Provisioning Web Services (PWS)

This chapter provides an overview of PWS, and describes the PWS data types, PWS operations, and some sample use cases of PWS usage in Prime Cable Provisioning.

Overview

The Provisioning Web Services (PWS) component provides a SOAP/RESTful based web interface that supports provisioning operation. The provisioning services include functionalities such as: adding, retrieving, updating, and removing objects necessary to support the provisioning and configuration generation of CPEs. The PWS objects include devices, classes of service, DHCP criteria, groups, and files.

The PWS component manages the following activities:

- Exposes a provisioning web service that provides functionality similar to the current API client.
- Supports stateless interactions.
- Supports both synchronous and asynchronous requests.
- Supports singular and plural operations.
- Supports both stop on failure and ignore on failure.
- Supports request that can operate on multiple objects.
- Supports SOAP v1.1 and 1.2.
- Supports WSDL v1.1.
- Supports WS-I Basic Profile v1.1.
- Supports both HTTP and HTTPS transport.
- Supports RESTful.

The web service is hosted on a Tomcat container, and it is recommended that you install it on a separate server.

For more information on PWS, see the *Cisco Prime Cable Provisioning 6.1.3 User Guide*.

PWS Concepts

The PWS component provides the following functionalities for running an operation on the device:

Session Management

The session management functionality enables you to establish a session between the WS client and the RDU to process multiple requests from WS client. A WS client can establish connection with multiple RDUs simultaneously. The client runs a **createSession** operation, and in response receives a context object on successful authentication. The context object identifies the session between the client and the RDU, and can be used as an authentication token to run all the requests in the session.

The client closes the session after processing all the requests. The session also gets closed automatically after a configured period of idle time. The default idle timeout is configured using a command line script for all the sessions. If the session gets closed and the client tries to reestablish the session, the session gets reestablished with the same session id.

Transactionality

The PWS operations can be classified into two categories:

- Single device operations - One operation for a single device. For single device operations, an operation is a single transaction in which all the changes are made or no change is made to the device.
- Multiple device operations - One operation for multiple devices. For multiple device operations, the transaction scope can be set to include all devices, or have each device change in its individual transaction.

You can set the execution option, transactionPerItem, to true to retrieve the transaction data for each batch. The OperationStatus returned will contain the status of the entire operation as well as the individual status if multiple transactions are used.

Error Handling

When a PWS operation fails, you may receive the following SOAP faults:

- ProvServiceException - A ProvServiceException is a generic fault that describes trouble with WS in handling the operation request.
- AccessDeniedException - Indicates that the user does not have proper privileges or the credentials are wrong.

PWS Data Types

The following data types are defined in WSDL that are supported by PWS:

- DeviceType - Attribute that defines the type of the device. For example, Computer; DOCSISModem; PacketCableMTA; CableHomeWanMan; CableHomeWanData; STB; eRouter; RPD.
- ClassOfService (COS) - Attribute that defines the class of service.
- DHCPCriteria - Attribute that defines the DHCP criteria.
- File - Attribute the defines the static configuration file, groovy scripts or dynamic configuration files, jar files, MIB files, template files, firmware image, and other generic files.
- Group - Attribute that defines a group. The devices are assigned to a group based on a specific logical criterion.

- Context - Attribute that defines the authentication data for a client or a client's session.
- Options - Attribute that helps you to customize the WS client's request either at the PWS level or the RDU level. The Options data type is categorized into the following two types:
  - Execution Options - Used to customize the execution process of the WS client's request. For example, you can run the client's request in reliable mode. In case of reliable mode, when the RDU receives a reliable request, the RDU persists the batch until it executes successfully. If the RDU restarts or the WS client restarts, a record of the Batch is retained. The RDU stores the last 1000 reliable batch responses.

Table 8-1 describes the execution options for PWS client's request.

Table 8-1 Execution Options for PWS Client's Request

Execution Options	Value
ActivationMode	AUTOMATIC- This mode specifies that the RDU will perform all the necessary steps to activate the devices in a batch. The batch processing in this mode involves performing batch writes to the database, generating a new device, configuration of the device the batch affects, downloading the new configuration to the appropriate DPE(s), and disrupting the device NO_ACTIVATION - Disable the AUTOMATIC activation mode.
ConfirmationMode	NO_CONFIRMATION - This mode specifies that the RDU will allow the batch to proceed, even if there is an error or warning during device disruption. CUSTOM_CONFIRMATION - This mode is currently ignored by the RDU and is equivalent to NO_CONFIRMATION.
PublishingMode	NO_PUBLISHING - This mode specifies that the RDU will not publish changes from this batch. PUBLISHING_CONFIRMATION - This mode specifies that the RDU will publish the changes from this batch, but if the publishing fails, the RDU will fail the batch. RDU will roll back all the publishing changes and any changes made by the batch. PUBLISHING_NO_CONFIRMATION- This mode specifies the RDU will publish the changes from this batch, but if publishing fails, the RDU will not fail the batch. RDU will roll back the publishing for this batch.
asynchronous	true - Activate asynchronous mode. The status of the web service operation will represent whether the RDU interaction was successful false - Deactivate asynchronous mode. The web service operation is blocked until the RDU interaction completes. The default value is false.
reliableMode	true - Activate reliable mode. false - Deactivate reliable mode. The default value is false.
timeout	Used to specify the timeout value in milliseconds.

Table 8-1 Execution Options for PWS Client's Request (continued)

Execution Options	Value
stopOnFailure	true - RDU will not execute the consecutive batch if the current batch processing fails. false - RDU will run all the batches even if any batch fails. The default value is true.
transactionPerItem	true - RDU will return the result of each individual transaction. false - No individual transaction result is available. The default value is false.

  - Operation Options - Used to customize the operation process of WS client's request. For example, to retrieve the lease data from a getDevice() request, add the flag **includeLeaseInfo** in the getDevice() request and set its value to **True**. The individual properties of the lease data are retrieved in the response. If you want to delete the devices behind the managed device, add the flag **deletedevicesbehind** in the deleteDevice() or deleteDevices() request and set its value to **True**.

- DeviceId - Attribute that allocates a unique identity to the device. The supported unique identities are types of device ids such as MACAddressType, DUIDType and FQDNType.
- Properties - Attribute that defines the provisioning criteria for the devices. The provisioning criteria is defined in terms of key-value list, where the key and value are both strings.
- SearchResult - Attribute that is used to retrieve the search results.
- Search - Attribute used to search devices, files, and groups based on a specific logical criterion. The QueryType attribute is used to define the logical criterion for the search operation. Table 8-2 describes the supported logical criteria for the search operation.

Table 8-2 QueryType for Search Operation

QueryType	Description
DeviceSearchByCOS	Used to search devices that are associated with a specific class of service.
DeviceSearchByDHCPCriteria	Used to search devices that are associated with a specific DHCP criteria.
DeviceSearchByDefaultCOS	Used to search devices that are associated with the default class of service of a specific device type.
DeviceSearchByDefaultDHCP Criteria	Used to search devices that are associated with the default DHCP criteria of a specific device type.
DeviceSearchByDeviceType	Used to search devices that are associated with a specific device type.
DeviceSearchByGroupName	Used to search devices that are associated with a specific group.
DeviceSearchByProvGroupNa me	Used to search devices that are associated with a specific provisioning group
DeviceSearchByDeviceIdPatte rn	Used to search devices that are associated with a specific device identifier pattern. For example, you can use this attribute to search all the devices whose MAC address starts with "1,6,".

Table 8-2 QueryType for Search Operation (continued)

QueryType	Description
DeviceSearchByOwnerId	Used to search devices that are associated with a specific owner identifier. For a search request with this QueryType, no paging support is available from RDU and all the devices associated with this specific owner id are retrieved in a single instance.
FileSearchByFileNamePattern	Used to search files that are associated with a specific file name pattern.
FileSearchByFileType	Used to search files that are associated with a specific file type.
GroupSearchByGroupType	Used to search groups that are associated with a specific group type.
GroupSearchByRelatedGroup	Used to search groups that are related to specific group.
GroupSearchByGroupNamePattern	Used to search groups that are associated with a specific group name pattern.
CosSearchByDeviceType	Used to search class of services that are associated with a particular device type. For a search request with this QueryType, no paging support is available from RDU and all the class of services associated with this specific DeviceType are retrieved in a single instance.
DHCPCriteriaSearch	Used to search the DHCP Criteria. For a search request with this QueryType, no paging support is available from RDU and all DHCP criteria are retrieved in a single instance.

- PropertyFilter - Attribute that specifies the properties that you want to include in the response. If the PropertyFilter data type is set, only the listed properties will be returned.
- OperationStatus - Attribute that displays the response of a web service operation.
- DeviceOperationStatus - Attribute that displays the response of a web service operation on devices that involves returning operation result for multiple devices. In addition to the attributes of OperationStatus data type, this object also includes a device object. On success a device object with a SUCCESS code is returned. On failure, the operation status is returned with the description about the issue/failure.
- DevicesBehindOperationStatus - Attribute that displays the response of a web service operation on list of devices behind the specified devices.
- DeviceTypeOperationStatus - Attribute that displays the response of a web service operation on a specific device type, for example; Computer, DOCSISModem, PacketCableMTA, CableHomeWanMan, CableHomeWanData, STB, and eRouter. In addition to the attributes of OperationStatus data type, this object also includes a DeviceType object. On success, a DeviceType object with a SUCCESS code is returned. On failure, the operation status is returned with the description about the issue/failure.
- LeaseResultsOperationStatus - Attribute that displays the response of a web service operation for DHCP lease information query. In addition to the attributes of OperationStatus data type, this object also includes a DHCPLeaseinfo object. On success a DHCPLeaseinfo object with a SUCCESS code is returned. On failure, the operation status is returned with the description about the issue/failure.
- ClassOfServiceOperationStatus - Attribute that displays the response of a web service operation for a class of service query. In addition to the attributes of OperationStatus data type, this object also includes a ClassOfService object. On success a ClassOfService object with a SUCCESS code is returned. On failure, the operation status is returned with the description about the issue/failure.

- DHCPCriteriaOperationStatus - Attribute that displays the response of a web service operation for a DHCP criteria query. In addition to the attributes of OperationStatus data type, this object also includes a DHCPCriteria object. On success a DHCPCriteria object with a SUCCESS code is returned. On failure, the operation status is returned with the description about the issue/failure.
- FileOperationStatus - Attribute that displays the response of a web service operation for a File query. In addition to the attributes of OperationStatus data type, this object also includes a File object. On success, a File object with a SUCCESS code is returned. On failure, the operation status is returned with the description about the issue/failure.
- GroupOperationStatus - Attribute that displays the response of a web service operation for a group query. In addition to the attributes of OperationStatus data type, this object also includes a Group object. On success, a Group object with a SUCCESS code is returned. On failure, the operation status is returned with the description about the issue/failure.

PWS Operations

A Java like syntax is used to denote the operations of the Provisioning Web Service.

This section includes the following PWS operations:

Session Operations

Table 8-3 Creating a Session

Operation	Context createSession (String rduhost, int rduport, String username, String password)
Description	Authenticates a client and establishes a session between PWS client and the PWS
Pre-Condition	All parameters are required.
Post-Condition	A successful call will result in a Context object returned that can be used for all future requests.
Execution Options	None
Operation Options	None

Table 8-3 Creating a Session (continued)

Operation	Context createSession (String rduhost, int rduport, String username, String password)
Usage Restrictions	Authorization - The user must have a valid account in either the RDU or an external AAA. Concurrent Access - There is no limitation on the number of concurrent access of this interface. Concurrent calls results in concurrent RDU requests.
Error Handling	AccessDeniedException - User does not have proper privileges or credentials are wrong. ProvServiceException - The specified RDU is not reachable. A ProvServiceException is a generic fault that reflects the WS has trouble satisfying the operation request. If this error is displayed it suggests that no sessions could be established with an RDU for example, Connection info is wrong, RDU not reachable, or some internal error. The fault should contain some info to allow the client to take corrective action e.g. Fix the distribution.

Table 8-4 Closing a Session

Operation	OperationStatus closeSession(Context context)
Description	Releases the session between the user and the web service including closing and connection with Prime CP components.
Pre-Condition	A valid context must be provided
Post-Condition	A successful call will result in the context being released. No further requests using this context will be allowed.
Execution Options	None
Operation Options	None
Usage Restrictions	Authorization - The user must have a valid context. Concurrent Access - There is no limitation on the number of concurrent access of this interface.
Error Handling	AccessDenied - User does not have proper privileges or credentials are wrong. ProvServiceException - The context is not valid.

Device Provisioning Operations

Table 8-5 Adding a New Device

Operation	OperationStatus addDevice (Context context, Device device, Options options)
Description	Submit a request for the addition of a new device.
Pre-Condition	Context must be valid. Device argument must not be null. Options argument is optional. If null, default execution options will be used.

Table 8-5 Adding a New Device (continued)

Post-Condition	A successful call will result in device being added to Prime Cable Provisioning database. A failure will result in no change to the database.
Operation Options	None
Execution Options	See Table 8-1.
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The device objects contains invalid or missing data.

Table 8-6 Adding Multiple New Devices

Operation	OperationStatus addDevices (Context context, List<Device> devices, Options options)
Description	Submit a request to add multiple new devices. Using the execution options, each device can be wrapped in a separate transaction (batch) or a single transaction.
Pre-Condition	Context must be valid. List of device argument must not be null. Options argument is optional. If null, default execution options will be used.
Post-Condition	A OperationStatus object is returned. A successful call will result in all device being added to Prime Cable Provisioning database.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The device objects contains invalid or missing data.

Table 8-7 Retrieving Data of a Specific Device

Operation	DeviceOperationStatus getDevice (Context context, DeviceId deviceId, PropertyFilter filter, Options options)
Description	Retrieve data associated with a specific device.

Table 8-7 Retrieving Data of a Specific Device (continued)

Pre-Condition	Context must be valid. A valid device ID must be provided. PropertyFilter argument is optional. If null, all properties are returned. Otherwise all properties specified will be returned. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in device data being returned otherwise a fault will be returned.
Execution Options	See Table 8-1.
Operation Options	includeLeaseInfo - Include DHCP lease info in both IPv4 and IPv6 in the result. DHCP lease information will be retrieved from provisioning group DHCP server(s). If no DHCP server responds, an error condition is returned. If a DHCP server does respond, but there is no active lease, no lease information will be included in the result and there will be no error returned.
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The device id contains invalid or missing data or uniqueness constraints were violated. Device does not exist.

Table 8-8 Retrieving Data of Multiple Devices

Operation	List<DeviceOperationStatus> getDevices(Context context, List<DeviceId> deviceId, PropertyFilter filter, Options options)
Description	Retrieve data associated with the specific devices.
Pre-Condition	Context must be valid. A list of valid device ID must be provided. List must contain at least one entry. PropertyFilter argument is optional. If null, all properties are returned. Otherwise all properties specified will be returned. Is present, will apply to all devices. Options argument is optional. If null, default execution options will be used. Options present will apply to all devices.
Post-Condition	A successful call will result in list of DeviceOperationStatus being returned.
Execution Options	See Table 8-1.
Operation Options	includeLeaseInfo - Include DHCP lease info in both IPv4 and IPv6 in the result. DHCP lease information will be retrieved from provisioning group DHCP server(s). If no DHCP server responds, an error condition is returned. If a DHCP server does respond, but there is no active lease, no lease information will be included in the result and there will be no error returned.

Table 8-8 Retrieving Data of Multiple Devices (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access – There is no limitation on the number of concurrent access of this operation.
Error Handling	ProvServiceException – The device id contains invalid or missing data or uniqueness constraints were violated. Device does not exist.

Table 8-9 Retrieving List of Devices Downstream of a specific Device

Operation	DevicesBehindOperationStatus getDevicesBehindDevice(Context context, DeviceIdSet deviceId, PropertyFilter propertyFilter, Options options)
Description	Retrieve the list of devices downstream of a specific device. Either the device ids or the entire device object is returned.
Pre-Condition	Context must be valid. A valid device ID must be provided. PropertyFilter argument is optional. If null, all properties are returned. Otherwise all properties specified will be returned. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the DevicesBehindOperationStatus object which in turn includes the list of device objects. This operation will only return device ids or device objects that are behind this device.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation. The operation will fail if the specified device has more than 100 devices behind it, and the attribute **stopOnFailure** is set to **True.**
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The device id contains invalid or missing data or device does not exist

Table 8-10 Retrieving List of Devices Downstream of Specified Devices

Operation	List<DevicesBehindOperationStatus> getDevicesBehindDevices(Context context, List<DeviceIdSet> deviceId, PropertyFilter propertyFilter, Options options)
Description	Retrieve the list of devices downstream of the specified devices. Either the device IDs or the entire device object is returned.

Table 8-10 Retrieving List of Devices Downstream of Specified Devices (continued)

Pre-Condition	Context must be valid. A valid list of device IDs must be provided containing at least one device id. PropertyFilter argument is optional. If null, all properties are returned. Otherwise all properties specified will be returned. Is present, will apply to all devices. Options argument is optional. If null, default execution options will be used. Options present will apply to all devices.
Post-Condition	A successful call will result in the list of DevicesBehindOperationStatus which in turn holds the list of target devices. This operation will only return device ids or device objects that are behind the devices.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation. For 5.0 only, supports transactionPerItem set to true. The operation will fail if the specified device has more than 100 devices behind it, and the attribute **stopOnFailure** is set to **True.**
Error Handling	ProvServiceException – The device id contains invalid or missing data or device does not exist.

Table 8-11 Updating Properties of a Device

Operation	OperationStatus updateDevice(Context context, DeviceId deviceId, Device device, PropertiesToDelete propertiesToDelete, GroupsToUnassign groupsToUnassign, Options options)
Description	Updates the properties of the specified device.
Pre-Condition	Context must be valid. A valid Device ID must be provided. A valid Device must be provided to serve as a template of changes to make. Any non-null value will be used to update that respective field. All property and their values found in Properties will be either added if the property does not exist or updated if it does exist. PropertiesToDelete specifies those properties to be removed from the Device object. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in device being updated.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-11 Updating Properties of a Device (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object and its properties. Concurrent Access - There is no limitation on the number of concurrent access of this operation. However, if multiple calls to update the same object are made, the last update will win.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The Device contains invalid data or specifies non-existent device.

Table 8-12 Updating Properties of Multiple Devices

Operation	OperationStatus updateDevices(Context context, List<DeviceId> deviceId, Device device, PropertiesToDelete propertiesToDelete, GroupsToUnassign groupsToUnassign, Options options)
Description	This operation applies the data associated with a specific device object to all the devices specified in the list of device IDs. The properties that are excluded for this operation are deviceIds, hostName, fqdn, and embeddedDevices.
Pre-Condition	Context must be valid. A valid list of DeviceId must be provided containing at least one device id. A valid Device must be provided to serve as a template of changes to make. Any non-null value will be used to update that respective field. The changes will be applied to all specified devices. All property found in Properties will be either added if the property does not exist or updated if it does exist. PropertiesToDelete specifies those properties to be removed from the Device object. Options argument is optional. If null, default execution options will be used. Options present will apply to all devices. If present in the device object passed into this operation, the changing device id is ignored in this operation. Change of deviceIsd is not supported.
Post-Condition	A list of OperationStatus objects will be returned. A successful call will result in all devices being updated.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-12 Updating Properties of Multiple Devices (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object and its properties. Concurrent Access - There is no limitation on the number of concurrent access of this operation. However, if multiple calls to update the same object are made, the last update will win.
Error Handling	ProvServiceException – The Device contains invalid data or specifies non-existent device.

Table 8-13 Deleting a Device

Operation	OperationStatus deleteDevice(Context context, DeviceId deviceid, Options options)
Description	Deletes the specified device.
Pre-Condition	Context must be valid. A valid DeviceId must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	An OperationStatus object will be returned. A successful call will result in device being deleted.
Execution Options	See Table 8-1.
Operation Options	deleteDevicesBehind– A boolean flag to determine whether to delete the devices (if any) behind the specified IP device. If deleteDevicesBehind flag is set to “true”, all devices behind the specified device will be deleted. If deleteDevicesBehind flag is set to “false”, unregistered devices behind the specified device will be deleted. Registered devices will have their discovered DHCP data and provisioning group relationship delete leaving only their registered data. The default value is false.
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required arguments or DeviceId is missing or specifies non-existent device.

Table 8-14 Deleting Multiple Devices

Operation	OperationStatus deleteDevices (Context context, List<DeviceId> deviceid, Options options)
Description	Deletes the specified devices.

