Cisco Optical Network Controller 2.1 REST API Guide

First Published: April 27, 2023

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies are considered un-Controlled copies and the original on-line version should be referred to for latest version.

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com/go/trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Preface

This guide provides information about the APIs exposed by Cisco Optical Network Controller.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see *What's New in Cisco Product Documentation*.

To receive new and revised Cisco technical content directly to your desktop, you can subscribe to the What's New in Cisco Product Documentation RSS feed. The RSS feeds are a free service.

Introduction

Overview of Cisco Optical Network Controller REST API

Cisco Optical Network Controller is an optical domain controller which provides integration capabilities for external tools using REST APIs. Clients must authenticate themselves with Cisco Optical Network Controller before accessing these REST APIs. Any authenticated client can communicate with Cisco Optical Network Controller using these REST APIs to perform a range of functions that include:

- Device discovery
- Device deletion
- Device resync
- IPC creation
- IPC deletion
- Planning data import
- Alien import
- Fetching alarms information
- Fetching PM information
- Pre-provision passives
- Passive deletion
- Fetching topology and interface information
- Cross Connect deletion
- Cross Connect Fan Out Allocation
- Fetching Inventory
- Fetching logs
- Fetching IPC information
- Fetching Site information
- Site-Device Association
- Configuration Bulk Push
- Editing Rack View
- Connection Verification
- Patch Loss Measurement
- Device Addition
- Device Update
- Full Network Sync and status
- Onboard devices from file import
- Fetch Audit Log
- Fetch Audit Log for service

The following sections describe the REST API endpoints available to you.
Note: It is possible to enable and disable additional APIs for troubleshooting and debugging under the supervision of Cisco TAC.

Device Manager

Bulk Onboard

Description:

This API is used to onboard the devices into CONC using XLS file. The required information is input in the XLS file and then its fed into the ONC System.

URL: onc-devicemanager-service/v2/import/file
Type : POST
Inputs :
Request Body:
importDevice.xls(Sample file for import)

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Host Name	Node IP	Password Type	User Name	Password	Connectivity Type	Connectivity Port	ectivity Tir	oxy Addre	.1 Transpo	Site Name	Product Type	GRPC Port
2	vxrSite_1	10.64.98.20	USER_NETCONF	cisco	Ci$c0@321	NETCONF	65285	300			site_vxrSite_1	NCS1010	64501
3	vxrSite_2	10.64.98.20	USER_NETCONF	cisco	Ci$c0@321	NETCONF	61680	300			site_vxrSite_2	NCS1010	61991
4	vxrSite_3	10.64.98.20	USER_NETCONF	cisco	Ci$c0@321	NETCONF	60953	300			site_vxrSite_3	NCS1010	61994
5	vxrSite_4	10.64.98.20	USER_NETCONF	cisco	Ci$c0@321	NETCONF	63524	300			site_vxrSite_4	NCS1010	62297
6	vxrSite_5	10.64.98.20	USER_NETCONF	cisco	Ci$c0@321	NETCONF	61746	300			site_vxrSite_5	NCS1010	60509

Response:

```
{
   "httpCode": 200,
   "message": "Bulk Import initiated successfully"
}
```

Error conditions:

- Only one bulk import can be initiated at any given time. If another bulk import is initiated while one is ongoing user will get the following error:
  405 - Operation not allowed. An import operation is in-progress. Please wait till it completes
- If the XLS has incorrect format, user will get the following error -

```
{
   "httpCode": 400,
   "message": [
      "No Sheet with name Devices in the provided XLS"
   ]
}
```

- 500-XLS Processing failed ::
- 400- Mandatory columns : [GRPC Port] were missing in the provided XLS
- 400- column has invalid value.
- 409-ERROR: cannot add more than 16 devices per site
- 409- ERROR: Devices already present in DB, cannot add more than 16 devices per site

- 405 - Operation not allowed. An import operation is in-progress. Please wait till it completes
- 400- Invalid Ip Address format
- 400- Invalid port number
- 400-Invalid grpc port number
- 400- Invalid connectivity timeout value. Please provide value between 60-300 seconds
- 409- Device Already Exists with name
- 409 - Device Already Exists with Same IP - %s and port - %s
- 409 - Duplicate Site Error, SiteName " + siteName + " is already used"
- 409 - ERROR:Existing site " + siteName + " is of 1010 type,SVO Devices are not allowed in this site
- 409- ERROR:Existing site " + siteName + " is of SVO type,1010 Devices are not allowed in this site
- 409 - Site Deletion " + siteName + " is in progress, cannot add devices to the site
- 500 - Internal server error
- 500-Protocol not supported

Add Device

Description:

This API is used to onboard one device at a time to CONC by giving ip, port, sitename, type, username, and password.

