresses display existing IP addresses, you have configured all the existing alert destinations and must reuse a disabled alert.	

Parameter Name	Type	Allowed Values
	 NOTE: The destination community must be the same as the iDRAC community.	

Table 99. Parameters for `idrac.config_mgr.configure_snmp_trap_destination`

Parameter Name	Type	Allowed Values
<code>snmp_v3_username</code>	String	User defined string
Description	From the drop-down list, select the SNMP v3 user to whom you want to send the SNMP v3 format traps.	

Table 100. Parameters for `idrac.config_mgr.configure_snmp_trap_destination`

Parameter Name	Type	Allowed Values
<code>state</code>	Enum	State_SNMPLAlertTypes.Enabled State_SNMPLAlertTypes.Disabled
Description	To enable the IP address to receive traps, select this option.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>/<FILE-NAME>",
                      mount_point='Z:', isFolder=False,
                      creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

# Configure SNMP Trap Destination Settings
idrac.config_mgr.configure_snmp_trap_destination
(
    destination_number = 1,
    destination = "1.1.1.1",
    snmp_v3_username = None,
    state = State_SNMPLAlertTypes.Disabled
)
apply_status = idrac.config_mgr.apply_changes(reboot=False)
```

Returns: Success or Failure.

Return Type: JSON.

`idrac.config_mgr.configure_smtp_server_settings`

API string/Method: `idrac.config_mgr.configure_smtp_server_settings`

Protocol Support: Redfish.

Description: The `idrac.config_mgr.configure_smtp_server_settings()` allows you to configure the SMTP Server Settings.

Table 101. Parameters for idrac.config_mgr.configure_smtp_server_settings

Parameter Name	Type	Allowed Values
smtp_ip_address	String	SMTP IP Address
Description	Enter the IPv4 or IPv6 address of the SMTP server or the FQDN / DNS name. The IP address entered must be in the dot separated format. For example, 143.166.154.1 .	

Table 102. Parameters for idrac.config_mgr.configure_smtp_server_settings

Parameter Name	Type	Allowed Values
smtp_port	Number	SMTP Port number
Description	Enter the destination port for email alerts. The default value is 25 . Range is 1-65535.	

Table 103. Parameters for idrac.config_mgr.configure_smtp_server_settings

Parameter Name	Type	Allowed Values
authentication	Enum	SMTPAuthentication_RemoteHostsTypes.Enabled SMTPAuthentication_RemoteHostsTypes.Disabled
Description	To specify the user name and password for SMTP mail server authentication, select this option.	

Table 104. Parameters for idrac.config_mgr.configure_smtp_server_settings

Parameter Name	Type	Allowed Values
username	String	User defined string
Description	To connect to the SMTP server, enter the user name.	

Table 105. Parameters for idrac.config_mgr.configure_smtp_server_settings

Parameter Name	Type	Allowed Values
password	String	User defined string
Description	To connect for NFS or CIFS file system, enter the password. The maximum length is 255 characters.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>/<FILE-NAME>",
    mount_point='Z:', isFolder=False,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

    idrac.config_mgr.configure_smtp_server_settings
    (
        smtp_ip_address = "1.1.1.1", smtp_port = 26,
        authentication = SMTPAuthentication_RemoteHostsTypes.Enabled,
        username = None, password = None
```

```
)
apply_status = idrac.config_mgr.apply_changes(reboot=False)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_email_alerts

API string/Method: idrac.config_mgr.configure_email_alerts

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_email_alerts() allows you to configure email alerts Settings.

Table 106. Parameters for idrac.config_mgr.configure_email_alerts

Parameter Name	Type	Allowed Values
alert_number	Number	Alert number
Description	Configure up to 4 email destinations that can be set to receive alerts.	

Table 107. Parameters for idrac.config_mgr.configure_email_alerts

Parameter Name	Type	Allowed Values
state	Enum	Enable_EmailAlertTypes.Enabled Enable_EmailAlertTypes.Disabled
Description	Configure to receive the alerts, select this option.	

Table 108. Parameters for idrac.config_mgr.configure_email_alerts

Parameter Name	Type	Allowed Values
address	String	Destination email Address
Description	Enter the email address that must receive the alerts.	

Table 109. Parameters for idrac.config_mgr.configure_email_alerts

Parameter Name	Type	Allowed Values
custom_message	String	Custom Message
Description	Allows you to enter a custom email message.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

    idrac.config_mgr.configure_email_alerts
    (
        alert_number = 4, state=Enable_EmailAlertTypes.Enabled,
```

```

        address = "test@abc.com", custom_message = "test"
    )
    apply_status = idrac.config_mgr.apply_changes(reboot=False)

```

Returns: Success or Failure.