Table 8-14 Deleting Multiple Devices (continued)

Pre-Condition	Context must be valid. A valid list of DeviceIds must be provided. The list must contain at least one deviceid. Options argument is optional. If null, default execution options will be used. Options present will apply to all devices.
Post-Condition	An OperationStatus object will be returned. A successful call will result in all devices being deleted.
Execution Options	See Table 8-1.
Operation Options	deleteDevicesBehind– A boolean flag to determine whether to delete the devices (if any) behind the specified IP device. If deleteDevicesBehind flag is set to “true”, all devices behind the specified device will be deleted. If deleteDevicesBehind flag is set to “false”, unregistered devices behind the specified device will be deleted. Registered devices will have their discovered DHCP data and provisioning group relationship delete leaving only their registered data. The default value is false.
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	ProvServiceException – Missing required arguments or DeviceId is missing or specifies non-existent device.

Table 8-15 Unregistering a Device

Operation	OperationStatus unregisterDevice(Context context, DeviceId deviceid, Options options)
Description	Unregister a device.
Pre-Condition	Context must be valid. A valid DeviceId must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in device being unregistered. If the device is registered, transition the device to the unregistered state. If the device is unregistered, it will be deleted from the database.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-15 Unregistering a Device (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required arguments or DeviceId is missing or specifies non-existent device.

Table 8-16 Unregistering Multiple Devices

Operation	OperationStatus unregisterDevices(Context context, DeviceId deviceid, Options options)
Description	Unregister devices.
Pre-Condition	Context must be valid. A valid list of DeviceIds must be provided containing at least one device. Options argument is optional. If null, default execution options will be used. Options are applied to the transaction as a whole.
Post-Condition	A successful call will result in all specified device being unregistered. If a device is registered, transition the device to the unregistered state. If a device is unregistered, it will be deleted from the database.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	ProvServiceException – Missing required arguments or DeviceId is missing or specifies non-existent device.

Table 8-17 Retrieving DHCP Lease Information of a IP Address

Operation	LeaseResultsOperationStatus getDHCPLeaseInfo(Context context, IPAddress ipAddress, List<String> provGroups, Options options)
Description	Retrieves all known DHCP lease information about the specified IP Address.
Pre-Condition	Context must be valid. A valid device ID must be provided. IPAddress must be provided and can be either in IPv4 or IPv6 format. An optional list of provisioning groups to search. If null, DHCP servers in all provisioning groups are queried. Options argument is optional. If null, default execution options will be used.

Table 8-17 Retrieving DHCP Lease Information of a IP Address (continued)

Post-Condition	A successful call will result in DHCP both IPv4 and IPv6 lease query information returned. DHCP lease information will be retrieved from provisioning group DHCP server(s). If no DHCP server responds, an error condition is returned. If a DHCP server does respond, but there is no active lease, no lease information will be included in the result and there will be no error returned.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The IPAddress is missing or contains invalid data. This fault will also be raised if the DHCP server could not be contacted.

Table 8-18 Regenerating Configurations for the Devices that Match the Search Criteria

Operation	OperationStatus regenConfigs(Context context, Search deviceSearch, Options options)
Description	Submit a request to the RDU's Configuration Regeneration Service to regenerate configurations for the set of devices that match the specified search criteria.
Pre-Condition	Context must be valid. A valid Search must be provided. Search.query and Search.maxResults are the only required fields. Options argument is optional. If null, default execution options will be used.
Post-Condition	This operation returns immediately. A success signifies that it has been queued by the RDU for processing. The returned OperationStatus will contain the batch id that can be used to monitor the status of the regeneration.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-18 Regenerating Configurations for the Devices that Match the Search Criteria

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the devices. Concurrent Access – Duplicate requests of this operation will be not be allowed.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong.

Table 8-19 Rebooting a Device

Operation	OperationStatus rebootDevice(Context context, DeviceId deviceid, Options options)
Description	Reboot a device.
Pre-Condition	Context must be valid. A valid DeviceId must be provided. Under Execution options, the activationMode must be AUTOMATIC.
Post-Condition	A successful call will result in device being rebooted.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required arguments or DeviceId is missing or specifies non-existent device.

DeviceType Operations

Table 8-20 Adding a New Device Type

Operation	OperationStatus addDeviceType (Context context, String deviceType, Options options)
Description	Submit a request for the addition of a new device type.
Pre-Condition	Context must be valid. DeviceType argument must not be null. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in a new device type being added to Prime CP' database.

Table 8-20 Adding a New Device Type (continued)

Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The device objects contains invalid or missing data.

Table 8-21 Retrieving Data of a Device Type Definition

Operation	DeviceTypeOperationStatus getDeviceTypes(Context context, Options options)
Description	Retrieves all the device types available in the database.
Pre-Condition	Context must be valid. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in all the device types being returned.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The device id contains invalid or missing data or uniqueness constraints were violated.

Table 8-22 Deleting a Device Type

Operation	OperationStatus deleteDeviceType(Context context, String deviceType, Options options)
Description	Deletes the specified device type.
Pre-Condition	Context must be valid. A valid DeviceType name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the device type being deleted.
Execution Options	See Table 8-1.

Table 8-22 Deleting a Device Type (continued)

Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required DeviceType name or non-existent DeviceType.

Table 8-23 Updating a Device Type

Operation	OperationStatus updateDeviceType(Context context, String deviceTypeName, Map<String, String> propertiesToUpdate, PropertiesToDelete propertiesToDelete,Options options)
Description	Updates the specified device type.
Pre-Condition	Context must be valid. A valid DeviceType name must be provided. The properties to be updated or deleted must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the device type being updated.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required DeviceType name or non-existent DeviceType.

Generic Device Operation

Table 8-24 Generic Operation

Operation	OperationStatus deviceOperation(Context context, List<DeviceId> deviceIds, DeviceCommand command,Options options)
Description	A generic operation that sends an opaque command and parameters to all specified devices.

Table 8-24 Generic Operation (continued)

	Context must be valid. A valid list of one or more DeviceIds must be provided. A valid DeviceCommand specifying the device specific operation must exist. The command and its arguments are opaque to the service. Example of operations includes generation device's configuration, resetting the device, and enabling SNMP v3 access to the device. Each DeviceCommand includes an optional set of parameters that are specific to the type of command being executed.
Pre-Condition	Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in a list of asynchronous batch status being returned. The asynchronous batch status can be used for further query in pollOperation.
Execution Options	Only asynchronous mode is supported. See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to perform device operations. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	ProvServiceException – Missing required arguments or DeviceId is missing or specifies non-existent device.

Class Of Service Operations

Table 8-25 Adding a New Class of Service

Operation	OperationStatus addClassOfService(Context context, ClassOfService cos, Options options)
Description	Submit a request for the addition of a new class of service.
Pre-Condition	Context must be valid. ClassOfService argument must not be null. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in a new COS being added to Prime CP' database.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-25 Adding a New Class of Service (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The ClassOfService object contains invalid or missing required data.

Table 8-26 Retrieving Data of a Class of Service

Operation	ClassOfServiceOperationStatus getClassOfService(Context context, String cosName, Options options)
Description	Retrieve data associated with the specified COS.
Pre-Condition	Context must be valid. A valid COS name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in COS data being returned.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The COS name is missing.

Table 8-27 Updating Properties of a COS Object

Operation	OperationStatus updateClassOfService(Context context, String cosName, ClassOfService cos, PropertiesToDelete propertiesToDelete, Options options)
Description	Updates the properties of the specified COS object.
Pre-Condition	Context must be valid. A valid COS name must be provided. A valid ClassOfService attribute must be provided. Any non-null value will be used to update that respective field. All the properties and their values are either added if the properties does not exist, or updated if the properties exist. In 5.0, renaming a COS will not be supported. PropertiesToDelete specifies those properties to be removed from the ClassOfService object. Options argument is optional. If null, default execution options will be used.

Table 8-27 Updating Properties of a COS Object (continued)

Post-Condition	A successful call will result in ClassOfService being updated.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object and its properties. Concurrent Access - There is no limitation on the number of concurrent access of this operation. However, if multiple calls to update the same object are made, the last update will win.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – A ClassOfService by the name passed does not exist or the ClassOfService contains invalid or missing data.

Table 8-28 Deleting a Class of Service

Operation	OperationStatus deleteClassOfService(Context context, String cosName, Options options)
Description	Deletes the specified ClassOfService.
Pre-Condition	Context must be valid. A valid ClassOfService name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the ClassOfService being deleted.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required ClassOfService name or non-existent ClassOfService.

DHCP Criteria Operations

Table 8-29 Adding a New DHCP Criteria

Operation	OperationStatus addDHCPCriteria(Context context, DHCPCriteria dhcpCriteria, Options options)
Description	Submit a request for the addition of a new DHCPCriteria.

Table 8-29 Adding a New DHCP Criteria (continued)

Pre-Condition	Context must be valid. DHCPCriteria argument must not be null. It must contain a valid name and at least one of the following: clientClass, includeSectionTags, and/or excludeSelectionTags. Properties are optional. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in a new DHCPCriteria being added to Prime CP' database.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The DHCPCriteria object contains invalid or missing required data.

Table 8-30 Retrieving Data of a DHCP Criteria

Operation	DHCPCriteriaOperationStatus getDHCPCriteria(Context context, String dhcpCriteriaName, Options options)
Description	Retrieve data associated with the specified DHCPCriteria.
Pre-Condition	Context must be valid. A valid DHCPCriteria name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in DHCPCriteria data being returned.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-30 Retrieving Data of a DHCP Criteria (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The DHCPCriteria name is missing or invalid.

Table 8-31 Updating Properties of a DHCP Criteria Object

Operation	OperationStatus updateDHCPCriteria(Context context, String dhcpCriteriaName, DHCPCriteria dhcpCriteria, PropertiesToDelete propertiesToDelete, Options options)
Description	Updates the properties of the specified DHCPCriteria object.
Pre-Condition	Context must be valid. A valid DHCPCriteria name must be provided. A valid DHCPCriteria must be provided. Any non-null value will be used to update that respective field. All property and their values found in Properties will be either added if the property does not exist or updated if it does exist. In 5.0, renaming a DHCPCriteria will not be supported. PropertiesToDelete specifies those properties to be removed from the DHCPCriteria object. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in DHCPCriteria being updated.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object and its properties. Concurrent Access - There is no limitation on the number of concurrent access of this operation. However, if multiple calls to update the same object are made, the last update will win.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – A DHCPCriteria by the name passed does not exist or the DHCPCriteria contains invalid data.

Table 8-32 Deleting a DHCP Criteria

Operation	OperationStatus deleteDHCPCriteria(Context context, String dhcpCriteriaName, Options options)
Description	Deletes the specified DHCPCriteria.

Table 8-32 Deleting a DHCP Criteria (continued)

Pre-Condition	Context must be valid. A valid DHCPCriteria name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the DHCPCriteria being deleted.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required DHCPCriteria name or non-existent DHCPCriteria.

File Operations

Table 8-33 Adding a New File

Operation	OperationStatus addFile(Context context, File file, Options options)
Description	Submit a request for the addition of a new File.
Pre-Condition	Context must be valid. File argument must not be null. It must contain a valid filename, filetype. Properties are optional. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in a new File being added to Prime Cable Provisioning database.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-33 Adding a New File (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The File object contains invalid or missing required data.

Table 8-34 Retrieving Data of a File

Operation	FileOperationStatus getFile(Context context, String fileName, boolean retrieveFileData, PropertyFilter propertyFilter, Options options)
Description	Retrieve data associated with the specified File.
Pre-Condition	Context must be valid. A valid filename name must be provided. PropertyFilter is optional. If null, only the file's properties will be returned and not the data. If a PropertyFilter is provided with the property fileData, all the properties including data will be returned through MTOM. Also, if retrieveFileData is set to true, all properties including the data will be returned. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in specified File properties or data being returned.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The file name is missing or invalid or the file does not exist.

Table 8-35 Updating Properties of a File Object

Operation	OperationStatus updateFile(Context context, String fileName, File file, PropertiesToDelete propertiesToDelete, Options options)
Description	Updates the properties or data of the specified File object.

Table 8-35 Updating Properties of a File Object (continued)

	Context must be valid. A valid file name must be provided. A valid File must be provided. Any non-null value will be used to update that respective field. All property and their values found in Properties will be either added if the property does not exist or updated if it does exist. In 5.0, renaming a file is not supported. PropertiesToDelete specifies those properties to be removed from the File object.
Pre-Condition	Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the File being updated.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object and its properties. Concurrent Access - There is no limitation on the number of concurrent access of this operation. However, if multiple calls to update the same object are made, the last update will win.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The file name passed, doesn't exist.

Table 8-36 Deleting a File

Operation	OperationStatus deleteFile(Context context, String fileName, Options options)
Description	Deletes the specified File.
Pre-Condition	Context must be valid. A valid File name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the File being deleted.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required File name or non-existent File.

Group Operations

Table 8-37 Adding a New Group

Operation	OperationStatus addGroup(Context context, Group group, Options options)
Description	Add a new group.
Pre-Condition	Context must be valid. Group argument must not be null. It must contain a valid name and group type. Properties are optional. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in a new Group being added to Prime Cable Provisioning database.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The Group object contains invalid or missing required data.

Table 8-38 Retrieving Data of a Group

Operation	GroupOperationStatus getGroup(Context context, String groupName, Options options)
Description	Retrieve data associated with the specified Group name.
Pre-Condition	Context must be valid. A valid Group name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in Group data being returned.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-38 Retrieving Data of a Group (continued)

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to read the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation. In 5.0, the type of the group will not be returned.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The Group name is missing or invalid, or Group does not exist.

Table 8-39 Updating Properties of a Group

Operation	OperationStatus updateGroup(Context context, String groupName, Group group, PropertiesToDelete propertiesToDelete, Options options)
Description	Updates the properties of the specified Group.
Pre-Condition	Context must be valid. A valid Group name must be provided. A valid Group must be provided. Any non-null value will be used to update that respective field. All property and their values found in Properties will be either added if the property does not exist or updated if it does exist. PropertiesToDelete specifies those properties to be removed from the Group object. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in Group being updated.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to update the object and its properties. Concurrent Access - There is no limitation on the number of concurrent access of this operation. However, if multiple calls to update the same object are made, the last update will win.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – The group name passed, doesn't exist.

Table 8-40 Deleting a Group

Operation	OperationStatus deleteGroup(Context context, String groupName, Options options)
Description	Deletes the specified Group.

Table 8-40 Deleting a Group (continued)

Pre-Condition	Context must be valid. A valid Group name must be provided. Options argument is optional. If null, default execution options will be used.
Post-Condition	A successful call will result in the Group being deleted. Any member object will be automatically unassigned.
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges to delete the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException – User does not have proper privileges or credentials are wrong. ProvServiceException – Missing required Group name or non-existent Group.

Custom Property Operations

Table 8-41 Adding a New Custom Property

Operation	OperationStatus addCustomProperty(Context context, CustomProperty customProperty, boolean encryptedProperty).
Description	Submit a request for the addition of a new custom property.
Pre-Condition	Context must be valid. CustomProperty argument must not be null. encryptedProperty value can be any boolean value.
Post-Condition	A successful call will result in a new custom property being added to Prime CP' database.
Usage Restrictions	Authorization - Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException - User does not have proper privileges or credentials are wrong. ProvServiceException - The CustomProperty object contains invalid or missing required data.

Table 8-42 Retrieving Data of a Custom Property

Operation	CustomPropertyOperationStatus getCustomProperty(Context context)
Description	Retrieve data associated with the specified Custom Property.
Pre-Condition	Context must be valid.

Table 8-42 Retrieving Data of a Custom Property (continued)

Operation	CustomPropertyOperationStatus getCustomProperty(Context context)
Post-Condition	A successful call will result in Custom property data being returned.
Usage Restrictions	Authorization - Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException - User does not have proper privileges or credentials are wrong. ProvServiceException - The CustomProperty object contains invalid or missing required data.

Table 8-43 Deleting a Custom Property

Operation	OperationStatus deleteCustomProperty(Context context, String customProperty)
Description	Deletes the specified Custom property.
Pre-Condition	Context must be valid. A valid Custom property name must be provided.
Post-Condition	A successful call will result in Custom property data being deleted.
Usage Restrictions	Authorization - Can only be called by authenticated users with valid context and proper privileges to add the object. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	AccessDeniedException - User does not have proper privileges or credentials are wrong. ProvServiceException - The CustomProperty object contains invalid or missing required data.

pollOperation Status

Table 8-44 Querying Status of the RDU Asynchronous Requests

Operation	OperationStatus pollOperationStatus(Context context, String txId, Options options)
Description	Queries for the status of RDU asynchronous requests specified by the transaction ID.
Pre-Condition	Context must be valid. A non-null txId must be provided. Options argument is optional. If null, default execution options will be used.

Table 8-44 Querying Status of the RDU Asynchronous Requests (continued)

Post-Condition	A successful call will result in the OperationStatus of the request being return. For non-reliable asynchronous RDU requests that had completed, a request not found will be returned. The support is also available for the following API requests where the OperationStatus is returned: • getDeviceTypes • getDHCPCriteria • getDHCPLeaseInfo • getClassOfService • getDevice
Execution Options	See Table 8-1.
Operation Options	None
Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	ProvServiceException – Missing required data such as request id.

Search Operation

Table 8-45 Search Operation

Operation	SearchResult search(Context context, Search search, Options options)
Description	This is a generic operation. Retrieves devices or classes of service or files or DHCP Criterion or groups based on the search criterion defined in search object.
Pre-Condition	Context must be valid. A valid Search must be provided. Search.query is the only required field. Options argument is optional, if null, default execution options will be used.
Post-Condition	A successful call will result in devices or classes of service or files or DHCP Criterion or groups objects being returned. In 5.0, for device search based on owner id, Class of Service search and DHCP Criterion search, RDU doesn't support paging while retrieving the matching results from the database. Hence for these types of search operations, all matching objects will be retrieved at one shot.
Execution Options	See Table 8-1.
Operation Options	None

Table 8-45 Search Operation

Usage Restrictions	Authorization – Can only be called by authenticated users with valid context and proper privileges. Concurrent Access - There is no limitation on the number of concurrent access of this operation.
Error Handling	ProvServiceException – The Search object contains invalid or missing data.

PWS Use Cases - SOAP

The Prime Cable Provisioning Web Services (PWS) is a SOAP/RESTful based service, the SOAP related uses cases are described in this section. The PWS Web Service Description Language (WSDL) describes the operations, messages, and data types used when interacting with the service. The PWS uses the document/literal wrapped style to encode SOAP message exchanges. With this approach the request and response messages are completely defined in W3C XML Schema. In addition, the request message takes the name of the operation while the response message takes the name of the operation with Response appended. The name prefixing can vary based on the SOAP framework being used. The following WSDL operation fragment illustrates the SOAP framework used for PWS:

```
<wsdl:operation name="getDevice">
         <soap12:operation soapAction="" style="document"/>
         <wsdl:input name="getDevice">
            <soap12:body use="literal"/>
         </wsdl:input>
         <wsdl:output name="DeviceOperationStatus">
            <soap12:body use="literal"/>
</wsdl:output>

<wsdl:message name="getDevice">
      <wsdl:part element="tns:getDevice" name="parameters"/>
       </wsdl:message>
```

The payload of the request message is described in the getDevice XML schema definition.

```
<xs:complexType name="getDevice">
            <xs:sequence>
               <xs:element name="context" type="ns1:ContextType"/>
               <xs:element name="deviceId" type="ns1:DeviceIdTypeType"/>
               <xs:element minOccurs="0" name="propertyFilter"
type="ns1:PropertyFilterType"/>
               <xs:element minOccurs="0" name="options" type="ns1:OptionsType"/>
            </xs:sequence>
</xs:complexType>
```

The getDevice request takes a required context, a required device ID, an optional property filter, and/or an optional options element.

The payload of the response message is described in the DeviceOperationStatus XML Schema definition.

```
<xs:complexType name="DeviceOperationStatus">
            <xs:sequence>
               <xs:element minOccurs="0" name="device" type="ns1:DeviceType"/>
            </xs:sequence>
</xs:complexType>
```

The DeviceOperationStatus can return a device that matches the device ID provided in the request message. If no device is found, fault is returned.

This section illustrates some of the common device related operations and SOAP request/response messages that may be contained in them. For conciseness, only the request and response messages are depicted.

Note PWS can communicate only with PCP 5.0 RDU version or above. It is not compatible with RDUs of earlier releases of Prime Cable Provisioning.

This section includes the following use cases:

Registering a New Device

For a device to be operational in the home network and to get required access, a subscriber must connect the device to the network and register it. The device could be of type DOCSISModem, PacketCableMTA, CableHomeWanMan, CableHomeWanData, Set Top Box (STB), eRouter, computer.

Desired Outcome

Use this workflow to register a device and to bring the device online with the appropriate level of service.

