



## CHAPTER 6

# License Inventory Management Functions

---

This chapter provides information about the following license inventory management functions:

- [annotate\\_licenses](#), page 6-1
- [deploy\\_licenses](#), page 6-2
- [get\\_allowed\\_operation\\_by\\_device\\_platform](#), page 6-3
- [get\\_licenses\\_on\\_device](#), page 6-4
- [get\\_rehostable\\_skus\\_by\\_device](#), page 6-4
- [get\\_rehost\\_info](#), page 6-5
- [init\\_rehost\\_license](#), page 6-6
- [list\\_all\\_licenses\\_in\\_pak](#), page 6-7
- [obtain\\_license](#), page 6-7
- [obtain\\_license\\_for\\_rehost](#), page 6-8
- [re\\_obtain\\_license](#), page 6-9
- [read\\_licenses](#), page 6-9
- [rehost\\_license](#), page 6-10
- [resend\\_license](#), page 6-11
- [revoke\\_license\\_for\\_rehost](#), page 6-12
- [write\\_licenses](#), page 6-12

## annotate\_licenses

### Synopsis

```
annotate_licenses ($token, [@lic_ids], [@annotation])
```

### Description

This function allows you to annotate a license with comments that you provide.

This function blocks until the call completes. It returns a `Cisco::CLM::Common::IDStatus` object containing the status of the operation.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
lic_ids	Array of string, mandatory	ID is a string containing up to 256 ASCII characters in the range from x21 to x7A	ID of License objects.
annotation	Array of string, mandatory	Text string up to 99 characters	Text of the annotation for each license. Cisco License Manager does not check the length of the annotation parameter. Cisco IOS software truncates the text if it exceeds the character limit.

**Return**

The function returns a `Cisco::CLM::Common::IDStatus` on completion.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

## deploy\_licenses

**Synopsis**

```
deploy_licenses ($token, [@lic_ids])
```

**Description**

This function deploys the given licenses to their target devices.

This function blocks until the call completes. It returns a `Cisco::CLM::Common::IDStatus` object containing the status of the operation.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
lic_ids	Array of string, mandatory	ID is a string containing up to 256 ASCII characters in the range from x21 to x7A	An array of license ID.

**Return**

The function returns a `Cisco::CLM::Common::IDStatus` on completion.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

## get\_allowed\_operation\_by\_device\_platform

**Synopsis**

```
get_allowed_operation_by_device_platform ($token, $platform)
```

**Description**

This function returns an array of allowed license operations for a specified platform.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
platform	<code>Cisco::CLM::Common::Device::DevicePlatform</code> mandatory	CISCO12K, CSL, IPS, UNKNOWN, URLF, WNBU	Device Platform.

**Return**

The function returns an array of `Cisco::CLM::Common::Device::LicenseOperation` for a given Device Platform. If the given Device Platform is UNKNOWN, it will return an empty array.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

## get\_licenses\_on\_device

### Synopsis

```
get_licenses_on_device ($token, $dev_id)
```

### Description

This function retrieves license information that resides on the given device.

### Input Parameters

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
dev_ids	String, mandatory	ID is a string containing up to 256 ASCII characters in the range from x21 to x7A	The Device ID string.

### Return

This function returns a Cisco::CLM::Common::LicenseStatus object.

### Error and Exception

If a system error prevents the operation from completing, a RemoteException is thrown.

When an error occurs for an element in the input array, the error code and error message is contained in the returned status object.

## get\_rehostable\_skus\_by\_device

### Synopsis

```
get_rehostable_skus_by_device ($token, $dev_id)
```

### Description

This function retrieves an array of SKUs that can be used for license rehost for the specified input device. This function will in turn trigger a request to SWIFT to query all SKUs whose licenses have been deployed on the device.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
dev_id	String, mandatory	ID is a string containing up to 64 ASCII characters in the range from x21 to x7A	ID of the device.

**Return**

This function returns a `Cisco::CLM::Common::RehostableSKUStatus` object. If there are no licenses obtained for the device, the `RehostableSKU` field in the returned `RehostableSKUStatus` is set to null.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

When error occurs, the information is contained in the returned status object.