Return Type: JSON.

Boot Settings Configuration API list

This section lists the available Boot settings configuration APIs.

idrac.config_mgr.configure_boot_mode

API string/Method: idrac.config_mgr.configure_boot_mode

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_boot_mode() allows you to configure the boot Mode.

Table 110. Parameters for idrac.config_mgr.configure_boot_mode

Parameter Name	Type	Allowed Values
boot_mode	Enum	BootModeTypes.Bios BootModeTypes.Uefi
Description	Configures the boot mode to BIOS or UEFI.	

Example:

```

#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=False,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

# Configure Boot Mode
msg = idrac.config_mgr.configure_boot_mode(BootModeTypes.Uefi)
apply_status = idrac.config_mgr.apply_changes(reboot=True)

```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_onetime_boot_mode

API string/Method: idrac.config_mgr.configure_onetime_boot_mode

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_onetime_boot_mode() allows you to set the specified device as the first device in the boot order for the next boot cycle only. The device must be a device from the bootseq option device list.

Table 111. Parameters for idrac.config_mgr.configure_onetime_boot_mode

Parameter Name	Type	Allowed Values
onetime_boot_mode	Enum	OneTimeBootModeTypes.Disabled OneTimeBootModeTypes.OneTimeBootSeq OneTimeBootModeTypes.OneTimeCustomBootSeqStr OneTimeBootModeTypes.OneTimeCustomHddSeqStr OneTimeBootModeTypes.OneTimeCustomUefiBootSeqStr OneTimeBootModeTypes.OneTimeHddSeq OneTimeBootModeTypes.OneTimeUefiBootSeq
Description	Allows you to enable or disable the one time boot mode.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

idrac.config_mgr.configure_onetime_boot_mode(onetime_boot_mode=OneTimeBootModeTypes.OneTimeUefiBootSeq)
apply_status = idrac.config_mgr.apply_changes(reboot=True)
```

Returns: Success or Failure.**Return Type:** JSON.

idrac.config_mgr.configure_boot_sequence

API string/Method: idrac.config_mgr.configure_boot_sequence**Protocol Support:** Redfish.**Description:** The idrac.config_mgr.configure_boot_sequence() allows you to configure the boot sequence.**Table 112. Parameters for idrac.config_mgr.configure_boot_sequence**

Parameter Name	Type	Allowed Values
boot_mode	Enum	BootModeEnum.Bios BootModeEnum.Uefi
Description	Allows you to configure the boot mode to BIOS or UEFI.	

Table 113. Parameters for idrac.config_mgr.configure_boot_sequence

Parameter Name	Type	Allowed Values
boot_sequence	String	Boot devices FQDDs (Comma separated) in the sequential order for BIOS or UEFI Boot Sequence.
Description	User defined boot sequence.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

    idrac.config_mgr.configure_boot_sequence
    (
        boot_mode = BootModeEnum.Uefi,
        boot_sequence="device-1, device-2, device-3"
    )
    apply_status = idrac.config_mgr.apply_changes(reboot=True)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_nvme_mode

API string/Method: idrac.config_mgr.configure_nvme_mode

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_nvme_mode() allows you to configure the NVME Mode.

Table 114. Parameters for idrac.config_mgr.configure_nvme_mode

Parameter Name	Type	Allowed Values
nvme_mode	Enum	NvmeModeTypes.NonRaid NvmeModeTypes.Raid
Description	Configure the NVME mode. This feature is available in the BIOS settings page.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

    idrac.config_mgr.configure_nvme_mode(
        nvme_mode=NvmeModeTypes.Raid)
    apply_status = idrac.config_mgr.apply_changes(reboot=True)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.configure_secure_boot_mode

API string/Method: idrac.config_mgr.configure_secure_boot_mode

Protocol Support: Redfish.

Description: The idrac.config_mgr.configure_secure_boot_mode() allows you to configure the Secure boot Mode.

Table 115. Parameters for idrac.config_mgr.configure_secure_boot_mode

Parameter Name	Type	Allowed Values
secure_boot_mode	Enum	SecureBootModeTypes.AuditMode SecureBootModeTypes.DeployedMode SecureBootModeTypes.SetupMode SecureBootModeTypes.UserMode
Description	Configures how the BIOS uses the Secure boot Policy.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",
    mount_point='Z:', isFolder=True,
    creds=UserCredentials(<USERNAME>, <PASSWORD>))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)

    idrac.config_mgr.configure_secure_boot_mode(
        secure_boot_mode=SecureBootModeTypes.AuditMode)
apply_status = idrac.config_mgr.apply_changes(reboot=True)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.boot_to_network_iso

API string/Method: idrac.config_mgr.boot_to_network_iso

Protocol Support: WSMAN.

Description: The idrac.config_mgr.boot_to_network_iso() allows you to boot the provided ISO file for operating system deployment.

Table 116. Parameters for idrac.config_mgr.boot_to_network_iso

Parameter Name	Type	Allowed Values
myshare	String	User defined network bootable ISO path.
Description	Allows you to configure the network boot mode.	

Example:

```
#Set liason share
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>/<ISO Image name>",
```

```

        mount_point='Z:',
        isFolder=True,
        creds=UserCredentials(<USERNAME>, <PASSWORD>))
msg = idrac.config_mgr.boot_to_network_iso(myshare, ' ' )

```

Returns: Success or Failure.

Return Type: JSON.

idrac.update_mgr.update_from_repo

API string/Method: idrac.update_mgr.update_from_repo.

Protocol Support: WSMAN.

Description: The idrac.update_mgr.update_from_repo() allows you to update the server components firmware.

Table 117. Parameters for idrac.update_mgr.update_from_repo

Parameter Name	Type	Allowed Values
catalog_path	Enum	User defined network catalog path.
Description	Allows you to select the path of the catalog.	

Table 118. Parameters for idrac.update_mgr.update_from_repo

Parameter Name	Type	Allowed Values
apply_update	Boolean	True
Description	Allows you to apply the update to the firmware.	

Table 119. Parameters for idrac.update_mgr.update_from_repo

Parameter Name	Type	Allowed Values
reboot_needed	Boolean	True or False
Description	Allows you to perform a reboot.	

Table 120. Parameters for idrac.update_mgr.update_from_repo

Parameter Name	Type	Allowed Values
job_wait	Boolean	True/False
Description	True - Wait for the performed export Server Configuration Profile jobs to complete. False - Returns only the JobID.	

Example:

```

#Set liason share
myshare=FileOnShare(remote='<IP or Hostname>:<NFS Share Path>/<File name its Catalog.xml >',
mount_point='<Mount Path>',creds=UserCredentials('<Username>', '<Password>'))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)
status = idrac.update_mgr.update_from_repo(
                                catalog_path=myshare,

```