Step 1 Create a connection with the respective RDU by sending a create session SOAP request. The session SOAP request must contain user and RDU details as shown below:

```
<v5:createSession>
      <v5:username>user</v5:username>
      <v5:password>password</v5:password>
      <v5:rduHost>rdu-1-lnx</v5:rduHost>
      <v5:rduPort>49187</v5:rduPort>
   </v5:createSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:context>
            <cptype:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</cptype:sessionId>
         </ns2:context>
      </ns2:createSessionResponse>
   </soap:Body>
```

```
</soap:Envelope>
```

Step 2 Create class of service required for registering the device using a SOAP request. The supported device types are DOCSISModem, Set Top Box (STB), PacketCableMTA, eRouter, computer, CableHomeWanMan, and CableHomeWanData.

```
<v5:addClassOfService>
         <v5:context>
            <!-- This session id is the response from the create session request -->
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
         <v5:cos>
           <!-- Provide the new class of service name -->
            <v51:name>provisioned-docsis</v51:name>
<!-- supported device types are DOCSISModem, PacketCableMTA, eRouter, Computer,
CableHomeWanMan, CableHomeWanData -->
            <v51:deviceType>DOCSISModem</v51:deviceType>
<!-- The properties to be added for class of service -->
             <v51:properties>
               <v51:entry>
                 <v51:name>/cos/docsis/file</v51:name>
                 <v51:value>bronze.cm</v51:value>
               </v51:entry>
             </v51:properties>
         </v5:cos>
      </v5:addClassOfService>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addClassOfServiceResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>886f8071-c13b-4534-8023-d9d3ed2f39a4</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:16f21478:13bb65ff98b:8000000b</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addClassOfServiceResponse>
   </soap:Body>
</soap:Envelope>
```

Step 3 Create DHCP criteria required for registering the device using SOAP request.

```
<v5:addDHCPCriteria>
         <v5:context>
            <!-- This session id is the response from the create session request -->
<v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
        </v5:context>
         <v5:dhcpCriteria>
<!-- Provide the new dhcp criteria name, client class name or inclusion tag or exclusion
tag -->
            <v51:name>sample-provisioned-dhcpcriteria</v51:name>
```

```
            <v51:clientClass>ccName</v51:clientClass>
            <v51:includeSelectionTags>tagInclude</v51:includeSelectionTags>
            <v51:excludeSelectionTags>tagExclude</v51:excludeSelectionTags>
         </v5:dhcpCriteria>
      </v5:addDHCPCriteria>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addDHCPCriteriaResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>7d3c16e9-a557-4dc4-a3b0-6f0c5c787053</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:16f21478:13bb65ff98b:8000000d</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addDHCPCriteriaResponse>
   </soap:Body>
</soap:Envelope>
```

Step 4 Add group using SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.cisco.com/prime/cp/5.0">
   <soap:Header/>
   <soap:Body>
      <v5:addGroup>
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
         <v5:group>
            <v51:name>GROUP1</v51:name>
            <v51:groupType>system</v51:groupType>
         </v5:group>
      </v5:addGroup>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addGroupResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>3328e5cf-a132-4b8a-a67d-f2dca29d13ea</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:16f21478:13bb65ff98b:80000013</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
```

```
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addGroupResponse>
   </soap:Body>
</soap:Envelope>
```

Step 5 Add a custom property using SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:addCustomProperty>
         <v5:context>
            <!--Optional:-->
            <v51:sessionId>B2F21D625CE60213B13B99ECB4DD891840E4A1AF</v51:sessionId>
         </v5:context>
         <v5:customProperty>
            <!--Optional:-->
            <v51:customProperty>testcus</v51:customProperty>
            <!--Optional:-->
            <v51:dataType>INTEGER</v51:dataType>
         </v5:customProperty>
         <v5:encryptedProperty>false</v5:encryptedProperty>
      </v5:addCustomProperty>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addCustomPropertyResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:addCustomProperty>
            <cptype:operationId>e2d6bf65-ad60-402b-b6e7-d88d63fce414</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bac
-lnx-xxx/127.0.0.1:450b2bbe:166cc5fe76f:80000006</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:addCustomProperty>
      </ns2:addCustomPropertyResponse>
   </soap:Body>
</soap:Envelope>
```

Step 6 Use the device ID such as MAC address or DUID or FQDN and the device type for adding a new device to the SOAP request. You can assign a group to the new device.

The add device SOAP request also passes the subscriber's information, Class of Service, DHCP Criteria, Hostname and domain to Prime Cable Provisioning, which then registers the subscriber's device such as modem and computer.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:addDevice>
         <v5:context>
          <v51:sessionId>B58F0EE3195CAF4F82EE92F5D819EB2EC269459B</v51:sessionId>
         </v5:context>
         <v5:device>
            <!-- supported device types are DOCSISModem, PacketCableMTA, eRouter,
Computer, CableHomeWanMan,CableHomeWanData, RPD -->
            <v51:deviceType>DOCSISModem</v51:deviceType>
            <v51:deviceIds>
           <v51:macAddress type="MACAddressType">1,6,bb:1b:10:b0:10:01</v51:macAddress>
            </v51:deviceIds>
         </v5:device>
         <v5:options>
            <v51:executionOptions>
               <v51:activationMode>AUTOMATIC</v51:activationMode>
               <v51:asynchronous>true</v51:asynchronous>
            </v51:executionOptions>
            </v5:options>
      </v5:addDevice>
   </soap:Body>
</soap:Envelope>
```

SOAP response

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addDeviceResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:operationStatus>
            <cptype:operationId>9f43bacd-fc1a-40c8-8273-cbf76e018240</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>

<cptype:txId>Batch:pcp-lnx-86.cisco.com/10.65.125.111:88bf3ab:15b83c95409:800000c6</cptype
:txId>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addDeviceResponse>
   </soap:Body>
</soap:Envelope>
```

Step 7 Provision the device by connecting it to the network. Prime Cable Provisioning provides the device its registered service level.

The device is now a provisioned device with access to the appropriate level of service.

Step 8 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
<v5:closeSession>
         <v5:context>
            <!-- This session id is the response from the create session request -->
           <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </v5:closeSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:closeSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>c90793d9-4905-47b0-b6a5-37f5c59d5ae7</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
         </ns2:operationStatus>
      </ns2:closeSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Unregistering a Device

A device can be unregistered from Prime Cable Provisioning using the unregister SOAP request. This deletes the records from the Prime Cable Provisioning and subscriber gets the appropriate service level.

Desired Outcome

Unregistering a device removes the device details from Prime Cable Provisioning and gets the unprovisioned level of service.

Use this workflow to unregister a device and to bring the device online with the appropriate level of service.

Step 1 Create a connection with the respective RDU by sending a create session SOAP request. The session SOAP request must contain user and RDU details as shown below.

```
<v5:createSession>
         <v5:username>user</v5:username>
         <v5:password>password</v5:password>
         <v5:rduHost>rdu-1-lnx</v5:rduHost>
         <v5:rduPort>49187</v5:rduPort>
      </v5:createSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:context>
            <cptype:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</cptype:sessionId>
         </ns2:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Step 2 Use the device ID such as MAC address or DUID for unregistering a device. Unregister the device using the following SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:unregisterDevice>
         <v5:context>
            <v51:sessionId>4BB38F5EB0F35F3173EC14AF84ABC809C5CCBB50</v51:sessionId>
         </v5:context>
         <v5:deviceId>
            <v51:macAddress type="MACAddressType">1,6,ab:00:00:00:00:19 </v51:macAddress>
         </v5:deviceId>
      </v5:unregisterDevice>
   </soap:Body>
</soap:Envelope>
```

SOAP response

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:unregisterDeviceResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:operationStatus>
            <cptype:operationId>e492401e-5bf5-4486-8fde-4ce2e3f32b39</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>

<cptype:txId>Batch:pcp-lnx-86.cisco.com/10.65.125.111:88bf3ab:15b83c95409:800000e3</cptype
:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:unregisterDeviceResponse>
   </soap:Body>
</soap:Envelope>
```

The device is now a unregistered and unprovisioned device with access to the appropriate level of service.

Step 3 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
    <v5:closeSession>
         <v5:context>
            <!-- This session id is the response from the create session request -->
           <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </v5:closeSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
```

```
      <ns2:closeSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>5cad4a47-41c9-4b19-a4d6-8701f13884dc</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
         </ns2:operationStatus>
      </ns2:closeSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Getting DHCP Lease Information of a Device

A registered or unregistered device is assigned an IP address when it is provisioned. This IP address is used for collecting lease information such as, network register server state, and list of provisioning group in this lease.

Desired Outcome

Lists the details of the lease information from the Prime Cable Provisioning for the provisioned devices.

Use this workflow to collect the lease information of a device.

Step 1 Use an existing session or create a connection with the respective RDU by sending a create session SOAP request. The session SOAP request must contain user and RDU details as shown below.

```
    <v5:createSession>
         <v5:username>user</v5:username>
         <v5:password>password</v5:password>
         <v5:rduHost>rdu-1-lnx</v5:rduHost>
         <v5:rduPort>49187</v5:rduPort>
      </v5:createSession>
   </soap:Body>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Step 2 Use the IP address along with its provisioning group name or just the IP address of the provisioned devices in the SOAP request.

```
<v5:getDHCPLeaseInfo>
         <v5:context>
            <!--Optional:-->
            <v51:sessionId>?</v51:sessionId>
            <!--Optional:-->
            <v51:username>admin</v51:username>
            <!--Optional:-->
            <v51:password>password1</v51:password>
            <!--Optional:-->
            <v51:rduHost>bacblr-63-8-lnx</v51:rduHost>
            <!--Optional:-->
```

```
            <v51:rduPort>49187</v51:rduPort>
         </v5:context>
         <v5:ipAddress>4.0.0.3</v5:ipAddress>
         <!--Zero or more repetitions:-->
         <v5:provGroup>default</v5:provGroup>
         <!--Optional:-->
         <v5:options>
            <!--Optional:-->
            <v51:executionOptions>
               <!--Optional:-->
               <v51:activationMode>NO_ACTIVATION</v51:activationMode>
               <!--Optional:-->
               <v51:confirmationMode>NO_CONFIRMATION</v51:confirmationMode>
               <!--Optional:-->
               <v51:publishingMode>NO_PUBLISHING</v51:publishingMode>
               <!--Optional:-->
               <v51:asynchronous>false</v51:asynchronous>
               <!--Optional:-->
               <v51:reliableMode>false</v51:reliableMode>
               <!--Optional:-->
               <v51:timeout>30000</v51:timeout>
               <!--Optional:-->
               <v51:stopOnFailure>true</v51:stopOnFailure>
               <!--Optional:-->
               <v51:transactionPerItem>false</v51:transactionPerItem>
            </v51:executionOptions>
            <!--Optional:-->
            <v51:operationOptions>
               <!--Zero or more repetitions:-->
               <v51:entry>
                  <v51:name>includeleaseinfo</v51:name>
                  <!--Optional:-->
                  <v51:value>true</v51:value>
               </v51:entry>
            </v51:operationOptions>
         </v5:options>
      </v5:getDHCPLeaseInfo>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:getDHCPLeaseInfoResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:LeaseResultsOperationStatus>
            <cptype:operationStatus>

<cptype:operationId>5d438a08-aa90-47e7-8db7-24e36d0855fa</cptype:operationId>
               <cptype:code>SUCCESS</cptype:code>
               <cptype:message>Operation successful</cptype:message>
               <cptype:subStatus>
                  <cptype:status>

<cptype:txId>Batch:bac-rhel5-vm97/10.81.89.167:6186dd93:13cf1dbaef9:80000015</cptype:txId>
                     <cptype:cmdCodes>
                        <cptype:index>0</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:code>CMD_OK</cptype:code>
                     <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
                  </cptype:status>
               </cptype:subStatus>
```

```
            </cptype:operationStatus>
            <cptype:leaseResults>
               <cptype:leaseResult>
                  <cptype:device>
                     <cptype:deviceType>DOCSISModem</cptype:deviceType>
                     <cptype:deviceIds>
                        <cptype:id xsi:type="cptype:MACAddressType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">1,6,00:00:00:00:00:06</cptype:id>
                        <cptype:macAddress>1,6,00:00:00:00:00:06</cptype:macAddress>
                     </cptype:deviceIds>
                     <cptype:properties>
                        <cptype:entry>
                           <cptype:name>/generic/oidRevisionNumber</cptype:name>
                           <cptype:value>1008806346595762283-1361255488000</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/node</cptype:name>
                           <cptype:value>[]</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
                           <cptype:value>true</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/reason</cptype:name>
                           <cptype:value>NOT_REGISTERED</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
                           <cptype:value>true</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/dhcpCriteria/selected</cptype:name>
                           <cptype:value>unprovisioned-docsis</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/properties/detected</cptype:name>
                           <cptype:value xsi:nil="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/access</cptype:name>
                           <cptype:value>DEFAULT</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/network/docsisVersion</cptype:name>
                           <cptype:value>3.0</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/explanation</cptype:name>
                           <cptype:value>Because the device is not
registered</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/provGroup</cptype:name>
                           <cptype:value>default</cptype:value>
                        </cptype:entry>
                        <cptype:entry>

<cptype:name>/provisioning/classOfService/selected</cptype:name>
                           <cptype:value>unprovisioned-docsis</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>/provisioning/domain</cptype:name>
```

```
                              <cptype:value>RootDomain</cptype:value>
                           </cptype:entry>
                           <cptype:entry>
                              <cptype:name>/provisioning/isRegistered</cptype:name>
                              <cptype:value>false</cptype:value>
                           </cptype:entry>
                           <cptype:entry>
                              <cptype:name>/discoveredData/raw/dhcpv4</cptype:name>
                              <cptype:value>{REQUEST={giaddr=04:00:00:01,
chaddr=00:00:00:00:00:06,dhcp-parameter-request-list={01,03,06,07,0c,0f,33,36,04,02,43,42}
, client-id=01:00:00:00:00:00:06, relay-agent-remote-id=02:06:00:00:00:00:00:06,
relay-agent-info=01:04:80:01:03:ef:02:06:00:00:00:00:00:06:09:0b:00:00:11:8b:06:01:04:01:0
2:03:00, relay-agent-circuit-id=01:04:80:01:03:ef,
client-id-created-from-mac-address=00:00:00:00, dhcp-message-type=01, htype=01,
dhcp-class-identifier=AIC Echo,docsis3.0:, hlen=06}}</cptype:value>
                           </cptype:entry>
                           <cptype:entry>
                              <cptype:name>/provisioning/properties/selected</cptype:name>
                              <cptype:value>{/docsis/version=3.0}</cptype:value>
                           </cptype:entry>
                        </cptype:properties>
                        <cptype:discoveredData>
                           <cptype:dhcpv4RequestData>
                              <cptype:entry>
                                 <cptype:name>giaddr</cptype:name>
                                 <cptype:value>4.0.0.1</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>chaddr</cptype:name>
                                 <cptype:value>00:00:00:00:00:06</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>dhcp-parameter-request-list</cptype:name>
                                 <cptype:value>{1,3,6,7,12,15,51,54,4,2,67,66}</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>client-id</cptype:name>
                                 <cptype:value>01:00:00:00:00:00:06</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>relay-agent-remote-id</cptype:name>
                                 <cptype:value>02:06:00:00:00:00:00:06</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>relay-agent-info</cptype:name>
                                 <cptype:value>(circuit-id 1 80:01:03:ef), (remote-id 2
00:00:00:00:00:06), (v-i-vendor-opts 9 enterprise-id 4491, (cmts-capabilities 1
(docsis-version 1 03:00)))</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>relay-agent-circuit-id</cptype:name>
                                 <cptype:value>01:04:80:01:03:ef</cptype:value>
                              </cptype:entry>
                              <cptype:entry>

<cptype:name>client-id-created-from-mac-address</cptype:name>
                                 <cptype:value>0</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>dhcp-message-type</cptype:name>
                                 <cptype:value>1</cptype:value>
                              </cptype:entry>
                              <cptype:entry>
                                 <cptype:name>htype</cptype:name>
```

```
                  <cptype:value>01</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>dhcp-class-identifier</cptype:name>
                  <cptype:value>AIC Echo,docsis3.0:</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>hlen</cptype:name>
                  <cptype:value>06</cptype:value>
               </cptype:entry>
            </cptype:dhcpv4RequestData>
         </cptype:discoveredData>
         <cptype:leaseData>
            <cptype:dhcpv4LeaseQueryData>
               <cptype:entry>
                  <cptype:name>giaddr</cptype:name>
                  <cptype:value>10.106.2.58</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>dhcp-server-identifier</cptype:name>
                  <cptype:value>10.81.89.233</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>client-ipaddress</cptype:name>
                  <cptype:value>4.0.0.3</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>dhcp-lease-time</cptype:name>
                  <cptype:value>603024</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>client-mac-address</cptype:name>
                  <cptype:value>1,6,00:00:00:00:00:06</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>relay-agent-remote-id</cptype:name>
                  <cptype:value>00:00:00:00:00:06</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>client-id</cptype:name>
                  <cptype:value>01:00:00:00:00:00:06</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>relay-agent-circuit-id</cptype:name>
                  <cptype:value>80:01:03:ef</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>routers</cptype:name>
                  <cptype:value>4.0.0.1</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>client-last-transaction-time</cptype:name>
                  <cptype:value>1776</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>subnet-mask</cptype:name>
                  <cptype:value>255.255.255.0</cptype:value>
               </cptype:entry>
            </cptype:dhcpv4LeaseQueryData>
         </cptype:leaseData>
      </cptype:device>
      <cptype:isActive>true</cptype:isActive>
      <cptype:isPartialAnswer>false</cptype:isPartialAnswer>
      <cptype:isV4>true</cptype:isV4>
```

```
                    <cptype:isV6>false</cptype:isV6>
                 </cptype:leaseResult>
              </cptype:leaseResults>
           </ns2:LeaseResultsOperationStatus>
        </ns2:getDHCPLeaseInfoResponse>
     </soap:Body>
</soap:Envelope>
```

Step 3 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
<v5:closeSession>
         <v5:context>
            <!-- This session id is the response from the create session request -->
           <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </v5:closeSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Updating Device Details

You can update the device's class of service, DHCP criteria, assign or unassign a group, and change or add properties. You can update any or all of this information as part of single updateDevice operation.

Embedded devices are not supported in Prime Cable Provisioning. The following workflow updates a single device. Before updating the device a new class of service and new dhcp criteria are added.

Desired Outcome

Prime Cable Provisioning regenerates the configuration file after updating the device with the required changes. The device then gets re-provisioned with updated level of service.

Use this workflow to update a device's details and provision it with the updated level of service.

Step 1 Create a connection with the respective RDU by sending a session SOAP request. The session SOAP request must contain user and RDU details as shown below.

```
    <v5:createSession>
         <v5:username>user</v5:username>
         <v5:password>password</v5:password>
         <v5:rduHost>rdu-1-lnx</v5:rduHost>
         <v5:rduPort>49187</v5:rduPort>
      </v5:createSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Step 2 Add a new class of service for the device using the following SOAP request:

```
<v5:addClassOfService>
         <v5:context>
            <!-- This session id is the response from the create session request -->
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
         <v5:cos>
           <!-- Provide the new class of service name -->
            <v51:name>updated-provisioned-docsis</v51:name>
<!-- supported device types are DOCSISModem, PacketCableMTA, eRouter, Computer,
CableHomeWanMan, CableHomeWanData -->
            <v51:deviceType>DOCSISModem</v51:deviceType>
<!-- The must property to be added for class of service  -->
             <v51:properties>
               <v51:entry>
                 <v51:name>/cos/docsis/file</v51:name>
                 <v51:value>silver.cm</v51:value>
               </v51:entry>
       <v51:entry>
                 <v51:name>/IPDevice/mustBeInProvGroup</v51:name>
                 <v51:value>south-california</v51:value>
               </v51:entry>
             </v51:properties>
         </v5:cos>
      </v5:addClassOfService>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addClassOfServiceResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>d5330e77-aa3f-491a-9405-22bad58fe16b</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:33d869b2:13bbb9ced4e:80000015</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addClassOfServiceResponse>
   </soap:Body>
```

```
</soap:Envelope>
```

Step 3 Add a new DHCP criteria for the device using the following SOAP request:

```
<v5:addDHCPCriteria>
         <v5:context>
            <!-- This session id is the response from the create session request -->
<v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
</v5:context>
         <v5:dhcpCriteria>
<!-- Provide the new dhcp criteria name, client class name or inclusion tag or exclusion
tag  -->
            <v51:name>updated-provisioned-dhcpcriteria</v51:name>
            <v51:clientClass>ccNameUpdated</v51:clientClass>
            <v51:includeSelectionTags>tagInclude</v51:includeSelectionTags>
            <v51:excludeSelectionTags>tagExclude</v51:excludeSelectionTags>
          </v5:dhcpCriteria>
      </v5:addDHCPCriteria>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addDHCPCriteriaResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>f8a46a35-fb58-4117-9483-38776c8f4480</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:33d869b2:13bbb9ced4e:80000017</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addDHCPCriteriaResponse>
   </soap:Body>
</soap:Envelope>
```

Step 4 If you want to assign a group that is not yet created, create it using the following SOAP request.