URL: onc-devicemanager-service/v2/devices/device
Type : POST
Inputs :
Request Body:
Sample JSON to onboard a device

```
{
   "ipAddress": "10.1.1.1",
   "name": "svo1",
   "owner": "admin",
   "password": "cisco123",
   "port": 333,
   "protocol": "NETCONF",
   "siteName": "site1",
   "state": "ENABLED",
   "type": "SVO",
   "username": "admin",
```

```
    "grpcPort": 333
}

Response:
{
    "deviceId": "a4037a0c-a33c-34f9-8fcd-4989e7d0503d",
    "username":
"AAAADAsex+Yg83ZKq0Gn4AWsOVaRmG1HlvrjMfowt4Pj+Rw4cA==",
    "password":
"AAAADJ1KtGH+bCnVb+V9rv3m2gH9WFPCmQAQFkaXEH+n9BbeQ/1fuw==",
    "ipAddress": "10.1.1.2",
    "port": 333,
    "name": "svo1",
    "type": "SVO",
    "grpcPort": 333,
    "statusCode": 202,
    "status": null,
    "importId": null,
    "collectorStatus": "Waiting_for_connection",
    "deployerStatus": "Waiting_for_connection",
    "serviceStates": null,
    "owner": "admin",
    "creationTime": null,
    "siteName": "site1",
    "lastUpdated": "2023-04-06T18:41:02.040962",
    "lockStatus": "UNLOCK",
    "message": "Device Addition Requested",
    "version": null,
    "protocol": "NETCONF",
    "state": "ENABLED",
    "proxyAddress": null,
    "tl1Transport": null,
    "deviceVersion": null,
    "deleteAcks": null,
    "lastSuccessfulCollectionTime": null
}
```


Error Conditions:

- Error format:


```
{
  "errorCode": "400",
```

"errorMessage": "Invalid Ip Address format",
"path": "/devicemanager/v2/devices/device",
"httpStatus": "BAD_REQUEST"
}

- 405 - Operation not allowed. An import operation is in-progress. Please wait till it completes
- 400- Invalid Ip Address format
- 400- Invalid port number
- 400-Invalid grpc port number
- 400- Invalid connectivity timeout value. Please provide value between 60-300 seconds
- 409- Device Already Exists with name
- 409 - Device Already Exists with Same IP - %s and port - %s
- 409 - Duplicate Site Error, SiteName " + siteName + " is already used"
- 409 - ERROR:Existing site " + siteName + " is of 1010 type,SVO Devices are not allowed in this site
- 409- ERROR:Existing site " + siteName + " is of SVO type,1010 Devices are not allowed in this site
- 409 - Site Deletion " + siteName + " is in progress, cannot add devices to the site
- 409 - ERROR: cannot add more than 16 devices per site
- 500 - Internal server error
- 500-Protocol not supported

Get Devices

Description:

This API is used to fetch the information that is required in the UI, about the devices that are present in the ONC.

URL: /api/onc-devicemanager-service/v3/devices
Type: GET
Inputs:
Request Body: NA
Response:

[
{
"deviceId": "67dd22fb-8caf-3ef2-b312-28bc85406cdb",
"ipAddress": "10.64.98.20",
"port": 61680,
"name": "vxrSite_2",
"type": "NCS1010",
"serviceStates": {

```
      "Circuit": "COMPLETED",
      "Topology": "COMPLETED",
      "Inventory": "COMPLETED"
    },
    "collectionFailedCount": "0",
    "lockStatus": "UNLOCK",
    "protocol": "NETCONF",
    "status": "Connected",
    "collectorStatus": "Connected",
    "deployerStatus": "Connected",
    "siteName": "site_vxrSite_2",
    "state": "ENABLED",
    "resyncState": null,
    "resyncCount": null,
    "lastSuccessfulCollectionTime": 1675929444606,
    "grpcPort": 61991,
    "message": "Successfully reconnected to device"
  }
]
```


Get Single Device Detail

Description:

This API is used to fetch the Device Details information based on the device name.
URL: /api/onc-devicemanager-service/v2/devices/device/{name}
Type: GET
Inputs: ron_ncs1010_olt1(devicename)
Request Body:
Response:


```
{
    "deviceId": "5b5bc40a-b4e3-3b43-8a15-3aae1dd43327",
    "username": "cisco",
    "password": "AAAADAYeEa/7LNLySoHRjgwG4uPGtpHhLQVXrw2DdkWr4uOytOG/rIo=",
    "ipAddress": "10.64.98.20",
    "port": 61691,
    "name": "vxrSite_2",
    "type": "NCS1010",
    "grpcPort": 65072,
    "statusCode": 200,
    "status": "Connected",
```

```
    "importId": null,
    "collectorStatus": "Connected",
    "deployerStatus": "Connected",
    "serviceStates": {
      "Circuit": "COMPLETED",
      "Topology": "COMPLETED",
      "Inventory": "COMPLETED"
    },
    "owner": "admin",
    "creationTime": "2023-04-18T12:05:00.350813",
    "siteName": "site_vxrSite_2",
    "lastUpdated": "2023-04-18T12:53:47.340246",
    "lockStatus": "UNLOCK",
    "message": "Successfully reconnected to device",
    "version": null,
    "protocol": "NETCONF",
    "state": "ENABLED",
    "proxyAddress": null,
    "tl1Transport": null,
    "deviceVersion": "0",
    "deleteAcks": null,
    "lastSuccessfulCollectionTime": 1681822427324
}
```


Error conditions:

- If device name is not valid - 404-Device not present

Get Banner Details

Description:

This API is used to fetch the information that is required in the UI, on the files that are currently being imported into the ONC. It has information like if the Resync is in progress, Import Status, Last updated time of import.

URL: /api/onc-devicemanager-service/v2/devices/getBannerDetails

Type: GET

Inputs:

Request Body: NA

Response:


```
{
  "isFullNetworkSyncInProgress": false,
  "nextFNSSchedule": 1676419200895,
  "isXlsImportinProgress": false,
  "xlsImportLastUpdate": "Tue, 14 Feb 2023 10:51:54 GMT",
  "lastFNSPerformedAt": "Not Performed yet",
```

"xlsImportStatus": "COMPLETED"
}

Error conditions:

- 500 - Internal server error

Fetch list of device types

Description:

This API is used to fetch the list of all supported device types like SVO, NCS1010.

URL: /api/onc-devicemanager-service/v2/devices/supported-device-types

Type: GET

Inputs:

Request Body: NA

Response:

[
"NCS1004",
"NCS4000",
"SVO",
"NCS2000",
"NCS1010"
]

Error conditions:

- 500 - Internal server error

Delete The Device

Description:

This API call is used to delete devices based on the device name.

URL: /api/onc-devicemanager-service/v2/devices

Type: DELETE

Inputs:

Request Body:

[
{

```
    "name": "vxrSite_2"
  }
]
```


Response:


```
[
 {
   "name": "vxrSite_2",
   "statusCode": "204",
   "statusMessage": "Deletion in Progress"
 }
                                                                    ]
```


Error conditions:

- 500 - Internal server error
- 405-Operation not allowed. An import operation is in-progress. Please wait till it completes
- 405-No devices requested for deletion
- 405-Device cannot be deleted because Circuit spanning across the device
- 405-Device cannot be deleted because Waiting for Connection
- 405-Device cannot be deleted because Collection is in Progress
- 405-Device Deletion is already in Progress
- 404-Device requested for deletion does not exist

Reconnect Device

Description: This API is used to reconnect the device based on the device name when device is in disconnected state.

URL: onc-devicemanager-service/v2/devices/{deviceName}

Type: POST

Inputs:

Parameters: deviceName

Response:


```
{
"deviceId": "7709884a-bc32-357d-9e6f-4d7dac1843f7",
"username": "AAAADBHkCqjOOlkvmCVFJAoWbsxt+bI68x7tkyjvLtGSG+57Mg==",
"password":"AAAADMR3LA4Y60K9kKKPQXQmPTLHB+hh2e/+c/11VgHbqh+HnW0AtcY=,
    "ipAddress": "10.64.98.19",
      "port": 63776,
      "name": "vxr59_site1",
      "type": "NCS1010",
    "grpcPort": 65305,
    "statusCode": 200,
    "status": "Connected",
```