## get\_rehost\_info

**Synopsis**

```
get_rehost_info ($token, [@dev_ids])
```

**Description**

This function returns the `RehostInfo` of each given device. Each `RehostInfo` contains a rehost request and a permission ticket or a rehost ticket.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
dev_id	String Array, mandatory	ID is a string containing up to 256 ASCII characters in the range from x21 to x7A	An array of device IDs.

**Return**

This function returns a `Cisco::CLM::Common::RehostInfoStatus` object.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

The `RehostInfoStatus` object contains the error code and error message if an operation error occurs. Otherwise, you must traverse the `RehostInfoStatusItem` array to retrieve all of the `RehostInfo` objects.

## init\_rehost\_license

**Synopsis**

```
init_rehost_license ($token, $rehost_req)
```

**Description**

The limitation of rehosting from the Cisco Product License Registration Portal is that there can be only one `PermissionTicket` acquired per device until a new license is obtained. This means that there is only one `PermissionTicket` and one `RehostTicket` per device at any time.

This function is the first step of the rehost process. The process consists of several steps, including getting a permission ticket from the Cisco Product License Registration Portal, retrieving the rehost ticket from the device, sending the rehost ticket to the Cisco Product License Registration Portal to obtain the license, and deploying the license to the destination device.

The obtained `PermissionTicket` is stored in local storage and is later used to revoke the license from the source device.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
rehost_req	RehostRequest, mandatory	—	An object that represents the request.

**Return**

This function returns the `Cisco::CLM::Common::Status` object.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

`Status` contains an error code and an error message if the operation is not successful.

# list\_all\_licenses\_in\_pak

## Synopsis

```
list_all_licenses_in_pak ($token, $pak_id)
```

## Description

This function returns an array of License IDs that belong to the given PAK.

## Input Parameters

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
pak_id	String, mandatory	ID is a string containing up to 64 ASCII characters in the range from x21 to x7A	The PAK ID that contains the licenses.

## Return

This function returns a string array of License ID contained by the PAK. If the pak\_id is invalid, this function returns null.

## Error and Exception

If a system error prevents the operation from completing, a RemoteException is thrown.

When an error occurs, this function returns null.

# obtain\_license

## Synopsis

```
obtain_license ($token, [@lic_req], $deploy)
```

## Description

This function downloads the information that is associated with the given product authorization key (PAK) IDs from the Cisco Product License Registration Portal and stores the information in the inventory. The first function only obtains the licenses; the second function obtains the licenses and deploys them.

This function blocks until the call completes. It returns a Cisco::CLM::Common::IDStatus object containing the status of the operation.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
lic_reqs	Array of LicenseRequest, mandatory	—	An array of LicenseRequest.
deploy	Boolean, mandatory	True, False	True to ask the server to deploy all licenses obtained.

**Return**

The function returns a `Cisco::CLM::Common::IDStatus` on completion.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

## obtain\_license\_for\_rehost

**Synopsis**

```
obtain_license_for_rehost ($token, $rehost_req)
```

**Description**

The limitation of rehosting from the Cisco Product License Registration Portal is that there can be only one `PermissionTicket` acquired per device until a new license is obtained. This means that there is only one `PermissionTicket` and one `RehostTicket` per device at any time.

This function is the third step of the rehost process. The process consists of several steps, including getting a permission ticket from the Cisco Product License Registration Portal, retrieving the rehost ticket from the device, sending the rehost ticket to the Cisco Product License Registration Portal to obtain the license, and deploying the license to the destination device.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
rehost_req	RehostRequest, mandatory	—	An object that represents the request.

**Return**

This function returns a `Cisco::CLM::Common::LicenseStatus` object.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

The `Status` object contains an error code and error message if an operation error occurs. Otherwise, you must traverse the `LicenseStatusItem` array to retrieve all of the license objects.

## re\_obtain\_license

**Synopsis**

```
re_obtain_license ($token, $dev_id)
```

**Description**

This function requests that the Cisco Product License Registration Portal resend the license. After licenses are received, it updates and synchronizes Cisco License Manager data storage. It does not deploy licenses to a device.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
dev_id	String	ID is a string containing up to 64 ASCII characters in the range from x21 to x7A	ID of device to which to resend the license.