```
<v5:addGroup>
         <v5:context>
            <!-- This session id is the response from the create session request -->
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>

         </v5:context>
         <v5:group>
<!-- The new group name and group type it belongs to; Group type should exists in the BAC
server; system group type is the default group exists in BAC server  -->
            <v51:name>GROUP2</v51:name>
            <v51:groupType>system</v51:groupType>
         </v5:group>
    </v5:addGroup>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
```

```
      <ns2:addGroupResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>abb116d0-0963-40a3-9a2a-514760dca9e9</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:33d869b2:13bbb9ced4e:80000019</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addGroupResponse>
   </soap:Body>
</soap:Envelope>
```

Step 5 Use the device ID such as MAC address or DUID or FQDN and the device type to update the device details. You can update the subscriber's details as well as assign or unassign the device to a group by using the following SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:updateDevice>
         <v5:context>
            <v51:sessionId>D9D97B4D2516F484668CAEC4217B445E32ED208A</v51:sessionId>
         </v5:context>
         <v5:deviceId>
            <v51:macAddress type="MACAddressType">1,6,ab:00:00:00:00:01</v51:macAddress>
         </v5:deviceId>
         <v5:device>
           <v51:deviceType>DOCSISModem</v51:deviceType>
           <v51:cos>sample-gold-docsis</v51:cos>
         </v5:device>
      </v5:updateDevice>
   </soap:Body>
</soap:Envelope>
```

The device gets a regenerated configuration and is provisioned with updated service level.

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:updateDeviceResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:operationStatus>
            <cptype:operationId>55a9bc61-5f61-4ac0-b8c4-5f15a6e41119</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>

<cptype:txId>Batch:pcp-lnx-86.cisco.com/10.65.125.111:88bf3ab:15b83c95409:800000ff</cptype
:txId>
                  <cptype:cmdCodes>
```

```
                        <cptype:index>0</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>1</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:code>BATCH_COMPLETED</cptype:code>
                     <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
                  </cptype:status>
               </cptype:subStatus>
            </ns2:operationStatus>
         </ns2:updateDeviceResponse>
   </soap:Body>
</soap:Envelope>
```

Step 6 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
<v5:closeSession>
         <v5:context>
            <!-- This session id is the response from the create session request -->
           <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </v5:closeSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Searching a Device

The subscriber has devices such as DOCSIS, MTA, DPoE and computer in Prime Cable Provisioning. These devices can be searched from the Prime Cable Provisioning and get the results as a list in SOAP response.

Desired Outcome

Searches device(s) from Prime Cable Provisioning based on specific search criteria.

Use this workflow to search a device and get the list that is generated based on the search criteria.

Step 1 Create a connection with the respective RDU by sending a session SOAP request. The session SOAP request must contain user and RDU details as shown below.

```
<soap:Body>
 <v5:createSession>
```

```
   <v5:username>user</v5:username>
   <v5:password>password</v5:password>
   <v5:rduHost>rdu-1-lnx</v5:rduHost>
   <v5:rduPort>49187</v5:rduPort>
 </v5:createSession>
</soap:Body>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Step 2 Use the query to create a search criteria.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <soap:Header/>
   <soap:Body>
      <v5:search>
         <v5:context>
            <v51:sessionId>9CBCBD32F6437D346C4C068C7AB6832E32154407</v51:sessionId>
         </v5:context>
         <v5:search>
        <v51:query xsi:type="v51:DeviceSearchByDeviceIdPatternType">
        <v51:deviceIdPattern>
        <v51:macAddressPattern>*</v51:macAddressPattern>
        </v51:deviceIdPattern>
        <v51:returnParameters>BASIC</v51:returnParameters>
        </v51:query>
        <v51:start>1,6,aa:00:00:00:00:01</v51:start>
          <v51:maxResults>1</v51:maxResults>
        </v5:search>
      </v5:search>
   </soap:Body>
</soap:Envelope>
```

Note For the first query, the start element must not be included.

For all the supported search requests Search.query and Search.maxResults elements are required. Search.start is used to support paging across large search results. The first search operation must not include Search.start element. Its absence indicates that a new search is being initiated. A response to a search operation contains a SearchResult. SearchResult contains a *next* element which itself is a Search object. The entire content of the next element (query, start, size) should be used to page to the next batch of search results. The value of the start element is used by RDU's internal paging mechanism. Hence, from second request onwards, Search.start is required to page over the results. At the end of the search process, an empty SearchResult where SearchResult.size equals zero will be returned.

Step 3 Ensure that the response contains the next query request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
```

```
   <soap:Body>
      <ns2:searchResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:results>
   <cptype:item xsi:type="cptype:DeviceType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"   ">
           <cptype:deviceType>Computer</cptype:deviceType>
            <cptype:deviceIds>
                <cptype:macAddress>1,6,00:00:ca:be:52:49</cptype:macAddress>
            </cptype:deviceIds>
         <!-- repeat the list of item elements based on the available device ids in Cisco
BAC -->
          ---
          ---
          ---
         <cptype:size>500</cptype:size>
         <cptype:next>
<cptype:query xsi:type="ns4:DeviceSearchByDeviceIdPatternType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ns4="http://www.cisco.com/prime/xsd/cp/5.0">
                  <cptype:deviceIdPattern>
                     <cptype:macAddressPattern>*</cptype:macAddressPattern>
                  </cptype:deviceIdPattern>
                  <cptype:returnParameters>BASIC</cptype:returnParameters>
       </cptype:query>
       <cptype:start>1,6,05:00:00:00:03:87</cptype:start>
       <cptype:maxResults>500</cptype:maxResults>
       </cptype:next>
     </ns2:results>
</ns2:searchResponse>
</soap:Envelope>
```

Step 4 Use the next element contents for the next search query criteria.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.cisco.com/prime/cp/5.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xns="http://www.cisco.com/prime/xsd/cp/5.0">
   <soap:Header/>
   <soap:Body>
      <v5:search>
         <v5:context>
            <!-- This session id is the response from the create session request -->
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
         <v5:search>
            <v51:query xsi:type="v51:DeviceSearchByDeviceIdPatternType"
                  <v51:deviceIdPattern>
                     <v51:macAddressPattern>*</v51:macAddressPattern>
                  </v51:deviceIdPattern>
                  <v51:returnParameters>BASIC</v51:returnParameters>
       </v51:query>
       <v51:start>1,6,05:00:00:00:03:87</v51:start>
       <v51:maxResults>500</v51:maxResults>
         </v5:search>
      </v5:search>
   </soap:Body>
</soap:Envelope>
```

Step 5 Repeat step 4 and step 5 till you receive the search response size as zero.

```
<ns2:searchResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
                  <ns2:results>
                  <!-- Make sure that the size tag contains 0 and there is no next tag -->
```

```
                                    <cptype:size>0</cptype:size>
                </ns2:results>
</ns2:searchResponse>
```

All the devices are available based on the search request.

Step 6 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
<v5:closeSession>
<v5:context>
<!-- This session id is the response from the create session request -->
<v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
</v5:context>
</v5:closeSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:closeSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>bd013d5e-ba16-4687-bfdb-440d1ed960ec</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
         </ns2:operationStatus>
      </ns2:closeSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Note You can also run search query based on different object types such as, device, Class of Service, DHCP Criteria, file and group. A sample search query pattern with these object types is shown below:

Supported Query Elements

The supported search elements in Prime Cable Provisioning 6.1.3 are:

Searching basic device details of a registered device with class of service as *sample-gold-docsis*:

```
    <v51:query xsi:type="v51:DeviceSearchByCOSType">
            <v51:classOfService>sample-gold-docsis</v51:classOfService>
            <v51:associationType>REGISTERED</v51:associationType>
            <v51:returnParameters>BASIC</v51:returnParameters>
    </v51:query>
                <v51:maxResults>5000</v51:maxResults>
```

Searching basic device details of a registered device with DHCP criteria as *unprovisioned-docsis*:

```
<v51:query xsi:type="v51:DeviceSearchByDHCPCriteriaType">
        <v51:dhcpCriteria> unprovisioned-docsis </v51:dhcpCriteria>
        <v51:associationType>REGISTERED</v51:associationType>
        <v51:returnParameters>BASIC</v51:returnParameters>
 </v51:query>
             <v51:maxResults>5000</v51:maxResults>
```

Searching basic device details of a registered device with default CoS device type as *DOCSISModem:*

```
<v51:query xsi:type="v51:DeviceSearchByDefaultCOSType">
        <v51:deviceType>DOCSISModem</v51:deviceType>
        <v51:associationType>REGISTERED</v51:associationType>
        <v51:returnParameters>BASIC</v51:returnParameters>
       </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Searching basic device details of a registered device with default DHCP criteria device type as *DOCSISModem:*

```
<v51:query xsi:type="v51:DeviceSearchByDefaultDHCPCriteriaType">
        <v51:deviceType>DOCSISModem</v51:deviceType>
        <v51:associationType>REGISTERED</v51:associationType>
        <v51:returnParameters>BASIC</v51:returnParameters>
       </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Searching all the device details of a device with device ID type as *macAddress*:

```
<v51:query xsi:type="v51:DeviceSearchByDeviceIdPatternType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/5.0">
                  <v51:deviceIdPattern>
                     <v51:macAddressPattern>*</v51:macAddressPattern>
                  </v51:deviceIdPattern>
                  <v51:returnParameters>ALL</v51:returnParameters>
       </v51:query>        <v51:maxResults>500</v51:maxResults
```

Searching all the device details of a device with device type as *DocsisModem:*

```
<v51:query xsi:type="v51:DeviceSearchByDeviceTypeType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/5.0">
                  <v51:deviceType>DOCSISModem</v51:deviceType>
                  <v51:returnParameters>ALL</v51:returnParameters>
       </v51:query> <v51:maxResults>500</v51:maxResults>
```

Searching all the device details of a device with group name as *testGroup*:

```
<v51:query xsi:type="v51:DeviceSearchByProvGroupNameType">
        <v51:provGroupName>testGroup</v51:provGroupName>
        <v51:returnParameters>ALL</v51:returnParameters>
        </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Search the entire device by ownerId as testOwner:

```
<v51:query xsi:type="v51:DeviceSearchByOwnerIdType">
        <v51:ownerId>testOwner</v51:ownerId>
        </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Searching all class of service with device type as DOCSISModem:

```
<v51:query xsi:type="v51:CosSearchByDeviceTypeType">
        <v51:deviceType>DOCSISModem</v51:deviceType>
        </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Searching all DHCP criteria:

```
<v51:query xsi:type="v51:DHCPCriteriaSearchType">
        </v51:query>
```

```
            <v51:maxResults>5000</v51:maxResults>
```

Searching all files with file type as *CABLELABS_CONFIGURATION_TEMPLATE*:

```
<v51:query xsi:type="v51:FileSearchByFileTypeType">
        <v51:fileType>CABLELABS_CONFIGURATION_TEMPLATE</v51:fileType>
        </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Searching all files with a file name pattern:

```
<v51:query xsi:type="v51:FileSearchByFileNamePatternType">
        <v51:fileNamePattern>*</v51:fileNamePattern>
        </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Searching all groups with group type as *system*:

```
<v51:query xsi:type="v51:GroupSearchByGroupTypeType">
        <v51:groupType>system</v51:groupType>
        </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Searching all groups with a group name pattern:

```
<v51:query xsi:type="v51:GroupSearchByGroupNamePatternType">
        <v51:groupNamePattern>*</v51:groupNamePattern>
        </v51:query>
            <v51:maxResults>5000</v51:maxResults>
```

Deleting a Device

The existing devices in Prime Cable Provisioning can be deleted using the delete SOAP request.

Desired Outcome

Deleting a device removes the device details from the Prime Cable Provisioning.

Use this workflow to delete a device.

Step 1 Create a connection with the respective RDU by sending a session SOAP request. The session SOAP request must contain user and RDU details as shown below.

```
<soap:Body>
      <v5:createSession>
         <v5:username>user</v5:username>
         <v5:password>password</v5:password>
         <v5:rduHost>rdu-1-lnx</v5:rduHost>
         <v5:rduPort>49187</v5:rduPort>
      </v5:createSession>
   </soap:Body>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
```

```
   </soap:Body>
</soap:Envelope>
```

Use the device ID such as MAC address or DUID to delete the device.

Step 2 Delete the device using the following SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:deleteDevice>
         <v5:context>
            <v51:sessionId>F0C22BB822814E85757A80F1B11928C918085914</v51:sessionId>
         </v5:context>
         <v5:deviceId>
           <v51:macAddress type="MACAddressType">1,6,ab:00:10:00:10:03</v51:macAddress>
         </v5:deviceId>
      </v5:deleteDevice>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:deleteDeviceResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:operationStatus>
            <cptype:operationId>6618a53f-6487-4e93-a23a-04f0ea22664d</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>

<cptype:txId>Batch:pcp-lnx-86.cisco.com/10.65.125.111:88bf3ab:15b83c95409:80000110</cptype
:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:deleteDeviceResponse>
   </soap:Body>
</soap:Envelope>
```

Step 3 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
<v5:closeSession>
<v5:context>
<!-- This session id is the response from the create session request -->
<v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
</v5:context>
</v5:closeSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:closeSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>bd013d5e-ba16-4687-bfdb-440d1ed960ec</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
         </ns2:operationStatus>
      </ns2:closeSessionResponse>
   </soap:Body>
</soap:Envelope>
```

The device record is deleted from Prime Cable Provisioning.

Multiple Devices Operations in a Single Request

Multiple devices in Prime Cable Provisioning can be added, retrieved, updated, or deleted using a single SOAP request.

Desired Outcome

Use this workflow to add, retreive, update, and delete multiple devices.

Step 1 Create a connection with the respective RDU by sending a session SOAP request. The session SOAP request must contain user and RDU details as shown below.

```
<soap:Body>
      <v5:createSession>
         <v5:username>user</v5:username>
         <v5:password>password</v5:password>
         <v5:rduHost>rdu-1-lnx</v5:rduHost>
         <v5:rduPort>49187</v5:rduPort>
      </v5:createSession>
   </soap:Body>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Step 2 Add five devices using a single SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:addDevices>
         <v5:context>
          <v51:sessionId>4BC1949E2101D24DEAF7BEBFE195F33132571B3A</v51:sessionId>
```

```
         </v5:context>
         <v5:devices>
            <v51:id>?</v51:id>
            <!-- supported device types are DOCSISModem, PacketCableMTA, eRouter,
Computer, CableHomeWanMan, CableHomeWanData, RPD -->
            <v51:deviceType>DOCSISModem</v51:deviceType>
            <v51:deviceIds>
           <v51:macAddress type="MACAddressType">1,6,ab:00:10:00:10:03</v51:macAddress>
            </v51:deviceIds>
         </v5:devices>
        <v5:devices>
            <v51:id>?</v51:id>
            <!-- supported device types are DOCSISModem, PacketCableMTA, eRouter,
Computer, CableHomeWanMan, CableHomeWanData, RPD -->
            <v51:deviceType>Computer</v51:deviceType>
            <v51:deviceIds>
           <v51:macAddress type="MACAddressType">1,6,ab:b0:10:00:10:03</v51:macAddress>
            </v51:deviceIds>
         </v5:devices>
      </v5:addDevices>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:addDevicesResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:operationStatus>
            <cptype:operationId>865170e8-c14a-45ff-9948-71d952fd5cfa</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>

<cptype:txId>Batch:pcp-lnx-86.cisco.com/10.65.125.111:88bf3ab:15b83c95409:800000ea</cptype
:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:cmdCodes>
                     <cptype:index>1</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>BATCH_COMPLETED</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:addDevicesResponse>
   </soap:Body>
</soap:Envelope>
```

Step 3 Get five devices using a single SOAP request

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.cisco.com/prime/cp/5.0">
   <soap:Header/>
   <soap:Body>
      <v5:getDevices>
         <v5:context>
            <v51:sessionId>${#TestSuite#SessionID}</v51:sessionId>
```

```
         </v5:context>
         <!--1st device -->
          <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:00</v51:id>
          </v5:deviceIds>
         <!--2nd device-->
         <v51:deviceIds>
               <v51:id xsi:type="v51:DUIDType">00:03:03:10:00:00:00:00:01</v51:id>
         </v51:deviceIds>
         <!--3rd device -->
         <v5:deviceIds>
               <!--<v51:id xsi:type="v51:FQDNType">${#TestSuite#deviceFQDN}</v51:id>-->
               <v51:id xsi:type="v51:DUIDType">00:03:03:10:00:00:00:00:02</v51:id>
         </v5:deviceIds>
         <!--4th device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:03</v51:id>
         </v5:deviceIds>
         <!--5th device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:04</v51:id>
         </v5:deviceIds>
      </v5:getDevices>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:getDevicesResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:deviceOperationStatus>
            <cptype:operationStatus>

<cptype:operationId>81d3849f-9eb2-4c4b-965d-9de13d3b424d</cptype:operationId>
               <cptype:code>SUCCESS</cptype:code>
               <cptype:message>Operation successful</cptype:message>
               <cptype:subStatus>
                  <cptype:status>
                     <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:16f21478:13bb65ff98b:80000028</cptype:txId>
                     <cptype:cmdCodes>
                        <cptype:index>0</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>1</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>2</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>3</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>4</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:code>BATCH_COMPLETED</cptype:code>
                     <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
                  </cptype:status>
```

```
      </cptype:subStatus>
   </cptype:operationStatus>
   <v5:devices>
      <cptype:deviceType>DOCSISModem</deviceType>
      <cptype:deviceIds>
         <cptype:macAddress>1,6,10:00:00:00:00:00</cptype:macAddress>
         <cptype:duid>00:03:03:10:00:00:00:00:00</cptype:duid>
         <cptype:fqdn>1-6-10-00-00-00-00-00.cisco.com</cptype:fqdn>
      </cptype:deviceIds>
      <cptype:subscriberId>Subscriber1</cptype:subscriberId>
      <cptype:cos>unprovisioned-docsiscptype:cos>
      <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
      <cptype:hostName>1-6-10-00-00-00-00-00</cptype:hostName>
      <cptype:domainName>cisco.com</cptype:domainName>
      <cptype:properties>
         cptype:entry>
            <cptype:name>/generic/oidRevisionNumber</cptype:name>
            <cptype:value>1008806320825958501-1355978642001</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
            <cptype:value>pg1</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/domain</cptype:name>
            <cptype:value>RootDomain</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/node</cptype:name>
            <cptype:value>[]</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
            <cptype:value>false</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/isRegistered</cptype:name>
            <cptype:value>true</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
            <cptype:value>false</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/properties/detected</cptype:name>
            <cptype:value/>
         </cptype:entry>
      </cptype:properties>
   </v5:devices>
   <v5:devices>
      <cptype:deviceType>CableHomeWanMan</cptype:deviceType>
      <cptype:deviceIds>
         <cptype:duid>00:03:03:10:00:00:00:00:01</cptype:duid>
         <cptype:fqdn>00-03-03-10-00-00-00-00-01.cisco.com</cptype:fqdn>
      </cptype:deviceIds>
      <cptype:subscriberId>Subscriber1</cptype:subscriberId>
      <cptype:cos>unprovisioned-cablehome-wan-man</cptype:cos>
      <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
      <cptype:hostName>00-03-03-10-00-00-00-00-01</cptype:hostName>
      <cptype:domainName>cisco.com</cptype:domainName>
      <cptype:properties>
         <cptype:entry>
            <cptype:name>/generic/oidRevisionNumber</cptype:name>
            <cptype:value>1008806320825958502-1355978642001</cptype:value>
```

```
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
               <cptype:value>pg1</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/domain</cptype:name>
               <cptype:value>RootDomain</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/node</cptype:name>
               <cptype:value>[]</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isRegistered</cptype:name>
               <cptype:value>true</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/properties/detected</cptype:name>
               <cptype:value/>
            </cptype:entry>
         </cptype:properties>
      </v5:devices>
      <v5:devices>
         <cptype:deviceType>CableHomeWanData</cptype:deviceType>
         <cptype:deviceIds>
            <cptype:duid>00:03:03:10:00:00:00:00:02</cptype:duid>
         </cptype:deviceIds>
         <cptype:subscriberId>Subscriber1</cptype:subscriberId>
         <cptype:cos>unprovisioned-cablehome-wan-data</cptype:cos>
         <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
         <cptype:properties>
            <cptype:entry>
               <cptype:name>/generic/oidRevisionNumber</cptype:name>
               <cptype:value>1008806320825958503-1355978642001</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
               <cptype:value>pg1</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/domain</cptype:name>
               <cptype:value>RootDomain</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/node</cptype:name>
               <cptype:value>[]</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isRegistered</cptype:name>
               <cptype:value>true</cptype:value>
            </cptype:entry>
```

```
            <cptype:entry>
               <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/properties/detected</cptype:name>
               <cptype:value/>
            </cptype:entry>
         </cptype:properties>
      </v5:devices>
      <v5:devices>
         <cptype:deviceType>PacketCableMTA</cptype:deviceType>
         <cptype:deviceIds>
            <cptype:macAddress>1,6,10:00:00:00:00:03</cptype:macAddress>
            <cptype:fqdn>1-6-10-00-00-00-00-03.cisco.com</cptype:fqdn>
         </cptype:deviceIds>
         <cptype:subscriberId>Subscriber1</cptype:subscriberId>
         <cptype:cos>unprovisioned-packet-cable-mta</cptype:cos>
         <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
         <cptype:hostName>1-6-10-00-00-00-00-03</cptype:hostName>
         <cptype:domainName>cisco.com</cptype:domainName>
         <cptype:properties>
            <cptype:entry>
               <cptype:name>/generic/oidRevisionNumber</cptype:name>
               <cptype:value>1008806320825958504-1355978642001</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
               <cptype:value>pg1</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/domain</cptype:name>
               <cptype:value>RootDomain</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/node</cptype:name>
               <cptype:value>[]</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isRegistered</cptype:name>
               <cptype:value>true</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/properties/detected</cptype:name>
               <cptype:value/>
            </cptype:entry>
         </cptype:properties>
      </v5:devices>
      <v5:devices>
         <cptype:deviceType>Computer</cptype:deviceType>
         <cptype:deviceIds>
            <cptype:macAddress>1,6,10:00:00:00:00:04</cptype:macAddress>
         </cptype:deviceIds>
         <cptype:subscriberId>Subscriber1</cptype:subscriberId>
         <cptype:cos>unprovisioned-computer</cptype:cos>
         <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
```

```
                  <cptype:properties>
                     <cptype:entry>
                        <cptype:name>/generic/oidRevisionNumber</cptype:name>
                        <cptype:value>1008806320825958505-1355978642001</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
                        <cptype:value>pg1</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>/provisioning/domain</cptype:name>
                        <cptype:value>RootDomain</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>/node</cptype:name>
                        <cptype:value>[]</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
                        <cptype:value>false</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>/provisioning/isRegistered</cptype:name>
                        <cptype:value>true</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
                        <cptype:value>false</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>/provisioning/properties/detected</cptype:name>
                        <cptype:value/>
                     </cptype:entry>
                  </cptype:properties>
               </cptype:devices>
            </ns2:deviceOperationStatus>
         </ns2:getDevicesResponse>
      </soap:Body>
   </soap:Envelope>
```