```
apply_update=True,  
reboot_needed=True,  
job_wait=True  
)
```

Returns: Success or Failure.

Return Type: JSON.

RAID Configuration API list

This section lists the available RAID configuration APIs.

idrac.config_mgr.RaidHelper.new_virtual_disk

API string/Method: idrac.config_mgr.RaidHelper.new_virtual_disk

Protocol Support: WSMAN, Redfish.

Description: The idrac.config_mgr.RaidHelper.new_virtual_disk() allows you to create a RAID.

Table 121. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
vd_name	String	User defined
Description	Name of the virtual disk.	

Table 122. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
Span_depth	Number	User defined values
Description	Depth of the disk span.	

Table 123. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
Span_length	Number	User defined values
Description	Length of the disk span.	

Table 124. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDTypes	ENUM	RAIDTypesTypes.RAID_0 RAIDTypesTypes.RAID_1 RAIDTypesTypes.RAID_10 RAIDTypesTypes.RAID_5 RAIDTypesTypes.RAID_50

Parameter Name	Type	Allowed Values
		RAIDTypesTypes.RAID_6 RAIDTypesTypes.RAID_60
Description	Type of the RAID array.	

Table 125. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDaction	ENUM	RAIDactionTypes.Create
Description	RAID action to be performed.	

Table 126. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDinitOperation	ENUM	RAIDinitOperationTypes.Fast RAIDinitOperationTypes.T_None
Description	Initializes a RAID operation.	

Table 127. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
DiskCachePolicy	ENUM	DiskCachePolicyTypes.Default DiskCachePolicyTypes.Enabled DiskCachePolicyTypes.Disabled
Description	Allows you to set the default disk cache policy.	

Table 128. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDdefaultWritePolicy	ENUM	RAIDdefaultWritePolicyTypes.WriteThrough RAIDdefaultWritePolicyTypes.WriteBackForce RAIDdefaultWritePolicyTypes.WriteBack
Description	Allows you to set the default RAID write policy.	

Table 129. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDdefaultReadPolicy	ENUM	RAIDdefaultReadPolicyTypes.Adaptive RAIDdefaultReadPolicyTypes.AdaptiveReadAhead

Parameter Name	Type	Allowed Values
		RAIDdefaultReadPolicyTypes.NoReadAhead RAIDdefaultReadPolicyTypes.ReadAhead
Description	Allows you to set the default RAID read policy.	

Table 130. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDEnhancedAutoImportForeignConfig	ENUM	RAIDEnhancedAutoImportForeignConfigTypes.Enabled RAIDEnhancedAutoImportForeignConfigTypes.Disabled
Description	Allows you to configure the auto import setting.	

Table 131. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDresetConfig	ENUM	RAIDresetConfigTypes.T_False RAIDresetConfigTypes.T_True
Description	Allows you to reset the RAID configuration.	

Table 132. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDbgiRate	Number	User defined RAID bgi rate.
Description	Allows you to set the speed of the background initialization of the virtual disks.	

Table 133. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDprMode	ENUM	RAIDprModeTypes.Automatic RAIDprModeTypes.Disabled RAIDprModeTypes.Manual
Description		

Table 134. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDrebuildRate	Number	User defined RAID rebuild rate.
Description	Allows you to set the RAID rebuild speed.	

Table 135. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDforeignConfig	ENUM	RAIDforeignConfigTypes.Clear RAIDforeignConfigTypes.Ignore RAIDforeignConfigTypes.Import
Description	Allows you to set the external configuration types.	

Table 136. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDreconstructRate	Number	User defined RAID reconstruct rate.
Description	Allows you to set the RAID reconstruct rate.	

Table 137. Parameters for idrac.config_mgr.RaidHelper.new_virtual_disk

Parameter Name	Type	Allowed Values
RAIDccMode	ENUM	RAIDccModeTypes.Normal RAIDccModeTypes.StopOnError
Description	Allows you to configure the CC mode.	

Example:

```
myshare=FileOnShare(remote='<IP OR Hostname>:/<NFS Share Path>/',
                    mount_point='<mount path>',
                    isFolder=True,
                    creds=UserCredentials('<username>', '<password>'))
liason_share_status = idrac.config_mgr.set_liason_share(myshare)
pd_filter = '((disk.parent.parent is Controller and disk.parent.parent.FQDD._value ==
"{0}")'.format('RAID.Mezzanine.1-1' )
pd_filter += ' or (disk.parent is Controller and disk.parent.FQDD._value ==
"{0}")'.format('RAID.Mezzanine.1-1')
msg=idrac.config_mgr.RaidHelper.new_virtual_disk
(
    vd_name="newvdomsdk",
    SpanDepth=1,
    SpanLength=1,
    NumberDedicatedHotSpare=1,
    NumberGlobalHotSpare=1,
    RAIDTypes=RAIDTypesTypes.RAID_0,
    RAIDaction=RAIDactionTypes.Create,
    RAIDinitOperation=RAIDinitOperationTypes.T_None,
    DiskCachePolicy=DiskCachePolicyTypes.Default,
    RAIDdefaultWritePolicy=RAIDdefaultWritePolicyTypes.WriteThrough,
    RAIDdefaultReadPolicy=RAIDdefaultReadPolicyTypes.NoReadAhead,
    StripeSize=64 * 1024,
    RAIDEnhancedAutoImportForeignConfig=RAIDEnhancedAutoImportForeignConfigTypes.Enabled,
    RAIDresetConfig=RAIDresetConfigTypes.T_False,
    RAIDbgiRate="30",
    RAIDprMode=RAIDprModeTypes.Automatic,
    RAIDrebuildRate="30",
    RAIDforeignConfig=RAIDforeignConfigTypes.Clear,
    RAIDreconstructRate="30",
    RAIDccMode=RAIDccModeTypes.Normal,
    PhysicalDiskFilter=pd_filter
```

```
)  
apply_status = idrac.config_mgr.apply_changes(reboot=True)
```

Returns: Success or Failure.

Return Type: JSON.

idrac.config_mgr.RaidHelper.delete_virtual_disk

API string/Method: idrac.config_mgr.RaidHelper.delete_virtual_disk

Protocol Support: WSMAN, Redfish.

Description: The idrac.config_mgr.RaidHelper.delete_virtual_disk() allows you to create a RAID.

Table 138. Parameters for idrac.config_mgr.RaidHelper.delete_virtual_disk

Parameter Name	Type	Allowed Values
Name	String	User defined
Description	Name of the virtual disk.	

Example:

```
#Set liason share  
myshare = FileOnShare(remote="<IP OR HOSTNAME>:/<NFS-SHARE-PATH>",  
    mount_point='Z:\', isFolder=True,  
    creds=UserCredentials(<USERNAME>, <PASSWORD>))  
liason_share_status = idrac.config_mgr.set_liason_share(myshare)  
  
    idrac.config_mgr.RaidHelper.delete_virtual_disk  
    (  
        Name="Virtual Disk 0"  
    )  
    apply_status = idrac.config_mgr.apply_changes(reboot=True)
```

Returns: Success or Failure.

Return Type: JSON.

iDRAC LC Jobs API list

This section lists the available iDRAC LC Jobs APIs.

idrac.job_mgr.get_job_status

API string/Method: idrac.job_mgr.get_job_status

Protocol Support: WSMAN.

Description: The get_job_status(jobid) method is used to get the job status for the provided job ID.

Example:

```
# Get Job Status  
job_status = idrac.job_mgr.get_job_status(jobid="JID_123456789012").
```

Returns: Success or Failure.

Return Type: JSON.

idrac.job_mgr.delete_job

API string/Method: idrac.job_mgr.delete_job

Protocol Support: WSMAN.

Description: The delete_job(jobid) method is used to delete a job.

Example:

```
# Delete a job
job_status = idrac.job_mgr.delete_job(jobid="JID_123456789012")
```

Returns: Success or Failure.

Return Type: JSON.

idrac.job_mgr.delete_all_jobs

API string/Method: idrac.job_mgr.delete_all_jobs

Protocol Support: WSMAN.

Description: The delete_all_jobs() is used to delete all jobs.

Example:

```
# Delete all jobs
job_status = idrac.job_mgr.delete_all_jobs()
```

Returns: Success or Failure.

Return Type: JSON.

iDRAC LC Status Check API list

This section lists the available iDRAC LC Status Check APIs.

idrac.config_mgr.LCStatus

API string/Method: idrac.config_mgr.LCStatus

Protocol Support: Redfish.

Description: The idrac.config_mgr.LCStatus() method is used to check whether LC is in use or Ready.

Example:

```
# Check the LC status
msg = idrac.config_mgr.LCStatus
```

Returns: In Use or Ready.

Return Type: String.

idrac.config_mgr.LCReady

API string/Method: idrac.config_mgr.LCReady

Protocol Support: WSMAN.

Description: This property provides the information regarding whether the LC is Ready.

Example:

```
msg = idrac.config_mgr.LCReady
```

Returns: True or False.

Return Type: String.

Getting Started

The following sections provide information about using OMPSDK to connect, disconnect and communicate with the device.

Topics:

- [OMPSDK Infrastructure](#)
- [API to setup share](#)
- [iDRAC server information](#)
- [Administration Tasks](#)
- [Server configuration profile overview](#)
- [Import export server configuration profiles](#)
- [Exporting LC Logs](#)
- [Server iDRAC settings and configuration](#)
- [BIOS Boot settings](#)
- [RAID Configuration](#)
- [iDRAC LC jobs](#)
- [iDRAC LC status check](#)

OMPSDK Infrastructure

This section provides information to initialize to OMPSDK infrastructure.

Initialization of OMPSDK Infrastructure

Run the following command to initialize the OMPSDK infrastructure

```
sdkinfra
```

```
importPath
```

The above two methods initialize the SDK Infrastructure and loads the device drivers such as iDRAC, CMC, F10 etc present in the path `<python_lib> omdrivers`.

Table 139. Drivers list

Method	Description
<code>importPath</code>	This function reads all the driver files that exist in the omdrivers directory.

Table 140. Monitoring API methods

Method	Description
<code>get_driver</code>	This function creates and returns the driver instance based on the input details including driver name and on successful connection. User can specify a particular Driver type [e.g iDRAC, CMC, F10] and get the driver instance back without looping through all the drivers present inside omdrivers folder.
Parameters: <code>driver_en</code>	Device driver enumerator. For example: iDRAC : <code>sd.driver_enum.iDRAC</code> , CMC, Compellent.
<code>ipaddr</code>	Device IP address or the hostname.
<code>creds</code>	Device credentials - bundle of credentials for finding the device driver.
<code>protopref</code>	The preferred protocol to be used if the device supports the protocol.
<code>pOptions</code>	Other protocol specific parameters options to be passed. For example: SNMP timeout, retry, port.
Returns	The driver handle for further configuration or monitoring.
<code>find_driver</code>	This function creates and returns the driver instance based on the input details without any specific driver name and on successful connection. It queries with the given device by trying each available driver in omdrivers directory until it finds a appropriate driver. Once the driver is found, it returns the driver instance.
Parameters: <code>ipaddr</code>	Device IP address or the hostname.
<code>creds</code>	Device credentials.
<code>protopref</code>	The preferred protocol to be used if the device supports the protocol.
<code>pOptions</code>	Other protocol specific parameters options to be passed. For example: SNMP timeout, retry, port.
Returns	The device handle for further configuration or monitoring.

Table 141. Base Driver

Method	Description
<code>get_entityjson</code>	Returns the complete entity details in JSON format.
Returns	True if successful
<code>get_partial_entityjson</code>	Return the entity details in JSON format, after applying the filtration criteria.
Parameters: <code>en</code>	List of Components of the device whose JSON is required
Returns	JSON only the components passed in <code>en</code> .

Table 142. Base Driver Properties

Method	Description
<code>driver_enum</code>	Returns an python enumeration of the available drivers in the system.
<code>ComponentEnum</code>	Returns the list of components supported for the device.
<code>ContainmentTree</code>	Returns the containment tree of iDRAC.
<code>device_type</code>	Returns the type of this device.

API to setup share

The following section provides information about the network share.

Setting up a remote network share

To setup a network file share run the following command:

NOTE: This network share is required to have a share between iDRAC and OMP SDK for configuration.

Pre-requisite - The share needs to be manually mounted.