**Return**

This function returns a `Cisco::CLM::Common::IDStatus` object.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

When an error occurs, the information is contained in the returned `Status` object.

## read\_licenses

**Synopsis**

```
read_licenses ($token, [@lic_ids])
```

**Description**

This function retrieves an array of license objects from the inventory using the given device IDs.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
lic_ids	Array of string, mandatory	ID is a string containing up to 256 ASCII characters in the range from x21 to x7A	An array of License ID.

**Return**

This function returns a `Cisco::CLM::Common::LicenseStatus` object.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

When an error occurs on an element in the input array, a `LicenseStatus` object is returned with information about the error.

## rehost\_license

**Synopsis**

```
rehost_license ($token, $rehost_req)
```

**Description**

This function sends requests to rehost licenses from one device to another. This process contains several steps, including retrieving a permission ticket from the Cisco Product License Registration Portal, retrieving a rehost ticket from the source device, and sending the rehost ticket to the Cisco Product License Registration Portal to obtain new licenses for the destination device and deploy the new licenses to the destination device. These steps are encapsulated by this function as a single operation. The obtained license is stored in local storage and can be used later to be deployed to the destination devices.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
rehost_req	RehostRequest, mandatory	—	An object that represents the request.

**Return**

This function returns a `Cisco::CLM::Common::Status` object, which contains the error code and message. If the operation is successful, the `ClmErrors.SUCCESS` error code is returned. If it is unsuccessful, the `none` `ClmErrors.SUCCESS` error code and the error message are returned.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

When an error occurs, this function returns a `Status` object which details the error code and error message.

## resend\_license

**Synopsis**

```
resend_license ($token, $dev_id)
```

**Description**

This function resends licenses to a device to restore corrupted license files. The function requests all licenses that have been obtained from the Cisco Product License Registration Portal, saves them into the License Manager database, and then replays them to the device.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
dev_id	String, mandatory	ID is a string containing up to 64 ASCII characters in the range from x21 to x7A	ID of device to which to resend the license.

**Return**

This function returns a `Cisco::CLM::Common::Status` object, which contains the error code and error message. If the operation is successful, the `ClmErrors.SUCCESS` error code is returned.

**Error and Exception**

If a system error prevents the operation from completing, a `RemoteException` is thrown.

When an error occurs, this function returns `false`.

# revoke\_license\_for\_rehost

## Synopsis

```
revoke_license_for_rehost ($token, $rehost_req)
```

## Description

This function handles the situation when rehost fails in the middle of an operation. The limitation of rehosting from the Cisco Product License Registration Portal is that there can be only one PermissionTicket acquired per device until a new license is obtained. This means that there is only one PermissionTicket and one RehostTicket per device at any time.

This function is the second step of the rehost process. The process consists of several steps, including retrieving a permission ticket from the Cisco Product License Registration Portal, retrieving the rehost ticket from the device, sending the rehost ticket to the Cisco Product License Registration Portal to obtain the license, and deploying the license to the destination device.

The obtained PermissionTicket is stored in local storage and is later used to revoke the license from the source device. It is removed if the revoke operation is successful, and the RehostTicket is stored in local storage for next step of rehost process.

## Input Parameters

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
rehost_req	RehostRequest, mandatory	—	An object that represents the request.

## Return

This function returns a Cisco::CLM::Common::Status object.

## Error and Exception

If a system error prevents the operation from completing, a RemoteException is thrown.

Status contains an error code and error message if the operation is not successful.

# write\_licenses

## Synopsis

```
write_licenses ($token, [@lics])
```

## Description

This function writes the given License objects into the inventory. The input License objects can be existing instances of License retrieved from the inventory by the function read\_licenses.

**Input Parameters**

Parameter	Type	Value	Description
token	UserToken, mandatory	—	A token that represents your authorization pass, which is obtained after you invoke the login function and are authenticated by the back-end server.
lics	Array of License, mandatory	ID is a string containing up to 256 ASCII characters in the range from x21 to x7A	An array of License objects.

**Return**

This function returns a Cisco::CLM::Common::IDStatus object.

**Error and Exception**

If a system error prevents the operation from completing, a RemoteException is thrown.

When an error occurs on an element in the input array, a status object is returned with information about the error.