Step 4 Update two devices using a single SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:updateDevices>
         <v5:context>
            <v51:sessionId>D9D97B4D2516F484668CAEC4217B445E32ED208A</v51:sessionId>
         </v5:context>
         <v5:deviceIds>
            <v51:macAddress type="MACAddressType">1,6,ab:00:00:00:00:01</v51:macAddress>
         </v5:deviceIds>
         <v5:device>
            <v51:cos>sample-gold-docsis</v51:cos>
         </v5:device>
      </v5:updateDevices>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
```

```
   <soap:Body>
      <ns2:updateDevicesResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:operationStatus>
            <cptype:operationId>ec87f9f3-e053-4976-afe3-fc60ae93e95b</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>

<cptype:txId>Batch:pcp-lnx-86.cisco.com/10.65.125.111:88bf3ab:15b83c95409:80000101</cptype
:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:cmdCodes>
                     <cptype:index>1</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>BATCH_COMPLETED</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:updateDevicesResponse>
   </soap:Body>
</soap:Envelope>
```

Step 5 Get updated five devices details using a single SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.cisco.com/prime/cp/5.0">
   <soap:Header/>
   <soap:Body>
      <v5:getDevices>
         <v5:context>
            <v51:sessionId>${#TestSuite#SessionID}</v51:sessionId>
         </v5:context>
         <!--1st device -->
          <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:00</v51:id>
          </v5:deviceIds>
         <!--2nd device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:DUIDType">00:03:03:10:00:00:00:00:01</v51:id>
         </v5:deviceIds>
         <!--3rd device -->
         <v5:deviceIds>
               <!--<fqdn>${#TestSuite#deviceFQDN}</fqdn>-->
               <v51:id xsi:type="v51:DUIDType">00:03:03:10:00:00:00:00:02</v51:id>
         </v5:deviceIds>
         <!--4th device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:03</v51:id>
         </v5:deviceIds>
         <!--5th device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:04</v51:id>
         </v51:deviceIds>
      </v5:getDevices>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:getDevicesResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:deviceOperationStatus>
            <ns2:operationStatus>

<cptype:operationId>fc81ac5e-a5f5-4066-a6af-4f8e40b6e90a</cptype:operationId>
               <cptype:code>SUCCESS</cptype:code>
               <cptype:message>Operation successful</cptype:message>
               <cptype:subStatus>
                  <cptype:status>
                     <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:16f21478:13bb65ff98b:8000002c</cptype:txId>
                     <cptype:cmdCodes>
                        <cptype:index>0</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>1</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>2</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>3</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>4</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:code>BATCH_COMPLETED</cptype:code>
                     <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
                  </cptype:status>
               </cptype:subStatus>
            </ns2:operationStatus>
            <v5:devices>
               <cptype:deviceType>DOCSISModem</deviceType>
               <cptype:deviceIds>
                  <cptype:macAddress>1,6,10:00:00:00:00:00</cptype:macAddress>
                  <cptype:duid>00:03:03:10:00:00:00:00:00</cptype:duid>
                  <cptype:fqdn>1-6-10-00-00-00-00-00.cisco.com</cptype:fqdn>
               </cptype:deviceIds>
               <cptype:subscriberId>Subscriber2</cptype:subscriberId>
               <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
               <cptype:hostName>1-6-10-00-00-00-00-00</cptype:hostName>
               <cptype:domainName>cisco.com</cptype:domainName>
               <cptype:properties>
                  <cptype:entry>
                     <cptype:name>/generic/oidRevisionNumber</cptype:name>
                     <cptype:value>1008806320825958501-1355978831000</cptype:value>
                  </cptype:entry>
                  <cptype:entry>
                     <cptype:name>/provisioning/domain</cptype:name>
                     <cptype:value>RootDomain</cptype:value>
                  </cptype:entry>
                  <cptype:entry>
                     <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
                     <cptype:value>false</cptype:value>
                  </cptype:entry>
```

```
            <cptype:entry>
               <cptype:name>/node</cptype:name>
               <cptype:value>[system-diagnostics]</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isRegistered</cptype:name>
               <cptype:value>true</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/properties/detected</cptype:name>
               <cptype:value/>
            </cptype:entry>
         </cptype:properties>
      </v5:devices>
      <v5:devices>
         <cptype:deviceType>CableHomeWanMan</cptype:deviceType>
         <cptype:deviceIds>
            <cptype:duid>00:03:03:10:00:00:00:00:01</cptype:duid>
            <cptype:fqdn>00-03-03-10-00-00-00-00-01.cisco.com</cptype:fqdn>
         </cptype:deviceIds>
         <cptype:subscriberId>Subscriber2</cptype:subscriberId>
         <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
         <cptype:hostName>00-03-03-10-00-00-00-00-01</cptype:hostName>
         <cptype:domainName>cisco.com</cptype:domainName>
         <cptype:properties>
            <cptype:entry>
               <cptype:name>/generic/oidRevisionNumber</cptype:name>
               <cptype:value>1008806320825958502-1355978831000</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/domain</cptype:name>
               <cptype:value>RootDomain</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/node</cptype:name>
               <cptype:value>[system-diagnostics]</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isRegistered</cptype:name>
               <cptype:value>true</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
               <cptype:value>false</cptype:value>
            </cptype:entry>
            <cptype:entry>
               <cptype:name>/provisioning/properties/detected</cptype:name>
               <cptype:value/>
            </cptype:entry>
         </cptype:properties>
      </v5:devices>
      <v5:devices>
         <cptype:deviceType>CableHomeWanData</cptype:deviceType>
         <cptype:deviceIds>
            <cptype:duid>00:03:03:10:00:00:00:00:02</cptype:duid>
         </cptype:deviceIds>
```

```
      <cptype:subscriberId>Subscriber1</cptype:subscriberId>
      <cptype:cos>unprovisioned-cablehome-wan-data</cptype:cos>
      <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
      <cptype:properties>
         <cptype:entry>
            <cptype:name>/generic/oidRevisionNumber</cptype:name>
            <cptype:value>1008806320825958503-1355978642001</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
            <cptype:value>pg1</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/domain</cptype:name>
            <cptype:value>RootDomain</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/node</cptype:name>
            <cptype:value>[]</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
            <cptype:value>false</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/isRegistered</cptype:name>
            <cptype:value>true</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
            <cptype:value>false</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/properties/detected</cptype:name>
            <cptype:value/>
         </cptype:entry>
      </cptype:properties>
   </v5:devices>
   <v5:devices>
      <cptype:deviceType>PacketCableMTA</cptype:deviceType>
      <cptype:deviceIds>
         <cptype:macAddress>1,6,10:00:00:00:00:03</cptype:macAddress>
         <cptype:fqdn>1-6-10-00-00-00-00-03.cisco.com</cptype:fqdn>
      </cptype:deviceIds>
      <cptype:subscriberId>Subscriber1</cptype:subscriberId>
      <cptype:cos>unprovisioned-packet-cable-mta</cptype:cos>
      <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
      <cptype:hostName>1-6-10-00-00-00-00-03</cptype:hostName>
      <cptype:domainName>cisco.com</cptype:domainName>
      <cptype:properties>
         <cptype:entry>
            <cptype:name>/generic/oidRevisionNumber</cptype:name>
            <cptype:value>1008806320825958504-1355978642001</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
            <cptype:value>pg1</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/domain</cptype:name>
            <cptype:value>RootDomain</cptype:value>
         </cptype:entry>
         <cptype:entry>
            <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
```

```
                  <cptype:value>false</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/node</cptype:name>
                  <cptype:value>[]</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/isRegistered</cptype:name>
                  <cptype:value>true</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
                  <cptype:value>false</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/properties/detected</cptype:name>
                  <cptype:value/>
               </cptype:entry>
            </cptype:properties>
         </v5:devices>
         <v5:devices>
            <cptype:deviceType>Computer</cptype:deviceType>
            <cptype:deviceIds>
               <cptype:macAddress>1,6,10:00:00:00:00:04</cptype:macAddress>
            </cptype:deviceIds>
            <cptype:subscriberId>Subscriber1</cptype:subscriberId>
            <cptype:cos>unprovisioned-computer</cptype:cos>
            <cptype:dhcpCriteria>sample-provisioned</cptype:dhcpCriteria>
            <cptype:properties>
               <cptype:entry>
                  <cptype:name>/generic/oidRevisionNumber</cptype:name>
                  <cptype:value>1008806320825958505-1355978642001</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/IPDevice/mustBeInProvGroup</cptype:name>
                  <cptype:value>pg1</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/domain</cptype:name>
                  <cptype:value>RootDomain</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/node</cptype:name>
                  <cptype:value>[]</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/isBehindRequiredDevice</cptype:name>
                  <cptype:value>false</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/isRegistered</cptype:name>
                  <cptype:value>true</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/isInRequiredProvGroup</cptype:name>
                  <cptype:value>false</cptype:value>
               </cptype:entry>
               <cptype:entry>
                  <cptype:name>/provisioning/properties/detected</cptype:name>
                  <cptype:value/>
               </cptype:entry>
            </cptype:properties>
         </cptype:devices>
      </ns2:deviceOperationStatus>
```

```
      </ns2:getDevicesResponse>
   </soap:Body>
</soap:Envelope>
```

Step 6 Get five devices details with lease query information using a single SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:getDevices>
         <v5:context>
            <!--Optional:-->
            <v51:sessionId>?</v51:sessionId>
            <!--Optional:-->
             <v51:username>admin</v51:username>
            <!--Optional:-->
            <v51:password>password1</v51:password>
            <!--Optional:-->
            <v51:rduHost>bacblr-63-8-lnx</v51:rduHost>
            <!--Optional:-->
            <v51:rduPort>49187</v51:rduPort>
         </v5:context>
         <!--1 or more repetitions:-->
         <v5:deviceIds>
          <v51:id xsi:type="v51:MACAddressType"> 1,6,00:00:00:00:00:06</v51:id>
         </v5:deviceIds>
         <v5:deviceIds>
           <v51:id xsi:type="v51:MACAddressType"> 1,6,00:00:00:00:00:07</v51:id>
         </v5:deviceIds>
         <v5:deviceIds>
         <v51:id xsi:type="v51:MACAddressType"> 1,6,00:00:00:00:00:08</v51:id>
         </v5:deviceIds>
         <v5:deviceIds>
         <v51:id xsi:type="v51:MACAddressType"> 1,6,00:00:00:00:00:09</v51:id>
         </v5:deviceIds>
          <v5:deviceIds>
         <v51:id xsi:type="v51:MACAddressType"> 1,6,00:00:00:00:00:06</v51:id>
         </v5:deviceIds>
        <!--Optional:-->
         <v5:options>
            <!--Optional:-->
            <v51:executionOptions>
               <!--Optional:-->
               <v51:activationMode>NO_ACTIVATION</v51:activationMode>
               <!--Optional:-->
               <v51:confirmationMode>NO_CONFIRMATION</v51:confirmationMode>
               <!--Optional:-->
               <v51:publishingMode>NO_PUBLISHING</v51:publishingMode>
               <!--Optional:-->
               <v51:asynchronous>false</v51:asynchronous>
               <!--Optional:-->
               <v51:reliableMode>false</v51:reliableMode>
               <!--Optional:-->
               <v51:timeout>30000</v51:timeout>
               <!--Optional:-->
               <v51:stopOnFailure>true</v51:stopOnFailure>
               <!--Optional:-->
               <v51:transactionPerItem>false</v51:transactionPerItem>
            </v51:executionOptions>
            <!--Optional:-->
            <v51:operationOptions>
               <!--Zero or more repetitions:-->
```

```
               <v51:entry>
                  <v51:name>includeleaseinfo</v51:name>
                  <!--Optional:-->
                  <v51:value>true</v51:value>
               </v51:entry>
            </v51:operationOptions>
         </v5:options>
      </v5:getDevices>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:getDevicesResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:deviceOperationStatus>
            <cptype:operationStatus>

<cptype:operationId>529cbb30-25ba-4bfa-9ee9-6a244a405890</cptype:operationId>
               <cptype:code>SUCCESS</cptype:code>
               <cptype:message>Operation successful</cptype:message>
               <cptype:subStatus>
                  <cptype:status>

<cptype:txId>Batch:bac-rhel5-vm97/10.81.89.167:6186dd93:13cf1dbaef9:80000011</cptype:txId>
                     <cptype:cmdCodes>
                        <cptype:index>0</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>1</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>2</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>3</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:cmdCodes>
                        <cptype:index>4</cptype:index>
                        <cptype:code>CMD_OK</cptype:code>
                     </cptype:cmdCodes>
                     <cptype:code>BATCH_COMPLETED</cptype:code>
                     <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
                  </cptype:status>
               </cptype:subStatus>
            </cptype:operationStatus>
            <cptype:devices>
               <cptype:deviceIds>
                  <cptype:id xsi:type="cptype:MACAddressType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">1,6,00:00:00:00:00:06</cptype:id>
                  <cptype:macAddress>1,6,00:00:00:00:00:06</cptype:macAddress>
               </cptype:deviceIds>
               <cptype:leaseData>
                  <cptype:dhcpv4LeaseQueryData>
                     <cptype:entry>
                        <cptype:name>giaddr</cptype:name>
                        <cptype:value>10.106.2.58</cptype:value>
                     </cptype:entry>
```

```
                        <cptype:entry>
                           <cptype:name>dhcp-server-identifier</cptype:name>
                           <cptype:value>10.81.89.233</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-ipaddress</cptype:name>
                           <cptype:value>4.0.0.3</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-mac-address</cptype:name>
                           <cptype:value>1,6,00:00:00:00:00:06</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>dhcp-lease-time</cptype:name>
                           <cptype:value>603490</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>relay-agent-remote-id</cptype:name>
                           <cptype:value>00:00:00:00:00:06</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-id</cptype:name>
                           <cptype:value>01:00:00:00:00:00:06</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>relay-agent-circuit-id</cptype:name>
                           <cptype:value>80:01:03:ef</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>routers</cptype:name>
                           <cptype:value>4.0.0.1</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-last-transaction-time</cptype:name>
                           <cptype:value>1310</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>subnet-mask</cptype:name>
                           <cptype:value>255.255.255.0</cptype:value>
                        </cptype:entry>
                     </cptype:dhcpv4LeaseQueryData>
                  </cptype:leaseData>
               </cptype:devices>
               <cptype:devices>
                  <cptype:deviceIds>
                     <cptype:id xsi:type="cptype:MACAddressType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">1,6,00:00:00:00:00:07</cptype:id>
                     <cptype:macAddress>1,6,00:00:00:00:00:07</cptype:macAddress>
                  </cptype:deviceIds>
                  <cptype:leaseData>
                     <cptype:dhcpv4LeaseQueryData>
                        <cptype:entry>
                           <cptype:name>giaddr</cptype:name>
                           <cptype:value>10.106.2.58</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>dhcp-server-identifier</cptype:name>
                           <cptype:value>10.81.89.233</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-ipaddress</cptype:name>
                           <cptype:value>4.0.0.6</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
```

```
                        <cptype:name>client-mac-address</cptype:name>
                        <cptype:value>1,6,00:00:00:00:00:07</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>dhcp-lease-time</cptype:name>
                        <cptype:value>603490</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>relay-agent-remote-id</cptype:name>
                        <cptype:value>00:00:00:00:00:07</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-id</cptype:name>
                        <cptype:value>01:00:00:00:00:00:07</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>relay-agent-circuit-id</cptype:name>
                        <cptype:value>80:01:03:ef</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>routers</cptype:name>
                        <cptype:value>4.0.0.1</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-last-transaction-time</cptype:name>
                        <cptype:value>1310</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>subnet-mask</cptype:name>
                        <cptype:value>255.255.255.0</cptype:value>
                     </cptype:entry>
                  </cptype:dhcpv4LeaseQueryData>
               </cptype:leaseData>
            </cptype:devices>
            <cptype:devices>
               <cptype:deviceIds>
                  <cptype:id xsi:type="cptype:MACAddressType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">1,6,00:00:00:00:00:08</cptype:id>
                  <cptype:macAddress>1,6,00:00:00:00:00:08</cptype:macAddress>
               </cptype:deviceIds>
               <cptype:leaseData>
                  <cptype:dhcpv4LeaseQueryData>
                     <cptype:entry>
                        <cptype:name>giaddr</cptype:name>
                        <cptype:value>10.106.2.58</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>dhcp-server-identifier</cptype:name>
                        <cptype:value>10.81.89.233</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-ipaddress</cptype:name>
                        <cptype:value>4.0.0.4</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-mac-address</cptype:name>
                        <cptype:value>1,6,00:00:00:00:00:08</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>dhcp-lease-time</cptype:name>
                        <cptype:value>603489</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>relay-agent-remote-id</cptype:name>
```

```
                        <cptype:value>00:00:00:00:00:08</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-id</cptype:name>
                        <cptype:value>01:00:00:00:00:00:08</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>relay-agent-circuit-id</cptype:name>
                        <cptype:value>80:01:03:ef</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>routers</cptype:name>
                        <cptype:value>4.0.0.1</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-last-transaction-time</cptype:name>
                        <cptype:value>1311</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>subnet-mask</cptype:name>
                        <cptype:value>255.255.255.0</cptype:value>
                     </cptype:entry>
                  </cptype:dhcpv4LeaseQueryData>
               </cptype:leaseData>
            </cptype:devices>
            <cptype:devices>
               <cptype:deviceIds>
                  <cptype:id xsi:type="cptype:MACAddressType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">1,6,00:00:00:00:00:09</cptype:id>
                  <cptype:macAddress>1,6,00:00:00:00:00:09</cptype:macAddress>
               </cptype:deviceIds>
               <cptype:leaseData>
                  <cptype:dhcpv4LeaseQueryData>
                     <cptype:entry>
                        <cptype:name>giaddr</cptype:name>
                        <cptype:value>10.106.2.58</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>dhcp-server-identifier</cptype:name>
                        <cptype:value>10.81.89.233</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-ipaddress</cptype:name>
                        <cptype:value>4.0.0.5</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-mac-address</cptype:name>
                        <cptype:value>1,6,00:00:00:00:00:09</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>dhcp-lease-time</cptype:name>
                        <cptype:value>603490</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>relay-agent-remote-id</cptype:name>
                        <cptype:value>00:00:00:00:00:09</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>client-id</cptype:name>
                        <cptype:value>01:00:00:00:00:00:09</cptype:value>
                     </cptype:entry>
                     <cptype:entry>
                        <cptype:name>relay-agent-circuit-id</cptype:name>
                        <cptype:value>80:01:03:ef</cptype:value>
```