"importId": null,
"collectorStatus": "Connected", /*Internal status indicating connectivity from CONC to devices for data collection and notifications */
"deployerStatus": "Connected", /*Internal status indicating connectivity from CONC to devices for configuring devices */
"serviceStates": {
"Circuit": "COMPLETED",
"Topology": "COMPLETED",
"Inventory": "COMPLETED"
},
"owner": "admin",
"creationTime": null,
"siteName": "india",
"lastUpdated": "2023-04-21T09:00:30.075343",
"lockStatus": "UNLOCK", /* Indicates if a device is locked from deleting*/
"message": "Successfully reconnected to device",
"version": null,
"protocol": "NETCONF",
"state": "ENABLED",
"proxyAddress": null,
"tl1Transport": null,
"deviceVersion": "0",
"deleteAcks": null,
"lastSuccessfulCollectionTime": 1682067172422
}

Error conditions:

- User cannot reconnect device if an import operation is already in progress. User will get the following error:

{
errorCode: "405",
errorMessage: "Operation not allowed. An import operation is in-progress. Please wait till it completes",
httpStatus: "METHOD_NOT_ALLOWED",
path: "/onc-devicemanager-service/v2/devices/vxr59_site1"
}

- User cannot reconnect device if device deletion is in process.

{
errorCode: "405",
errorMessage: "Deletion_in_progress",
httpStatus: "METHOD_NOT_ALLOWED",
path: "/onc-devicemanager-service/v2/devices/vxr59_site1"
}

- When user trigger reconnect device and provided device name does not exist.

{

```
errorCode: "404"
errorMessage: "Device Does not Exists",
      httpStatus: "NOT_FOUND",
      path: "/onc-devicemanager-service/v2/devices/vxr59_site1"
    }
```

Resync Devices

Description: This API is used to resync bulk number of devices for the given device name list.

URL: onc-devicemanager-service/v2/resync/devices

Type: POST

Inputs:

Request Body: ["vxrSite_3","vxrSite_4"]

Response:

```
[
 {
      "name": "vxrSite_3",
      "statusCode": "409",
      "statusMessage": "Collection is in Progress, resync not triggered"
 },
 {
      "name": "vxrSite_4",
      "statusCode": "204",
      "statusMessage": "Resync triggered for device vxrSite_4"
 }
]
```

Error conditions:

1. User cannot resync devices if an import operation is already in progress. User will get the following error:

```
{
errorCode: "405",
errorMessage: "Operation not allowed. An import operation is in-progress. Please wait till it completes",
httpStatus: "METHOD_NOT_ALLOWED",
      path: “/onc-devicemanager-service/v2/resync/devices”
      }
```

2. User can not trigger resync devices if collection is in progress for any of the device. User will get the following error:

```
{
 errorCode: "405",
 errorMessage: "Collection Already in progress"
 httpStatus: "METHOD_NOT_ALLOWED",
 path: "/onc-devicemanager-service/v2/resync/devices"
}
```

3. User can not trigger resync devices if collection is in progress for any of the device. User will get the following error:

```
{
 errorCode: "405",
 errorMessage: "Deletion_in_progress",
 httpStatus: "METHOD_NOT_ALLOWED"
 path: "/onc-devicemanager-service/v2/resync/devices"
 }
```

4. User can not trigger resync devices if any of the device name length is more than 128 characters

```
{
 "errorCode":"400",
 "errorMessage": "Input Length Should be less than 128 characters,
"path":"/onc-devicemanager  service/v2/devices/Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1",
 "httpStatus": "BAD_REQUEST"
}
```

5. When the device in the list on which resync has been triggered does not exist.

```
{
 errorCode: "404",
 errorMessage: "Device Does not Exists",
 httpStatus: "NOT_FOUND",
 path: "/onc-devicemanager-service/v2/devices/vxr59_site1"
}
```

Update Device

Description: This API is used to update the device.
URL: onc-devicemanager-service/v2/devices/{deviceName}
Type: PUT
Inputs:
Parameters: deviceName
Request Body: {"owner":"admin","password”:” cisco123”}
Response:

```
{
    "deviceId": "34b4f03c-fff4-3505-be04-f89ed7669b56",
    "username": "cisco",
  "password":
"AAAADNMq89Q01c7UPEi8rPD/miS5HsEnrkKx+wSARtQdVkB/2KhyDvE=",
    "ipAddress": "10.64.98.19",
   "port": 62769,
  "name": "vxrSite_3",
  "type": "NCS1010",
```