```
FileOnShare(remote = '<public_IP>:/Share', mount_point = '/mnt/Share',  
            isFolder = True, creds)  
idrac.config_mgr.set_liason_share(myshare)
```

FileOnShare methods are: Network file share (NFS), Common internet file system (Windows Style, CIFS), Template (%D, %M, %Y... %ip), access credentials.

Table 143. Methods of remote path

Command	Description
<code>remote path= IP:/Share</code>	For network file share (NFS).
<code>remote path= \\IP\Share</code>	For Common Internet File System (Windows) (CIFS).

Table 144. Methods of mount point

Command	Description
<code>mount_point= /mnt/share</code>	For systems running on Linux.
<code>mount_point= Z:\\</code>	For systems running on Windows.

iDRAC server information

The export server profile and import server profile features in iDRAC, that allows IT administrators to do a backup and restore configuration and firmware for a PowerEdge server. User can import and export from local management station, and from a Network Share via NFS or CIFS. Using SCP, you can select and import or export component level configurations for BIOS, NIC and RAID.

Administration Tasks

Power Configuration

Enables you to remotely perform power control operations on the managed system like: power-on, power-off and power reset.

`idrac.config_mgr.power_boot(Power_state)`: Allows you to power On/Off the Server.

iDRAC Reset configuration

`disconnect()`: Disconnects from device.

`reset()`: Disconnects and clean up the internal cache.

`reconnect()`: Disconnects and reconnects to device.

Server configuration profile overview

Server profile import and export is a feature of Dell Lifecycle Controller that enables IT administrators capture and restore BIOS and firmware information in any system state. This capability provides a solution for hardware or firmware problems in scenarios such as motherboard replacement.

Import export server configuration profiles

The following section informs about import and export server configuration profiles.

Export server configuration profile

You can export Server Configuration Profile (SCP) with various components such as iDRAC, BIOS, NIC, RAID together or with one of these components. You can export SCP from iDRAC to a local or a network shared location. For shared location, make sure that a network share path is established. While exporting SCP to Network File System (NFS) or (CIFS) share, provide the IP address of the network share where the exported file needs to reside in the `share_name` parameter.

`idrac.config_mgr.scp_export()`: Method used to export the system configuration.

`synchronous`: When the status is synchronous, the server tracks end to end job completion providing the message ID or job ID, status of the job and completion of the job.

`asynchronous` - In asynchronous status the job in the server returns the job ID and result of the job.

`job_wait = True|False`. Use the `job_mgr` APIs to wait for the job to complete or get status.

Export modes are Clone or Replace. Clone mode exported SCP can be imported into another device. If you have to replicate the SCP to another systems, you should use export in Clone mode. If you are using the replace mode, then it would retire the entire server.

While exporting you can export to a template `%ip_%D%M%Y_%H%S.xml`.

NOTE: Export features using CIFS shares may fail if there is a mismatch of the SMB version in iDRAC and OS.

Import server configuration profile

You can import the Server Configuration Profile (SCP) which was previously exported for that same server, or group of servers. Importing SCP is useful in restoring the configuration of the server to the state stored in the profile. You can import SCP from a local or a remote share to iDRAC. For a remote share, make sure that a network share path and the file name are available.

`idrac.config_mgr.scp_import()`: Method used to import the system configuration.

Exporting LC Logs

iDRAC provides Lifecycle log that contains events related to system, storage devices, network devices, firmware updates, configuration changes, license messages, and so on.

`idrac.log_mgr.lclog_export`: Allows you to export the log from the Lifecycle Controller to a remote share.

Export Complete LC Logs

`idrac.log_mgr.complete_lclog_export()`: Allows you to export the full log from the Lifecycle Controller to a remote share.

Export TSR Logs

`idrac.config_mgr.export_tsr()`: Allows you to collect the TSR i.e hardware, OS and App data, then compressed the .zip file save into the respective remote share path.

Server iDRAC settings and configuration

The following attributes provide the information about the server iDRAC settings and configurations.

- iDRAC User Configuration
- iDRAC Network Configuration
- iDRAC Service Configuration
- iDRAC Systems Settings