```
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>routers</cptype:name>
                           <cptype:value>4.0.0.1</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-last-transaction-time</cptype:name>
                           <cptype:value>1310</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>subnet-mask</cptype:name>
                           <cptype:value>255.255.255.0</cptype:value>
                        </cptype:entry>
                     </cptype:dhcpv4LeaseQueryData>
                  </cptype:leaseData>
               </cptype:devices>
               <cptype:devices>
                  <cptype:deviceIds>
                     <cptype:id xsi:type="cptype:MACAddressType"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">1,6,00:00:00:00:00:06</cptype:id>
                     <cptype:macAddress>1,6,00:00:00:00:00:06</cptype:macAddress>
                  </cptype:deviceIds>
                  <cptype:leaseData>
                     <cptype:dhcpv4LeaseQueryData>
                        <cptype:entry>
                           <cptype:name>giaddr</cptype:name>
                           <cptype:value>10.106.2.58</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>dhcp-server-identifier</cptype:name>
                           <cptype:value>10.81.89.233</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-ipaddress</cptype:name>
                           <cptype:value>4.0.0.3</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-mac-address</cptype:name>
                           <cptype:value>1,6,00:00:00:00:00:06</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>dhcp-lease-time</cptype:name>
                           <cptype:value>603489</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>relay-agent-remote-id</cptype:name>
                           <cptype:value>00:00:00:00:00:06</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-id</cptype:name>
                           <cptype:value>01:00:00:00:00:00:06</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>relay-agent-circuit-id</cptype:name>
                           <cptype:value>80:01:03:ef</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>routers</cptype:name>
                           <cptype:value>4.0.0.1</cptype:value>
                        </cptype:entry>
                        <cptype:entry>
                           <cptype:name>client-last-transaction-time</cptype:name>
                           <cptype:value>1311</cptype:value>
                        </cptype:entry>
```

```
                     <cptype:entry>
                        <cptype:name>subnet-mask</cptype:name>
                        <cptype:value>255.255.255.0</cptype:value>
                     </cptype:entry>
                  </cptype:dhcpv4LeaseQueryData>
               </cptype:leaseData>
            </cptype:devices>
         </ns2:deviceOperationStatus>
      </ns2:getDevicesResponse>
   </soap:Body>
</soap:Envelope>
```

Step 7 Delete five devices in a single SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.cisco.com/prime/cp/5.0">
   <soap:Header/>
   <soap:Body>
      <v5:deleteDevices>
         <v5:context>
            <v51:sessionId>${#TestSuite#SessionID}</v51:sessionId>
         </v5:context>
         <!--1st device -->
          <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:00</v51:id>
          </v5:deviceIds>
         <!--2nd device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:DUIDType">00:03:03:10:00:00:00:00:01</v51:id>
         </v5:deviceIds>
         <!--3rd device -->
         <v5:deviceIds>
               <!--<fqdn>${#TestSuite#deviceFQDN}</fqdn>-->
               <v51:id xsi:type="v51:DUIDType">00:03:03:10:00:00:00:00:02</v51:id>
         </v5:deviceIds>
         <!--4th device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:03</v51:id>
         </v5:deviceIds>
         <!--5th device-->
         <v5:deviceIds>
               <v51:id xsi:type="v51:MACAddressType">1,6,10:00:00:00:00:04</v51:id>
         </v5:deviceIds>
      </v5:deleteDevices>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:deleteDevicesResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>336dd134-0541-4f57-b195-08c0b054602b</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:864dfeb:13bb818d9d9:80000012</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
```

```
                  <cptype:cmdCodes>
                     <cptype:index>1</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:cmdCodes>
                     <cptype:index>2</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:cmdCodes>
                     <cptype:index>3</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:cmdCodes>
                     <cptype:index>4</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>BATCH_COMPLETED</cptype:code>
                  <cptype:batchCode>BATCH_COMPLETED</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:deleteDevicesResponse>
   </soap:Body>
</soap:Envelope>
```

Step 8 Close the open session.

Note Idle sessions are automatically closed after 15 minutes.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.cisco.com/prime/cp/5.0">
   <soap:Header/>
   <soap:Body>
      <v5:closeSession>
         <v5:context>
            <v51:sessionId>${#TestSuite#SessionID}</v51:sessionId>
         </v5:context>
      </v5:closeSession>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:closeSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>00de9199-66f8-44da-81d0-ea34aa663c82</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
         </ns2:operationStatus>
      </ns2:closeSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Reboot of Device or Devices

The devices in Prime Cable Provisioning can be rebooted using the reboot SOAP request.

Desired Outcome

Use this workflow to reboot a device or multiple devices.

Step 1 Create a connection with the respective RDU by sending a session SOAP request. The session SOAP request must contain user and RDU details as shown below.

```
<soap:Body>
      <v5:createSession>
         <v5:username>user</v5:username>
         <v5:password>password</v5:password>
         <v5:rduHost>rdu-1-lnx</v5:rduHost>
         <v5:rduPort>49187</v5:rduPort>
      </v5:createSession>
   </soap:Body>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:createSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <v5:context>
            <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
      </ns2:createSessionResponse>
   </soap:Body>
</soap:Envelope>
```

Use the device ID such as MAC address or DUID to reboot the device.

Step 2 Reboot the device using the following SOAP request.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:v5="http://www.cisco.com/prime/cp/v5"
xmlns:v51="http://www.cisco.com/prime/xsd/cp/v5">
   <soap:Header/>
   <soap:Body>
      <v5:rebootDevice>
         <v5:context>
            <v51:sessionId>B8CF2D089A46DED22834069780DCFEBEEFDE3B01</v51:sessionId>
         </v5:context>
         <v5:deviceId>
            <v51:macAddress type="MACAddressType">1,6,aa:00:00:00:00:01</v51:macAddress>
         </v5:deviceId>
         <v5:options>
           <v51:executionOptions>
             <v51:activationMode>AUTOMATIC</v51:activationMode>
              <v51:asynchronous>true</v51:asynchronous>
            </v51:executionOptions>
         </v5:options>
      </v5:rebootDevice>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
```

```
      <ns2:rebootDeviceResponse xmlns:cptype="http://www.cisco.com/prime/xsd/cp/v5"
xmlns:ns2="http://www.cisco.com/prime/cp/v5">
         <ns2:operationStatus>
            <cptype:operationId>84478a54-6ee3-42eb-a88d-03410883f990</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>

<cptype:txId>Batch:pcp-lnx-86.cisco.com/10.65.125.111:88bf3ab:15b83c95409:800000d4</cptype
:txId>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:rebootDeviceResponse>
   </soap:Body>
</soap:Envelope>
```

The device record is initiated for reboot operation in Prime Cable Provisioning.

Step 3 Use the transaction ID (txId) to retrieve the operation status from the Prime Cable Provisioning.

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
xmlns:ns="http://www.cisco.com/prime/cp/5.0">
   <soap:Header/>
   <soap:Body>
      <v5:pollOperationStatus>
         <v5:context>
           <!-- This session id is the response from the create session request -->
           <v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
         </v5:context>
         <!--Use the transaction Id from the reboot response -->

<v5:requestId>Batch:bacbl-63-14-lnx/127.0.0.1:33d869b2:13bbb9ced4e:8000003b</v5:requestId>
      </v5:pollOperationStatus>
   </soap:Body>
</soap:Envelope>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:pollOperationStatusResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>97a9eeb9-3ad8-4467-a913-e17052494499</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
            <cptype:message>Operation successful</cptype:message>
            <cptype:subStatus>
               <cptype:status>
                  <cptype:txId>Batch:bacbl-63-14-
lnx/127.0.0.1:33d869b2:13bbb9ced4e:8000003b</cptype:txId>
                  <cptype:cmdCodes>
                     <cptype:index>0</cptype:index>
                     <cptype:code>CMD_OK</cptype:code>
                  </cptype:cmdCodes>
                  <cptype:code>CMD_OK</cptype:code>
                  <cptype:batchCode>BATCH_WARNING</cptype:batchCode>
               </cptype:status>
            </cptype:subStatus>
         </ns2:operationStatus>
      </ns2:pollOperationStatusResponse>
   </soap:Body>
</soap:Envelope>
```

Step 4 Close the open session.

Note Idle sessions are automatically closed after 15 minutes.

```
<v5:closeSession>
<v5:context>
<!-- This session id is the response from the create session request-- >
<v51:sessionId>2F77B9B1A03B09F59438914BA3B20509E1661632</v51:sessionId>
</v5:context>
</v5:closeSession>
```

SOAP response:

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
   <soap:Body>
      <ns2:closeSessionResponse xmlns:ns2="http://www.cisco.com/prime/cp/5.0">
         <ns2:operationStatus>
            <cptype:operationId>bd013d5e-ba16-4687-bfdb-440d1ed960ec</cptype:operationId>
            <cptype:code>SUCCESS</cptype:code>
         </ns2:operationStatus>
      </ns2:closeSessionResponse>
   </soap:Body>
</soap:Envelope>
```

PWS Use Cases - RESTful

This section illustrates some of the common device related operations and RESTful request/response messages. For conciseness, only the request and response messages are depicted.

Note PWS RESTful can communicate only with PCP 6.1.3 RDU version or above. It is not compatible with RDUs of earlier releases of Prime Cable Provisioning.

This section includes the following use cases:

Registering a New Device

For a device to be operational in the home network and to get required access, a subscriber must connect the device to the network and register it. The device could be of type DOCSISModem, PacketCableMTA, CableHomeWanMan, CableHomeWanData, Set Top Box (STB), RPD, eRouter, or computer.

Desired Outcome

Use this workflow to register a device and to bring the device online with the appropriate level of service.

Step 1 Create a connection with the respective RDU by sending a create session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below:

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Step 2 Create class of service required for registering the device using a RESTful request. The supported device types are DOCSISModem, Set Top Box (STB), PacketCableMTA, RPD, eRouter, computer, CableHomeWanMan, and CableHomeWanData.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addCOS
```

Request Body:

```
{"deviceType":"DocsisModem"
,"sessionId":"D7BD8FADF897EB80365C5092030F187001FCA0F0"
,"name":"TestCOSRestPWS"
,"activationMode":"NO_ACTIVATION"
,"confirmationMode":"NO_CONFIRMATION"
,"asynchronous":"false"
,"reliableMode":"false"
,"timeout":"30000"
,"stopOnFailure":"true"
,"transactionPerItem":"false"
,"publishingMode":"NO_PUBLISHING"
,"properties":{"/cos/docsis/file":"gold.cm"}
}
```

RESTful response:

```
{
    "operationId": "9e61c86e-507e-4b31-b728-0f15302ebaff",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:80000112",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
```

```
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 3 Create DHCP criteria required for registering the device using RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addDHCPCriteria
```

Request body:

```
{
"name":"RESTDHCP",
"sessionId":"B6CFC958C3A445A0968754FFBDB9AF670EBE0762",
"properties":{"/fqdn/domain":"cisco167.com"},
"clientClass":"TestCC",
"excludeSelectionTags":"Testest",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```

RESTful response:

```
{
    "operationId": "665aa915-d637-4b69-beca-9784ee281072",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:8000014c",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 4 Add group using RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addGroup
```

Request body:

```
{
"name":"Mytestgroup",
```

```
"sessionId":"B6CFC958C3A445A0968754FFBDB9AF670EBE0762",
"groupType":"system",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```


RESTful response:


```
{
    "operationId": "f41d073f-6d94-45f4-a7c4-c9dd15efbc0d",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:80000150",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```


Step 5 Add a Custom property using RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addCustomProperty
```

Request body:


```
{"sessionId":"E5BE0A6F4F465CACBF2A35ACC1F799930B9972B1","customProperty":"testcus","dataTy
pe":"INTEGER","encryptedProperty":"true"}
```


RESTful response:


```
{
    "operationId": "1d8bfe0e-1080-46d9-bca4-ab2eddb89f9e",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:bac-lnx-xxx/127.0.0.1:4a68f274:1669a553ad7:8000000c",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK",
                        "message": null
                    }
                ],
```

```
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED",
                "message": null
            }
        ]
    }
}
```

Step 6 Use the device ID such as MAC address or DUID or FQDN and the device type for adding a new device to the RESTful request. You can assign a group to the new device.

The add device RESTful request also passes the subscriber's information, Class of Service, DHCP Criteria, Hostname and domain to Prime Cable Provisioning, which then registers the subscriber's device such as modem and computer.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addDevice
```

Request body:

```
{
"deviceType":"DOCSISModem",
"sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F",
"activationMode":"NO_ACTIVATION"
,"confirmationMode":"NO_CONFIRMATION"
,"asynchronous":"false"
,"reliableMode":"false"
,"timeout":"30000"
,"stopOnFailure":"true"
,"transactionPerItem":"false"
,"publishingMode":"NO_PUBLISHING"
,"cos":"unprovisioned-docsis"
,"hostName":"TestHost"
,"domainName":"Newtestdomain.com"
,"properties":{"/fqdn/domain":"domain.com"},
"deviceIdSet":{"macAddress":"1,6,00:00:00:00:12:02"}
}
```

RESTful response

```
{
    "operationId": "6b7385b8-fb4f-4da7-98f3-6c2da1bb03c8",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:8000011e",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 7 Provision the device by connecting it to the network. Prime Cable Provisioning provides the device its registered service level.

The device is now a provisioned device with access to the appropriate level of service.

Step 8 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
    "subStatus": null
}
```

Unregistering a Device

A device can be unregistered from Prime Cable Provisioning using the unregister RESTful request. This deletes the records from the Prime Cable Provisioning and subscriber gets the appropriate service level.

Desired Outcome

Unregistering a device removes the device details from Prime Cable Provisioning and gets the unprovisioned level of service.

Use this workflow to unregister a device and to bring the device online with the appropriate level of service.

Step 1 Create a connection with the respective RDU by sending a create session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below.

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Step 2 Use the device ID such as MAC address or DUID for unregistering a device. Unregister the device using the following RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/unregisterDevice
```

Request body:

```
{
```

```
"sessionId":"5851A4A10D8DDFA65B5111757F8C6647F16E6170",
"macAddress":"1,6,b0:00:00:00:00:02",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```

RESTful response

```
{
    "operationId": "c4f37622-c1d7-4cfa-b024-82c1f49a982a",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:705306be:1645c87ed08:8000005e",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

The device is now a unregistered and unprovisioned device with access to the appropriate level of service.

Step 3 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
    "subStatus": null
}
```

Getting DHCP Lease Information of a Device

A registered or unregistered device is assigned an IP address when it is provisioned. This IP address is used for collecting lease information such as, network register server state, and list of provisioning group in this lease.

Desired Outcome

Lists the details of the lease information from the Prime Cable Provisioning for the provisioned devices.

Use this workflow to collect the lease information of a device.

Step 1 Use an existing session or create a connection with the respective RDU by sending a create session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below.

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Step 2 Use the IP address along with its provisioning group name or just the IP address of the provisioned devices in the RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/getDHCPLeaseInfo
```

Request body:

```
{
"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
"resourceName":"10.0.0.17",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```

RESTful response:

```
{
    "operationStatus": {
        "operationId": "048db9a1-34d4-463d-ac46-26bb4730003c",
        "code": "SUCCESS",
        "message": "Operation successful",
        "subStatus": {
            "statusList": [
                {
                    "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:705306be:1645c87ed08:80000037",
                    "cmdCodes": [
                        {
```

```
                            "index": 0,
                            "code": "CMD_OK"
                        }
                    ],
                    "code": "CMD_OK",
                    "batchCode": "BATCH_COMPLETED"
                }
            ]
        }
    },
    "leaseResults": [
        {
            "device": {
                "id": null,
                "deviceType": "DOCSISModem",
                "subscriberId": null,
                "cos": null,
                "dhcpCriteria": null,
                "hostName": null,
                "domainName": null,
                "properties": {
                    "/provisioning/classOfService/selected": "unprovisioned-docsis",
                    "/provisioning/properties/selected": "{/docsis/version=3.1}",
                    "/provisioning/access": "DEFAULT",
                    "/provisioning/domain": "RootDomain",
                    "/provisioning/explanation": "Because the device is not registered",
                    "/generic/oidRevisionNumber": "1008806320825958504-1530560966000",
                    "/provisioning/provGroup": "default",
                    "/provisioning/isRegistered": "false",
                    "/provisioning/dhcpCriteria/selected": "unprovisioned-docsis",
                    "/provisioning/isInRequiredProvGroup": "true",
                    "/provisioning/reason": "NOT_REGISTERED",
                    "/node": "[]",
                    "/discoveredData/raw/dhcpv4": "{REQUEST={dhcp-class-identifier=AIC
Echo,docsis3.1:, chaddr=b0:00:00:00:00:01, relay-agent-circuit-id=01:04:80:01:03:ef,
vendor-encapsulated-options=04:0c:62:30:30:30:30:30:30:30:30:30:31:05:06:76:33:2e:32:2e
:31:06:06:76:31:2e:30:2e:32:07:07:42:4f:4f:54:31:2e:30:08:06:62:30:30:30:30:30:09:07:58:6d
:61:6e:32:30:30:0a:09:43:49:53:43:4f:20:62:61:63, dhcp-message-type=01,
v-i-vendor-opts=00:00:11:8b:84:01:01:02:05:7f:05:7d:01:01:01:02:01:02:03:01:01:04:01:01:05
:01:01:06:01:01:07:01:0f:08:01:10:09:01:00:0a:01:01:0b:01:18:0c:01:01:0d:02:01:00:0e:02:01
:00:0f:01:01:10:04:00:00:00:01:14:01:00:1d:01:03:1e:01:20:1f:01:10:20:01:10:21:01:01:22:01
:01:24:01:00:25:01:00:26:01:00:27:01:01:2f:04:00:00:00:00:30:02:00:00:31:01:00:32:01:00:33
:01:00:34:02:00:00:35:02:00:00:36:01:01:37:01:00:38:01:00:39:01:00, giaddr=0a:00:00:01,
relay-agent-remote-id=02:06:b0:00:00:00:00:01,
client-id-created-from-mac-address=00:00:00:00, htype=01,
relay-agent-info=01:04:80:01:03:ef:02:06:b0:00:00:00:00:01:09:0b:00:00:11:8b:06:01:04:01:0
2:03:01, client-id=ff:00:00:00:00:00:03:00:01:b0:00:00:00:00:01, hlen=06,
dhcp-parameter-request-list={01,03,06,07,0c,0f,33,36,04,02,43,42,7d}}}",
                    "/network/docsisVersion": "3.1",
                    "/provisioning/properties/detected": null,
                    "/provisioning/isBehindRequiredDevice": "true"
                },
                "discoveredData": {
                    "dhcpv4RequestData": {
                        "dhcp-class-identifier": "AIC Echo,docsis3.1:",
                        "chaddr": "b0:00:00:00:00:01",
                        "relay-agent-circuit-id": "01:04:80:01:03:ef",
                        "vendor-encapsulated-options": "(device-serial-number 4
b00000000001), (hardware-version-number 5 v3.2.1), (software-version-number 6 v1.0.2),
(boot-rom-version 7 BOOT1.0), (vendor-oui 8 b00000), (model-number 9 Xman200),
(vendor-name 10 CISCO bac)",
                        "dhcp-message-type": "1",
```

```
                        "v-i-vendor-opts": "enterprise-id 4491, (oro 1 2),
(modem-capabilities 5
05:7d:01:01:01:02:01:02:03:01:01:04:01:01:05:01:01:06:01:01:07:01:0f:08:01:10:09:01:00:0a:
01:01:0b:01:18:0c:01:01:0d:02:01:00:0e:02:01:00:0f:01:01:10:04:00:00:00:01:14:01:00:1d:01:
03:1e:01:20:1f:01:10:20:01:10:21:01:01:22:01:01:24:01:00:25:01:00:26:01:00:27:01:01:2f:04:
00:00:00:00:30:02:00:00:31:01:00:32:01:00:33:01:00:34:02:00:00:35:02:00:00:36:01:01:37:01:
00:38:01:00:39:01:00)",
                        "giaddr": "10.0.0.1",
                        "relay-agent-remote-id": "02:06:b0:00:00:00:00:01",
                        "client-id-created-from-mac-address": "0",
                        "htype": "01",
                        "relay-agent-info": "(circuit-id 1 80:01:03:ef), (remote-id 2
b0:00:00:00:00:01), (v-i-vendor-opts 9 enterprise-id 4491, (cmts-capabilities 1
(docsis-version 1 03:01)))",
                        "client-id": "ff:00:00:00:00:00:03:00:01:b0:00:00:00:00:01",
                        "hlen": "06",
                        "dhcp-parameter-request-list":
"{1,3,6,7,12,15,51,54,4,2,67,66,125}"
                    },
                    "dhcpv6RequestData": null,
                    "dhcpv6RelayRequestData": null
                },
                "leaseData": {
                    "dhcpv4LeaseQueryData": {
                        "relay-agent-circuit-id": "80:01:03:ef",
                        "client-last-transaction-time": "5214",
                        "relay-agent-remote-id": "b0:00:00:00:00:01",
                        "client-mac-address": "1,6,b0:00:00:00:00:01",
                        "client-ipaddress": "10.0.0.12",
                        "dhcp-lease-time": "599586",
                        "subnet-mask": "255.255.255.0",
                        "giaddr": "xx.xx.xxx.xx",
                        "client-id": "ff:00:00:00:00:00:03:00:01:b0:00:00:00:00:01",
                        "routers": "10.0.0.1",
                        "dhcp-server-identifier": "xx.xx.xxx.xx"
                    },
                    "dhcpv6LeaseQueryData": null
                },
                "groups": null,
                "deviceIdSet": {
                    "ids": [
                        {
                            "type": null,
                            "id": "1,6,b0:00:00:00:00:01"
                        }
                    ],
                    "macAddress": {
                        "type": null,
                        "id": "1,6,b0:00:00:00:00:01"
                    },
                    "duid": null,
                    "fqdn": null,
                    "deviceIDList": [
                        {
                            "keyType": "MAC_ADDRESS",
                            "deviceId": "1,6,b0:00:00:00:00:01"
                        },
                        {
                            "keyType": "MAC_ADDRESS",
                            "deviceId": "1,6,b0:00:00:00:00:01"
                        }
                    ],
                    "firstDeviceID": {
                        "keyType": "MAC_ADDRESS",
```

```
                        "deviceId": "1,6,b0:00:00:00:00:01"
                    }
                }
            },
            "v4": true,
            "v6": false,
            "active": true,
            "partialAnswer": false
        }
    ]
}
```

Step 3 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
    "subStatus": null
}
```

Updating Device Details

You can update the device's class of service, DHCP criteria, assign or unassign a group, and change or add properties. You can update any or all of this information as part of single updateDevice operation.