"grpcPort": 64634,
"statusCode": 200,
"status": null,
"importId": null,
"collectorStatus": "Waiting_for_connection",
"deployerStatus": "Waiting_for_connection",
"serviceStates": null,
"owner": null,
"creationTime": null,
"siteName": "site_vxrSite_3",
"lastUpdated": "2023-04-21T11:29:31.516446",
"lockStatus": "UNLOCK",
"message": "Device Modification Requested",
"version": null,
"protocol": "NETCONF",
"state": "ENABLED",
"proxyAddress": null,
"tl1Transport": null,
"deviceVersion": "0",
"deleteAcks": null,
"lastSuccessfulCollectionTime": 1682069671885
}

Error conditions:

- User cannot update device if device name length is more than 128 characters.

{
"errorCode":"400",
"errorMessage": "Input Length Should be less than 128 characters,
"path":"/onc-devicemanager-service/v2/devices/
Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1
Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1Site_1
Site_1Site_1Site_1Site_1Site_1Site_1Site_1",
"httpStatus": "BAD_REQUEST"
}

- User cannot update device if an import operation is already in progress. User will get the following error:

{
“errorCode”: "405",
errorMessage: "Operation not allowed. An import operation is in-progress. Please wait till it completes",
httpStatus: "METHOD_NOT_ALLOWED",
path: “/onc-devicemanager-service/v2/resync/devices”
}

- User will get the following error if the updated device name already exists in db.

{
"errorCode": "409",

```
    "errorMessage": "Device Already Exists",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "CONFLICT"
}
```


- User will get the following error if the updated ip address is invalid.


```
{
    "errorCode": "400",
    "errorMessage": "Invalid Ip Address format",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "BAD_REQUEST"
}
```


- User will get the following error if the updated port is invalid.


```
{
    "errorCode": "400",
    "errorMessage": "Invalid port number",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "BAD_REQUEST"
}
```


- User will get the following error if the updated grpc port is invalid.


```
{
    "errorCode": "400",
    "errorMessage": "Invalid grpc port number",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "BAD_REQUEST"
}
```


- User will get the following error if the updated connectivity timeout is not between 60 - 300 seconds


```
{
    "errorCode": "400",
    "errorMessage": "Invalid connectivity timeout value. Please provide value between 60-300 sec-
onds",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "BAD_REQUEST"
}
```


- User will get the following error if the updated connectivity timeout is not between 60 - 300 seconds


```
{
    "errorCode": "400",
    "errorMessage": "Invalid connectivity timeout value. Please provide value between 60-300 sec-
onds",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "BAD_REQUEST"
}
```


- User will get the following error if the device name, device owner or site name is not valid


```
{
    "errorCode": "400",
```

```
    "errorMessage": " Invalid Characters length or Invalid Input",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "BAD_REQUEST"
}
```


- User will get the following error if the username, or password maximum length is more than 128 characters.


```
{
    "errorCode": "400",
    "errorMessage": " Invalid Characters length or Invalid Input",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "BAD_REQUEST"
}
```


- User will get the following error if protocol is not supported.


```
{
    "errorCode": "500",
    "errorMessage": "Protocol provided is not supported. Accepted values are: [NETCONF, GRPC, TL1]",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "INTERNAL_SERVER_ERROR"
}
```


- User will get the following error if the device name which is going to be update does not exist in db.


```
{
    "errorCode": "404",
    "errorMessage": "Device site1 not found",
    "path": "/onc-devicemanager-service/v2/devices/4343",
    "httpStatus": "NOT_FOUND"
}
```


- User cannot update the device if device deletion is already in progress. User will get the following error


```
        {
         errorCode: "405",
    errorMessage: "Device is marked for Deletion. Cannot update device: site1",
        httpStatus: "METHOD_NOT_ALLOWED",
     path: "/onc-devicemanager-service/v2/devices/vxr59_site1"
         }
```


- User will get the following error if the device name could not be updated due to internal server error.


```
{
    "errorCode": "500",
    "errorMessage": "Internal Server Error: device with name site1 could not be updated",
    "path": "/onc-devicemanager-service/v2/devices/vxrSite_5",
    "httpStatus": "INTERNAL_SERVER_ERROR"
}
```

Resync All

Description: This API is used to trigger resync for all the devices that were onboarded