iDRAC User Configuration settings

The following API's provides the user credentials settings for the WSMAN.

`idrac.user_mgr.User.new()`: Creates the user credentials for WSMAN communication.

`idrac.user_mgr.Users.remove()`: Deletes the user credentials for WSMAN communication.

iDRAC Network Configuration

The following API's provides information about the network configuration API's available.

- iDRAC DNS configuration
- iDRAC DHCP configuration
- iDRAC Static IP configuration
- iDRAC Time Zone configuration

iDRAC DNS configuration

`idrac.config_mgr.configure_dns()`: Allows you to configure the DNS server.

iDRAC DHCP configuration

`idrac.config_mgr.configure_ipv4()`: Allows you to configure the DHCP settings.

iDRAC Static IP configuration

`idrac.config_mgr.configure_static_ipv4()`: Allows you to configure the static IP settings.

iDRAC Time Zone configuration

`idrac.config_mgr.configure_timezone()`: Allows you to configure the time zone settings.

iDRAC Service Configuration

The following API's provide information of various service configurations available.

- SNMP Agent Configuration
- Alert Configuration
- WebServer Configuration

SNMP Agent Configuration

`idrac.config_mgr.snmp()`: Allows you to configure the SNMP Settings.

Alert Configuration

`idrac.config_mgr.configure_idrac_alerts()`: Allows you to enable or disable the iDRAC alerts Settings.

Webserver Configuration

`idrac.config_mgr.configure_web_server()`: Allows you to configure web server Settings.

iDRAC System Settings

The following API's provide information of various systems settings available.

- SysLog Configuration
- LockDown Configuration
- CSIOR Configuration
- SNMP traps Configuration
- SMTP (E-Mail) Configuration

SysLog Configuration

`idrac.config_mgr.enable_syslog()`: Allows you to enable the system log configuration.

`idrac.config_mgr.disable_syslog()`: Allows you to disable the system log configuration.

LockDown Configuration

`idrac.config_mgr.enable_system_lockdown()`: Allows you to enable the lockdown operation.

`idrac.config_mgr.disable_system_lockdown()`: allows you to disable the lockdown operation.

CSIOR Configuration

`idrac.config_mgr.enable_csior()`: Allows you to enable the CSIOR option in iDRAC which enables the collection of system inventory on reboot.

`idrac.config_mgr.disable_csior()`: Allows you to disable the CSIOR option in iDRAC which disables the collection of system inventory on reboot.

SNMP traps Configuration

`idrac.config_mgr.configure_snmp_trap_destination()`: Allows you to configure the SNMP trap destination settings.

SMTP (E-Mail) Configuration

`idrac.config_mgr.configure_email_alerts()`: Allows you to configure to receive the alerts.

BIOS Boot settings

The following section provides the information about the system BIOS Boot settings.

BIOS Boot settings

Using the System BIOS Settings utility, you can set the managed system to boot. The managed system attempts to boot from a bootable device based on the boot order.

- `idrac.config_mgr.configure_boot_mode()`: Allows you to configure the boot mode.
- `idrac.config_mgr.configure_onetime_boot_mode()`: Allows you to set the specified device as the first device in the boot order for the next boot cycle.
- `idrac.config_mgr.configure_boot_sequence()`: Allows you to configure the boot sequence.
- `idrac.config_mgr.configure_nvme_mode()`: Allows you to configure the NVME mode.

RAID Configuration

Storage Management enables you to perform controller and enclosure functions for all supported RAID and non-RAID controllers and enclosures. You can configure and manage the controller functions without accessing the BIOS.

Create Virtual Disk

A virtual disk refers to a storage created by a RAID controller from one or more physical disks. Although a virtual disk may be created from several physical disks, it is viewed by the operating system as a single disk.

`idrac.config_mgr.RaidHelper.new_virtual_disk()`: To create the virtual disk.

Delete Virtual Disk

Deleting a virtual disk destroys all information including file systems and volumes residing on the virtual disk.

`idrac.config_mgr.RaidHelper.delete_virtual_disk()`: To delete the virtual disk.

iDRAC LC jobs

The following section provides information about the status of jobs.

Job API's

The following API's provide the information about the job status from LC.

Delete single job

`idrac.job_mgr.delete_job()`: Allows you to delete the job.

Delete all jobs

`idrac.job_mgr.delete_all_jobs()`: Allows you to delete all jobs.

View job status

`idrac.job_mgr.get_job_status()`: Allows you to get the job status for the provided job ID.

iDRAC LC status check

The following API provides the information to check the LC status.

`idrac.config_mgr.LCStatus()`: Provides the Status of LifeCycle Controller.