Embedded devices are not supported in Prime Cable Provisioning. The following workflow updates a single device. Before updating the device a new class of service and new dhcp criteria are added.

Desired Outcome

Prime Cable Provisioning regenerates the configuration file after updating the device with the required changes. The device then gets re-provisioned with updated level of service.

Use this workflow to update a device's details and provision it with the updated level of service.

Step 1 Create a connection with the respective RDU by sending a session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below.

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Step 2 Add a new class of service for the device using the following RESTful request:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addCOS
```

Request Body:

```
{"deviceType":"DocsisModem"
,"sessionId":"D7BD8FADF897EB80365C5092030F187001FCA0F0"
,"name":"TestCOSRestPWS"
,"activationMode":"NO_ACTIVATION"
,"confirmationMode":"NO_CONFIRMATION"
,"asynchronous":"false"
,"reliableMode":"false"
,"timeout":"30000"
,"stopOnFailure":"true"
,"transactionPerItem":"false"
,"publishingMode":"NO_PUBLISHING"
,"properties":{"/cos/docsis/file":"gold.cm"}
}
```

RESTful response:

```
{
    "operationId": "9e61c86e-507e-4b31-b728-0f15302ebaff",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:80000112",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 3 Add a new DHCP criteria for the device using the following RESTful request:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addDHCPCriteria
```

Request body:

```
{
"name":"RESTDHCP",
"sessionId":"B6CFC958C3A445A0968754FFBDB9AF670EBE0762",
```

```
"properties":{"/fqdn/domain":"cisco167.com"},
"clientClass":"TestCC",
"excludeSelectionTags":"Testest",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```

RESTful response:

```
{
    "operationId": "665aa915-d637-4b69-beca-9784ee281072",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:8000014c",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 4 If you want to assign a group that is not yet created, create it using the following RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addGroup
```

Request body:

```
{
"name":"Mytestgroup",
"sessionId":"B6CFC958C3A445A0968754FFBDB9AF670EBE0762",
"groupType":"system",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```

RESTful response:

```
{
    "operationId": "f41d073f-6d94-45f4-a7c4-c9dd15efbc0d",
    "code": "SUCCESS",
```

```
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:80000150",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```


Step 5 Use the device ID such as MAC address or DUID or FQDN and the device type to update the device details. You can update the subscriber's details as well as assign or unassign the device to a group by using the following RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateDevice
```

Request body:

```
{
    "macAddress":"1,6,00:00:00:00:12:02",
"deviceType":"DOCSISModem",
"sessionId":"7C47EBD1E3436F41869DFBC4B6CA7B7F29A63D26",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING",
"cos":"TestCOSRestPWS",
"hostName":"test",
"domainName":"cisco.com",
"properties[/fqdn/domain]":"techmahindra.com"
}
```

The device gets a regenerated configuration and is provisioned with updated service level.

RESTful response:


```
{
    "operationId": "c0692ec3-53fd-4ff9-acd2-c9b2dab37a20",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:738933c7:16457b49b5e:80000017",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    },
                    {
```

```
                        "index": 1,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 2,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 3,
                        "code": "CMD_OK"
                    }
                ],
                "code": "BATCH_COMPLETED",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 6 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
    "subStatus": null
}
```

Searching a Device

The subscriber has devices such as DOCSIS, MTA, DPoE and computer in Prime Cable Provisioning. These devices can be searched from the Prime Cable Provisioning and get the results as a list in RESTful response.

Desired Outcome

Searches device(s) from Prime Cable Provisioning based on specific search criteria.

Use this workflow to search a device and get the list that is generated based on the search criteria.

Step 1 Create a connection with the respective RDU by sending a session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below.

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Step 2 Use the query to create a search criteria.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DeviceSearchByDeviceIdPattern",
"returnParameters":"ALL",
"maxResults":"10",
"name":"1,6,22:*",
"deviceIDPattern":"macaddresspattern"
}
```

Note For the first query, the start element must not be included.

For all the supported search requests Search.query and Search.maxResults elements are required. Search.start is used to support paging across large search results. The first search operation must not include Search.start element. Its absence indicates that a new search is being initiated. A response to a search operation contains a SearchResult. SearchResult contains a *next* element which itself is a Search object. The entire content of the next element (query, start, size) should be used to page to the next batch of search results. The value of the start element is used by RDU's internal paging mechanism. Hence, from second request onwards, Search.start is required to page over the results. At the end of the search process, an empty SearchResult where SearchResult.size equals zero will be returned.

Step 3 Ensure that the response contains the next query request.

```
{
    "operationStatus": {
        "operationId": "4f98e5a2-c420-4c7a-8c62-73647e3557dc",
        "code": "SUCCESS",
        "message": "Operation successful",
        "subStatus": {
            "statusList": [
                {
                    "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:3715dcc8:1647b882056:8000001a",
                    "cmdCodes": [
                        {
                            "index": 0,
                            "code": "CMD_OK"
                        }
                    ],
                    "code": "CMD_OK",
                    "batchCode": "BATCH_COMPLETED"
                }
            ]
```

```
        }
    },
    "results": [
        {
            "id": null,
            "deviceType": "DOCSISModem",
            "subscriberId": null,
            "cos": null,
            "dhcpCriteria": null,
            "hostName": null,
            "domainName": null,
            "properties": {
                "/provisioning/classOfService/selected": "unprovisioned-docsis",
                "/provisioning/properties/selected": "{/docsis/version=3.1}",
                "/provisioning/access": "DEFAULT",
                "/provisioning/domain": "RootDomain",
                "/provisioning/explanation": "Because the device is not registered",
                "/generic/oidRevisionNumber": "1008806320825958501-1531081017000",
                "/provisioning/provGroup": "default",
                "/provisioning/isRegistered": "false",
                "/provisioning/dhcpCriteria/selected": "unprovisioned-docsis",
                "/provisioning/isInRequiredProvGroup": "true",
                "/provisioning/reason": "NOT_REGISTERED",
                "/node": "[]",
                "/discoveredData/raw/dhcpv4": "{REQUEST={dhcp-class-identifier=AIC
Echo,docsis3.1:, chaddr=ab:00:00:00:00:01, relay-agent-circuit-id=01:04:80:01:03:ef,
vendor-encapsulated-options=04:0c:61:62:30:30:30:30:30:30:30:30:31:05:06:76:33:2e:32:2e
:31:06:06:76:31:2e:30:2e:32:07:07:42:4f:4f:54:31:2e:30:08:06:61:62:30:30:30:30:09:07:58:6d
:61:6e:32:30:30:0a:09:43:49:53:43:4f:20:62:61:63, dhcp-message-type=01,
v-i-vendor-opts=00:00:11:8b:84:01:01:02:05:7f:05:7d:01:01:01:02:01:02:03:01:01:04:01:01:05
:01:01:06:01:01:07:01:0f:08:01:10:09:01:00:0a:01:01:0b:01:18:0c:01:01:0d:02:01:00:0e:02:01
:00:0f:01:01:10:04:00:00:00:01:14:01:00:1d:01:03:1e:01:20:1f:01:10:20:01:10:21:01:01:22:01
:01:24:01:00:25:01:00:26:01:00:27:01:01:2f:04:00:00:00:00:30:02:00:00:31:01:00:32:01:00:33
:01:00:34:02:00:00:35:02:00:00:36:01:01:37:01:00:38:01:00:39:01:00, giaddr=0a:00:00:01,
relay-agent-remote-id=02:06:ab:00:00:00:00:01,
client-id-created-from-mac-address=00:00:00:00, htype=01,
relay-agent-info=01:04:80:01:03:ef:02:06:ab:00:00:00:00:01:09:0b:00:00:11:8b:06:01:04:01:0
2:03:01, client-id=ff:00:00:00:00:00:03:00:01:ab:00:00:00:00:01, hlen=06,
dhcp-parameter-request-list={01,03,06,07,0c,0f,33,36,04,02,43,42,7d}}}",
                "/network/docsisVersion": "3.1",
                "/provisioning/properties/detected": null,
                "/provisioning/isBehindRequiredDevice": "true"
            },
            "discoveredData": {
                "dhcpv4RequestData": {
                    "dhcp-class-identifier": "AIC Echo,docsis3.1:",
                    "chaddr": "ab:00:00:00:00:01",
                    "relay-agent-circuit-id": "01:04:80:01:03:ef",
                   "vendor-encapsulated-options": "(device-serial-number 4 ab0000000001),
(hardware-version-number 5 v3.2.1), (software-version-number 6 v1.0.2), (boot-rom-version
7 BOOT1.0), (vendor-oui 8 ab0000), (model-number 9 Xman200), (vendor-name 10 CISCO bac)",
                    "dhcp-message-type": "1",
                   "v-i-vendor-opts": "enterprise-id 4491, (oro 1 2), (modem-capabilities
5
05:7d:01:01:01:02:01:02:03:01:01:04:01:01:05:01:01:06:01:01:07:01:0f:08:01:10:09:01:00:0a:
01:01:0b:01:18:0c:01:01:0d:02:01:00:0e:02:01:00:0f:01:01:10:04:00:00:00:01:14:01:00:1d:01:
03:1e:01:20:1f:01:10:20:01:10:21:01:01:22:01:01:24:01:00:25:01:00:26:01:00:27:01:01:2f:04:
00:00:00:00:30:02:00:00:31:01:00:32:01:00:33:01:00:34:02:00:00:35:02:00:00:36:01:01:37:01:
00:38:01:00:39:01:00)",
                    "giaddr": "10.0.0.1",
                    "relay-agent-remote-id": "02:06:ab:00:00:00:00:01",
                    "client-id-created-from-mac-address": "0",
                    "htype": "01",
```

```
                    "relay-agent-info": "(circuit-id 1 80:01:03:ef), (remote-id 2
ab:00:00:00:00:01), (v-i-vendor-opts 9 enterprise-id 4491, (cmts-capabilities 1
(docsis-version 1 03:01)))",
                    "client-id": "ff:00:00:00:00:00:03:00:01:ab:00:00:00:00:01",
                    "hlen": "06",
                    "dhcp-parameter-request-list": "{1,3,6,7,12,15,51,54,4,2,67,66,125}"
                },
                "dhcpv6RequestData": null,
                "dhcpv6RelayRequestData": null
            },
            "leaseData": null,
            "groups": null,
            "deviceIdSet": {
                "ids": [
                    {
                        "type": null,
                        "id": "1,6,ab:00:00:00:00:01"
                    }
                ],
                "macAddress": {
                    "type": null,
                    "id": "1,6,ab:00:00:00:00:01"
                },
                "duid": null,
                "fqdn": null,
                "deviceIDList": [
                    {
                        "keyType": "MAC_ADDRESS",
                        "deviceId": "1,6,ab:00:00:00:00:01"
                    },
                    {
                        "keyType": "MAC_ADDRESS",
                        "deviceId": "1,6,ab:00:00:00:00:01"
                    }
                ],
                "firstDeviceID": {
                    "keyType": "MAC_ADDRESS",
                    "deviceId": "1,6,ab:00:00:00:00:01"
                }
            }
        }
    ],
    "size": 1,
    "next": {
        "query": null,
        "start": "1,6,ab:00:00:00:00:01",
        "QueryValue": null,
        "maxResults": 10,
        "propertyFilter": null
    }
}
```


Step 4 Use the next element contents for the next search query criteria.

Step 5 Repeat step 4 and step 5 till you receive the search response size as zero.

All the devices are available based on the search request.

Step 6 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
    "subStatus": null
}
```

Note You can also run search query based on different object types such as, device, Class of Service, DHCP Criteria, file and group. A sample search query pattern with these object types is shown below:

Supported Query Elements

The supported search elements in Prime Cable Provisioning 6.1.3 are:

Searching basic device details of a registered device with class of service:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DeviceSearchByCOS",
"returnParameters":"ALL",
"maxResults":"1",
"associationType":"SELECTED",
"name":"TestDocsis"
}
```

Searching basic device details of a registered device with DHCP criteria:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DeviceSearchByDHCPCriteria",
"returnParameters":"ALL",
"maxResults":"1",
"associationType":"SELECTED",
```

```
"name":"unprovisioned-docsis"
}
```

Searching basic device details of a registered device with default CoS device type:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"CosSearchByDeviceType",
"maxResults":"1",
"deviceType":"DOCSISModem"
}
```

Searching basic device details of a registered device with default DHCP criteria device type:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DeviceSearchByDefaultDHCPCriteria",
"returnParameters":"ALL",
"maxResults":"1",
"associationType":"SELECTED",
"name":"DOSCISModem"
}
```

Searching all the device details of a device with device type:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DeviceSearchByDeviceType",
"returnParameters":"ALL",
"maxResults":"1",
"name":"DOSCISModem"
}
```

Searching all the device details of a device with group name:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DeviceSearchByGroupName",
"returnParameters":"ALL",
"maxResults":"1",
"associationType":"SELECTED",
"name":"system-diagnostics"
}
```

Search the entire device by ownerId:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DeviceSearchByOwnerId",
"maxResults":"1",
"name":"cisco"
}
```

Searching all class of service with device type:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"CosSearchByDeviceType",
"maxResults":"1",
"deviceType":"DOCSISModem"
}
```

Searching all DHCP criteria:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"DHCPCriteriaSearch",
"maxResults":"1",
"deviceType":"DOCSISModem"
}
```

Searching all files with file type:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"FileSearchByFileType",
"maxResults":"1",
"fileType":"MIB"
}
```

Searching all files with a file name pattern:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
```

```
"query":"FileSearchByFileNamePattern",
"maxResults":"1",
"name":"*.cm"
}
```

Searching all groups with group type:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"GroupSearchByGroupType",
"maxResults":"1",
"name":"system"
}
```

Searching all groups with a group name pattern:

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices
```

Request body:

```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"query":"GroupSearchByGroupNamePattern",
"maxResults":"1",
"groupNamePattern":"*"
}
```

Deleting a Device

The existing devices in Prime Cable Provisioning can be deleted using the delete RESTful request.

Desired Outcome

Deleting a device removes the device details from the Prime Cable Provisioning.

Use this workflow to delete a device.

Step 1 Create a connection with the respective RDU by sending a session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below.

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Use the device ID such as MAC address or DUID to delete the device.

Step 2 Delete the device using the following RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteDevice
```

Request body:

```
{
    "sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F",
    "macAddress":"1,6,aa:00:00:00:00:01"
}
```

RESTful response:

```
{
    "operationId": "b6160005-ba83-4b81-bf07-eee3f8586efe",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:800000b9",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 3 Close the open sessions.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
    "subStatus": null
}
```

The device record is deleted from Prime Cable Provisioning.

Multiple Devices Operations in a Single Request

Multiple devices in Prime Cable Provisioning can be added, retrieved, updated, or deleted using a single RESTful request.

Desired Outcome

Use this workflow to add, retreive, update, and delete multiple devices.

Step 1 Create a connection with the respective RDU by sending a session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below.

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Step 2 Add five devices using a single RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/addDevices
```

Request body:

```
{
"sessionId":"F61D528E364BC96F6956FF9A73FFB1ABA7C2371E",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING",
"multiDeviceList":
[
    { "deviceIdSet": {"macAddress":"1,6,00:00:00:00:12:b2"},"deviceType":"DOCSISModem"},
    { "deviceIdSet": {"macAddress":"1,6,00:00:00:00:12:b1"},"deviceType":"Computer"}

]
 }
```

RESTful response:

```
{
    "operationId": "7a02904c-38a6-4c80-9b04-16ee37056d71",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:80000144",
                "cmdCodes": [
                    {
```

```
                        "index": 0,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 1,
                        "code": "CMD_OK"
                    }
                ],
                "code": "BATCH_COMPLETED",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 3 Using a single RESTful request, you can get more than one device.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/getDevices
```

Request body:

```
{
"sessionId":"57F67427C06F3C3E1DDA8186472ED39C3E9741E1",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"propertyMap":{"includeleaseinfo":"true"},
"multiDeviceIdList":
[
{"macAddress":"1,6,00:00:00:00:12:22","deviceType":"DOCSISModem"},

{"macAddress":"1,6,00:00:00:00:12:29","deviceType":"Computer"}


]
}
```

RESTful response:

```
[
    {
        "devices": [
            {
                "id": null,
                "deviceType": null,
                "subscriberId": null,
                "cos": null,
                "dhcpCriteria": null,
                "hostName": null,
                "domainName": null,
                "properties": {
                    "/provisioning/provGroup": "default"
                },
                "discoveredData": null,
                "leaseData": {
                    "dhcpv4LeaseQueryData": {
                        "relay-agent-circuit-id": "80:01:03:ef",
                        "client-last-transaction-time": "2542",
                        "relay-agent-remote-id": "b0:00:00:00:00:01",
                        "client-mac-address": "1,6,b0:00:00:00:00:01",
```

```
                "client-ipaddress": "10.0.0.12",
                "dhcp-lease-time": "602258",
                "subnet-mask": "255.255.255.0",
                "giaddr": "xx.xx.xxx.xx",
                "client-id": "ff:00:00:00:00:00:03:00:01:b0:00:00:00:00:01",
                "routers": "10.0.0.1",
                "dhcp-server-identifier": "xx.xx.xxx.xx"
            },
            "dhcpv6LeaseQueryData": null
        },
        "groups": null,
        "deviceIdSet": {
            "ids": [
                {
                    "type": null,
                    "id": "1,6,b0:00:00:00:00:01"
                }
            ],
            "macAddress": {
                "type": null,
                "id": "1,6,b0:00:00:00:00:01"
            },
            "duid": null,
            "fqdn": null,
            "deviceIDList": [
                {
                    "keyType": "MAC_ADDRESS",
                    "deviceId": "1,6,b0:00:00:00:00:01"
                },
                {
                    "keyType": "MAC_ADDRESS",
                    "deviceId": "1,6,b0:00:00:00:00:01"
                }
            ],
            "firstDeviceID": {
                "keyType": "MAC_ADDRESS",
                "deviceId": "1,6,b0:00:00:00:00:01"
            }
        }
    },
    {
        "id": null,
        "deviceType": null,
        "subscriberId": null,
        "cos": null,
        "dhcpCriteria": null,
        "hostName": null,
        "domainName": null,
        "properties": {
            "/provisioning/provGroup": "default"
        },
        "discoveredData": null,
        "leaseData": {
            "dhcpv4LeaseQueryData": {
                "relay-agent-circuit-id": "80:01:03:ef",
                "client-last-transaction-time": "2577",
                "relay-agent-remote-id": "a0:00:00:00:00:01",
                "client-mac-address": "1,6,aa:00:00:00:00:01",
                "client-ipaddress": "10.0.0.6",
                "dhcp-lease-time": "602223",
                "subnet-mask": "255.255.255.0",
                "giaddr": "xx.xx.xxx.xx",
                "client-id": "ff:00:00:00:00:00:03:00:01:a0:00:00:00:00:01",
                "routers": "10.0.0.1",
```

```
                            "dhcp-server-identifier": "xx.xx.xxx.xx"
                        },
                        "dhcpv6LeaseQueryData": null
                    },
                    "groups": null,
                    "deviceIdSet": {
                        "ids": [
                            {
                                "type": null,
                                "id": "1,6,aa:00:00:00:00:01"
                            }
                        ],
                        "macAddress": {
                            "type": null,
                            "id": "1,6,aa:00:00:00:00:01"
                        },
                        "duid": null,
                        "fqdn": null,
                        "deviceIDList": [
                            {
                                "keyType": "MAC_ADDRESS",
                                "deviceId": "1,6,aa:00:00:00:00:01"
                            },
                            {
                                "keyType": "MAC_ADDRESS",
                                "deviceId": "1,6,aa:00:00:00:00:01"
                            }
                        ],
                        "firstDeviceID": {
                            "keyType": "MAC_ADDRESS",
                            "deviceId": "1,6,aa:00:00:00:00:01"
                        }
                    }
                }
            ],
            "operationStatus": {
                "operationId": "de0b1eba-3cea-43e0-a25a-af53a72e1eb0",
                "code": "SUCCESS",
                "message": "Operation successful",
                "subStatus": {
                    "statusList": [
                        {
                            "txId":
"Batch:pcp-lnx-x/xx.xx.xxx.xx:705306be:1645c87ed08:8000000f",
                            "cmdCodes": [
                                {
                                    "index": 0,
                                    "code": "CMD_OK"
                                },
                                {
                                    "index": 1,
                                    "code": "CMD_OK"
                                }
                            ],
                            "code": "BATCH_COMPLETED",
                            "batchCode": "BATCH_COMPLETED"
                        }
                    ]
                }
            }
        }
]
```

Step 4 Using a single RESTful request, you can update more than one device.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateDevices
```

Request body:

```
{
"sessionId":"33A145758AF8D8377CF1370F1AF2CBD8CB4FD2ED",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING",
"domainName":"test231.com",
"properties":{"/fqdn/domain":"domain.com"},
"multiDeviceIdList":
[
{"macAddress":"1,6,00:00:00:00:12:22","deviceType":"DOCSISModem"},

{"macAddress":"1,6,00:00:00:00:12:29","deviceType":"Computer"}


]
}
```

RESTful response:

```
{
    "operationId": "07b1ae46-714f-40e6-a1c4-250001709312",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:3a0658b2:16459e4d58b:8000001a",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 1,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 2,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 3,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 4,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 5,
                        "code": "CMD_OK"
                    }
                ],
                "code": "BATCH_COMPLETED",
```

```
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 5 Using a single RESTful request, you can delete more than one device.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteDevice
```

Request body:

```
{
    "sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F",
    "macAddress":"1,6,aa:00:00:00:00:01"
}
```

RESTful response:

```
{
    "operationId": "9627c062-24d5-451a-9b3a-bb96bc83298c",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:4c43dbf3:164388870cb:800000da",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    },
                    {
                        "index": 1,
                        "code": "CMD_OK"
                    }
                ],
                "code": "BATCH_COMPLETED",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```

Step 6 Close the open session.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
```

```
    "subStatus": null
}
```

Reboot of Device or Devices

The devices in Prime Cable Provisioning can be rebooted using the reboot RESTful request.

Desired Outcome

Use this workflow to reboot a device or multiple devices.

Step 1 Create a connection with the respective RDU by sending a session RESTful request using POST method. The session RESTful request must contain user and RDU details as shown below.

```
http://pcp-lnx-x:9101/cp-ws-rest-prov/createSession
```

Request Body:

```
{"rduHost": "xx.xx.xxx.xx","rduPort": "49187","username":
"username","password":"password"}
```

RESTful response:

```
sessionId:B886BDD61590296429B530379AF5469CDAF78CF5
```

Use the device ID such as MAC address or DUID to reboot the device.

Step 2 Reboot the device using the following RESTful request.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/rebootDevice
```

Request body:

```
{
    "sessionId":"975B4C17880CF31FB606F4BDF8CBCBC11BC4F68A",
"macAddress":"1,6,b0:00:00:00:00:02",
"activationMode":"AUTOMATIC",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"true",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```

RESTful response:

```
{
    "operationId": "dcbe2462-7811-4581-8b79-4560fb0ac95d",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
```

```
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:705306be:1645c87ed08:80000057",
                "cmdCodes": [],
                "code": null,
                "batchCode": null
            }
        ]
    }
}
```


The device record is initiated for reboot operation in Prime Cable Provisioning.

Step 3 Use the transaction ID (txId) to retrieve the operation status from the Prime Cable Provisioning.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/pollOperationStatus
```

Request body:


```
{
"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
"requestID":"Batch:bac-lnx-xxx/xx.xx.xxx.xx:b8c61a6:165404e05bc:8000009b",
"activationMode":"NO_ACTIVATION",
"confirmationMode":"NO_CONFIRMATION",
"asynchronous":"false",
"reliableMode":"false",
"timeout":"30000",
"stopOnFailure":"true",
"transactionPerItem":"false",
"publishingMode":"NO_PUBLISHING"
}
```


RESTful response:


```
{
    "operationId": "2e647a75-b5f3-4604-ac08-7f972422974a",
    "code": "SUCCESS",
    "message": "Operation successful",
    "subStatus": {
        "statusList": [
            {
                "txId": "Batch:pcp-lnx-x/xx.xx.xxx.xx:705306be:1645c87ed08:80000057",
                "cmdCodes": [
                    {
                        "index": 0,
                        "code": "CMD_OK"
                    }
                ],
                "code": "CMD_OK",
                "batchCode": "BATCH_COMPLETED"
            }
        ]
    }
}
```


Step 4 Close the open session.

Note Idle sessions are automatically closed after 15 minutes.

```
http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession
```

Request body:

```
{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}
```

RESTful response:

```
{
    "operationId": "07591855-50bd-4e8d-9345-3202eae89331",
    "code": "SUCCESS",
    "message": null,
    "subStatus": null
}
```

PWS RESTful Operations

A URL and for parameters, a JSON like syntax is used to post the RESTful Provisioning Web Service.

This section includes the samples for PWS RESTful operations:

- Create Session
- Add Operation
- Update Operation
- Get Operation
- Delete Operation
- Status, Reboot, Unregister, Search Operations
- Close Session

Table 8-46 Create Session

Operation	URL	Request Body	Method
Create Session	http://bac-lnx-xxx:9101/cp-ws-rest-prov/createSession	{"rduHost": "bac-lnx-xxx","rduPort": "49187","username": "username","password":"password"}	POST

Table 8-47 Add Operation

Operation	URL	Request Body	Method
ADD COS	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addCOS	{"deviceType":"DocsisModem" ,"sessionId":"D7BD8FADF897EB80365C5092030F187001FCA0F0" ,"name":"TestCOSRestPWS" ,"activationMode":"NO_ACTIVATION" ,"confirmationMode":"NO_CONFIRMATION" ,"asynchronous":"false" ,"reliableMode":"false" ,"timeout":"30000" ,"stopOnFailure":"true" ,"transactionPerItem":"false" ,"publishingMode":"NO_PUBLISHING" ,"properties":{"/cos/docsis/file":"gold.cm"} }	POST

ADD Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addDevice	{
		"deviceType":"DOCSISModem",
		"sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F",
		"activationMode":"NO_ACTIVATION"
		,"confirmationMode":"NO_CONFIRMATION"
		,"asynchronous":"false"
		,"reliableMode":"false"
		,"timeout":"30000"
		,"stopOnFailure":"true"
		,"transactionPerItem":"false"
		,"publishingMode":"NO_PUBLISHING"
		,"cos":"unprovisioned-docsis"
		,"hostName":"TestHost"
		,"domainName":"Newtestdomain.com"
		,"properties":{"/fqdn/domain":"domain.com"},
		"deviceIdSet":{"macAddress":"1,6,00:00:00:00:12:02"}
		}
			POST

Table 8-47 Add Operation (continued)

ADD Devices	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addDevices	{
		"sessionId":"F61D528E364BC96F6956FF9A73FFB1ABA7C2371E",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING",
		"multiDeviceList":
		[
		{ "deviceIdSet": {"macAddress":"1,6,00:00:00:00:00:12:b2"},"deviceType":"DOCSISModem"},
		{ "deviceIdSet": {"macAddress":"1,6,00:00:00:00:00:12:b1"},"deviceType":"Computer"}
		
		]
		}
			POST

ADD DHCP Criteria	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addDHCPCriteria	{
		"name":"RESTDHCP",
		"sessionId":"B6CFC958C3A445A0968754FFBDB9AF670EBE0762",
		"properties":{"/fqdn/domain":"cisco167.com"},
		"clientClass":"TestCC",
		"excludeSelectionTags":"Testest",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Table 8-47 Add Operation (continued)

ADD File	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addFile	{"sessionId":"A37481F3AD753387D969ECDBB466D7BE0AD2D269","fileType":"GENERIC","fileName":"Test.txt","stream":"Testing Scenario"}	POST
	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addFile	{ "sessionId":"A37481F3AD753387D969ECDBB466D7BE0AD2D269", "fileType": "GENERIC", "fileName": "nega9.txt", "fileDataBytes": [35, 33, 47, 98, 105, 110, 47, 115]}	POST

ADD Group	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addGroup	{
		"name":"Mytestgroup",
		"sessionId":"B6CFC958C3A445A0968754FFBDB9AF670EBE0762",
		"groupType":"system",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Add Custom Property without encryption property	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addCustomProperty	{"sessionId": "E5BE0A6F4F465CACBF2A35ACC1F799930B9972B1", "customProperty":"testCus","dataType":"INTEGER"}	POST
Add Custom Property with encryption property	http://bac-lnx-xxx:9101/cp-ws-rest-prov/addCustomProperty	{"sessionId": "E5BE0A6F4F465CACBF2A35ACC1F799930B9972B1", "customProperty":"testCus","dataType":"INTEGER", "encryptedProperty":"true"}	POST

Table 8-48 Update Operation

Operation	URL	Request Body	Method

Update COS	http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateCOS	{
		"sessionId":"7C47EBD1E3436F41869DFBC4B6CA7B7F29A63D26",
		"resourceName":"TestCOSRestPWS",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING",
		"properties":{"/cos/docsis/file":"silver.cm"},
		"list[0]":"/fqdn/domain"
		}
			PUT

Update Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateDevice	{
		"macAddress":"1,6,00:00:00:00:12:02",
		"deviceType":"DOCSISModem",
		"sessionId":"7C47EBD1E3436F41869DFBC4B6CA7B7F29A63D26",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING",
		"cos":"TestCOSRestPWS",
		"hostName":"test",
		"domainName":"cisco.com",
		"properties[/fqdn/domain]":"techmahindra.com"
		}
			PUT

Table 8-48 Update Operation (continued)

Update Devices	http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateDevices	{
		"sessionId":"33A145758AF8D8377CF1370F1AF2CBD8CB4FD2ED",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING",
		"domainName":"test231.com",
		"properties":{"/fqdn/domain":"domain.com"},
		"multiDeviceIdList":
		[
		{"macAddress":"1,6,00:00:00:00:12:22","deviceType":"DOCSISModem"},
		
		{"macAddress":"1,6,00:00:00:00:12:29","deviceType":"Computer"}
		
		]
		}
			PUT

Table 8-48 Update Operation (continued)

Update DHCP Criteria	http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateDHCPCriteria	{
		"resourceName":"RESTDHCP",
		"sessionId":"7C47EBD1E3436F41869DFBC4B6CA7B7F29A63D26",
		"properties":{"/fqdn/domain":"cisco197.com"},
		"clientClass":"CiscoTech",
		"excludeSelectionTags":"CiscoTest",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			PUT

Update File	http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateFile	{
		"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
		"resourceName":"Test.txt",
		"stream":"Testing Scenario Updated"
		}
			PUT

Table 8-48 Update Operation (continued)

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateFile	{ "sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77", "fileName": "cisco.txt", "fileDataBytes": [35, 33, 47, 98, 105, 110, 47]	PUT

Update Group	http://bac-lnx-xxx:9101/cp-ws-rest-prov/updateGroup	{
		"resourceName":"Mytestgroup",
		"sessionId":"7C47EBD1E3436F41869DFBC4B6CA7B7F29A63D26",
		"groupType":"mysystem",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			PUT

Table 8-49 Get Operation

Operation	URL	Request Body	Method

Get COS	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getClassOfService	{
		"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
		"resourceName":"sample-gold-docsis",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Get Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getDevice	{
		"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
		"macAddress":"1,6,00:00:00:00:12:02",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Table 8-49 Get Operation (continued)

Get Devices	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getDevices	{
		"sessionId":"57F67427C06F3C3E1DDA8186472ED39C3E9741E1",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"propertyMap":{"includeleaseinfo":"true"},
		"multiDeviceIdList":
		[
		{"macAddress":"1,6,00:00:00:00:12:22","deviceType":"DOCSISModem"},
		{"macAddress":"1,6,00:00:00:00:12:29","deviceType":"Computer"}
		]
		}
			POST

Get Behind Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getBehindDevice	{
		"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
		"macAddress":"1,6,00:00:00:00:12:02",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Table 8-49 Get Operation (continued)

Get Behind Devices	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getBehindDevices	{
		"sessionId":"1EAFD21803275D48CDEE9243EE31A25D0CC8A9DE",
		"multiDeviceIdList":
		[
		{"macAddress":"1,6,22:00:00:00:00:09"},
		{"macAddress":"1,6,00:00:00:00:12:29"}
		]
		}
			POST

Get DHCP Criteria	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getDHCPCriteria/	{
		"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
		"resourceName":"RESTDHCP",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Table 8-49 Get Operation (continued)

Get DHCP LeaseInfo	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getDHCPLeaseInfo	{
		"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
		"resourceName":"10.0.0.17",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Get File	http://pcp-lnx-79:9101/cp-ws-rest-prov/getFile	{
		"sessionId": "68A2AB172E6DFC8AC1F85F33F75CE01B0AC0463D",
		"resourceName": "cisco.txt",
		"retrievedata": "false"
		}
			POST

Table 8-49 Get Operation (continued)

Get Group	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getGroup	{
		"sessionId":"3A8AE27D1D2F5F7CB58C10E25B7F95E8D5754A77",
		"resourceName":"system-diagnostics",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Get Custom Property	http://bac-lnx-xxx:9101/cp-ws-rest-prov/getCustomProperty	{"sessionId": "E5BE0A6F4F465CACBF2A35ACC1F799930B9972B1"}	POST

Table 8-50 Delete Operation

Operation	URL	Request Body	Method

Delete COS	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteCOS	{
		"resourceName":"TestCOSCM",
		"sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F"
		}
			Delete

Delete Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteDevice	{
		"sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F",
		"macAddress":"1,6,aa:00:00:00:00:01"
		}
			Delete

Table 8-50 Delete Operation (continued)

Delete Devices	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteDevices	{
		"sessionId":"55736B06B8877C4E864F37CDA2923558A3CF7A9A",
		"multiDeviceIdList":
		[
		{"macAddress":"1,6,b0:00:00:00:00:02"},
		{"macAddress":"1,6,b0:00:00:00:00:03"}
		]
		}
			Delete

Delete DHCP Criteria	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteDHCPCriteria	{
		"resourceName":"RESTDHCP",
		"sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F"
		}
			Delete

Delete File	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteFile	{
		"resourceName":"test.txt",
		"sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F"
		}
			Deleter

Delete Group	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteGroup	{
		"resourceName":"Mytestgroup",
		"sessionId":"42C76DAEAA76B1E90DC1F440D9E7E2F8D179C04F"
		}
			Delete

Delete Custom Property	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deleteCustomProperty	{"sessionId": "E5BE0A6F4F465CACBF2A35ACC1F799930B9972B1", "customProperty":"testCus"}	Delete

Table 8-51 Status, Reboot, Unregister, Search Operations

Operation	URL	Request Body	Method

Status Test	http://bac-lnx-xxx:9101/cp-ws-rest-prov/statusTest	{
		"statusTest":"Good"
		}
			POST

Reboot Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/rebootDevice	{
		"sessionId":"975B4C17880CF31FB606F4BDF8CBCBC11BC4F68A",
		"macAddress":"1,6,b0:00:00:00:00:02",
		"activationMode":"AUTOMATIC",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"true",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Regen Configs	http://bac-lnx-xxx:9101/cp-ws-rest-prov/regenConfigs	{
		"sessionId":"975B4C17880CF31FB606F4BDF8CBCBC11BC4F68A",
		"maxResults":"10",
		"query":"DeviceSearchByDefaultDHCPCriteria",
		"name":"DOCSISModem",
		"associationType":"REGISTERED"
		}
			POST

Table 8-51 Status, Reboot, Unregister, Search Operations (continued)

Device Operation	http://bac-lnx-xxx:9101/cp-ws-rest-prov/deviceOperation	{
		"sessionId":"2B9F8FE59CBD0F61F043F4F89D140598475E6BB2",
		"multiDeviceIdList":
		[
		{"macAddress":"1,6,00:00:00:00:12:22"},
		{"macAddress":"1,6,00:00:00:00:12:29"}
		
		],
		"asynchronous":"true",
		"command":"RESET",
		"activationMode":"AUTOMATIC"
		}
			POST

Unregister Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/unregisterDevice	{
		"sessionId":"5851A4A10D8DDFA65B5111757F8C6647F16E6170",
		"macAddress":"1,6,b0:00:00:00:00:02",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			PUT

Table 8-51 Status, Reboot, Unregister, Search Operations (continued)

Unregister Devices	http://bac-lnx-xxx:9101/cp-ws-rest-prov/unregisterDevices	{
		"sessionId":"5851A4A10D8DDFA65B5111757F8C6647F16E6170",
		"multiDeviceIdList":
		[
		{"macAddress":"1,6,00:00:00:00:00:12:22"},
		{"macAddress":"1,6,00:00:00:00:00:12:29"}
		
		],
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			PUT

Poll Operation Status	http://bac-lnx-xxx:9101/cp-ws-rest-prov/pollOperationStatus	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"requestID":"Batch:bac-lnx-xxx/xx.xx.xxx.xx:b8c61a6:165404e05bc:8000009b",
		"activationMode":"NO_ACTIVATION",
		"confirmationMode":"NO_CONFIRMATION",
		"asynchronous":"false",
		"reliableMode":"false",
		"timeout":"30000",
		"stopOnFailure":"true",
		"transactionPerItem":"false",
		"publishingMode":"NO_PUBLISHING"
		}
			POST

Table 8-51 Status, Reboot, Unregister, Search Operations (continued)

Search Device	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByDeviceIdPattern",
		"returnParameters":"ALL",
		"maxResults":"10",
		"name":"1,6,22:*",
		"deviceIDPattern":"macaddresspattern"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByCOS",
		"returnParameters":"ALL",
		"maxResults":"1",
		"associationType":"SELECTED",
		"name":"TestDocsis"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByDefaultCOS",
		"returnParameters":"ALL",
		"maxResults":"1",
		"associationType":"SELECTED",
		"name":"DOSCISModem"
		}
			POST

Table 8-51 Status, Reboot, Unregister, Search Operations (continued)

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByDHCPCriteria",
		"returnParameters":"ALL",
		"maxResults":"1",
		"associationType":"SELECTED",
		"name":"unprovisioned-docsis"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByDefaultDHCPCriteria",
		"returnParameters":"ALL",
		"maxResults":"1",
		"associationType":"SELECTED",
		"name":"DOSCISModem"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByDeviceType",
		"returnParameters":"ALL",
		"maxResults":"1",
		"name":"DOSCISModem"
		}
			POST

Table 8-51 Status, Reboot, Unregister, Search Operations (continued)

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByGroupName",
		"returnParameters":"ALL",
		"maxResults":"1",
		"associationType":"SELECTED",
		"name":"system-diagnostics"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByProvGroupName",
		"returnParameters":"ALL",
		"maxResults":"1",
		"associationType":"SELECTED",
		"name":"default"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DeviceSearchByOwnerId",
		"maxResults":"1",
		"name":"cisco"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"CosSearchByDeviceType",
		"maxResults":"1",
		"deviceType":"DOCSISModem"
		}
			POST

Table 8-51 Status, Reboot, Unregister, Search Operations (continued)

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"DHCPCriteriaSearch",
		"maxResults":"1",
		"deviceType":"DOCSISModem"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"FileSearchByFileNamePattern",
		"maxResults":"1",
		"name":"*.cm"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"FileSearchByFileType",
		"maxResults":"1",
		"fileType":"MIB"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"GroupSearchByGroupNamePattern",
		"maxResults":"1",
		"groupNamePattern":"*"
		}
			POST

Table 8-51 Status, Reboot, Unregister, Search Operations (continued)

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"GroupSearchByRelatedGroup",
		"maxResults":"1",
		"name":"system-diagnostics"
		}
			POST

	http://bac-lnx-xxx:9101/cp-ws-rest-prov/searchDevices	{
		"sessionId":"B34F984E600B25357CD0615BB0A8A27619E5E783",
		"query":"GroupSearchByGroupType",
		"maxResults":"1",
		"name":"system"
		}
			POST

Table 8-52 Close Session

Operation	URL	Request Body	Method
Close Session	http://bac-lnx-xxx:9101/cp-ws-rest-prov/closeSession	{"sessionId":"9E70A267079DB31B402BCB87941BED9031A3060F"}	POST