URL: /api/one-devicemanager-service/v2/devices/resyncAll

Type: Post

Request Body: {}

Response: Resync will be triggered for Device Ready, Disconnected and Resync failed Devices

Error conditions:

1. If the resync operation is in progress for any of the devices and try to trigger re-sync all, then we will get an error as shown below

Error Message: “Operation not allowed. An Import operation is in-progress. Please wait till it completes”

Error Code: 405

Full Network Sync

Description:

This API is used to enable/disable periodic full network sync for all the devices

URL: /api/one-devicemanager-service/v2/devices/fullNetworkSync

Type: Post

Input:

Request Payload:

{

enabled: true

fixedRate: 4

scheduledStartTime: "00:00:00"

}

Response:

1. Full network sync Sync initiated for all devices ----> for successful full network sync enable

2. Full network sync Sync disabled à for successful full network sync disable

Error conditions:

If the scheduledStartTime is not valid, we will get the below error

{

“errorCode”: “400”,

“errorMessage”: “Invalid Scheduled Start Time”

"path": "/devicemanager/v2/devices/fullNetworkSync",

"httpStatus": "BAD_REQUEST"
}
fixedRate should be between 1 and 23 otherwise we will get below error
{
“errorCode”: “400”,
“errorMessage”: “Fixed rate should be between 1-23”
"path": "/devicemanager/v2/devices/fullNetworkSync",
"httpStatus": "BAD_REQUEST"
}

Fetch Full Network Sync details (Full network sync status)

Description:

This api is used to fetch the information on FullNetworkSync. It has details like the scheduled start time of the full network sync, last run date of full network sync.
URL: /api/onc-devicemanager-service/v2/devices/getFullNetworkSyncDetails
Type: GET
Inputs:
Request Body: NA
Response:
{
"fullNetworkSyncId": "FullNetworkSync",
"scheduledStartTime": "2023-02-15 00:00:00",
"fixedDelay": 60,
"lastRun": null,
"recordVersion": 6,
"enabled": true
}
Error conditions:

- 500 - Internal server error

Node Setup

Import ANS File

Description:
This API is used to import the CONP generated NCMS based ANS file. The required information is input in the json file with NCMS based schema and then its fed into the ONC System.
URL: onc-nodesetup-service/api/ansPush/file/networkObject
Type : POST
Inputs :
Request Body:
CONP_ANS.json

Response:
networkObject imported successfully with uId: f760be17-c1c4-4d1b-8d88-e513c6e87852

Error conditions:

- If the input file format is not json, the user will get the following error : Invalid File type , Please provide valid ANS json File
- If the mandatory parameters are missing in the json input file, the user will get the following error:

```
{
 "errorCode": "400",
  "errorMessage": "Invalid Site details",
  "path": "/Nodesetupservice/api/ansPush/file/networkObject",
  "httpStatus": "BAD_REQUEST"
}
```

Below the mandatory parameters and the respective error messages:

- If Site label is null or empty, the user will get the following error : Invalid Site details Site Uid : 0eb0d583-1278-4fd4-ba1b-df836243b4d0
- If Site UID is null or empty, the user will get the following error: Invalid Site details
- For every location of ROADM Site type, if edgeSideUID is null or Empty, the user will get the following error: Invalid Location details Location Uid : dde740f2-8b9f-4050-9a3c-82ba644ede06
- If cardType is empty or null, the user wil get the following error : Invalid Card details Card Uid : card-f8ec647b-1d25-4ea2-bf6d-cd713bf0e1af
- If cardType is OLT/ILA and the card slot is missing, the user will get the error message : Invalid Card details Card Uid : card-f8ec647b-1d25-4ea2-bf6d-cd713bf0e1af
- If SiteLinks is empty or null, then the user will get the following error message : Invalid SiteLink details Link Id : 422ee48a-29af-43dc-b3c3-f4ceef8f0da6
- For ROADM Site type, If L0PhysicalSiteLink is empty, the user will get the error : Invalid SiteLink details Link Id : 422ee48a-29af-43dc-b3c3-f4ceef8f0da6

Associate Planned Devices

Description:
This API is used to associate the planned device to actual onboarded device in bulk. The Actual onboarded devices are listed to the user to choose one among them. This association is necessary step for performing bulk push operation.

URL: onc-nodesetup-service/api/ans/associate/device
Type : POST
Inputs :
Request Body:

```
[
 {
   "plannedDevice": "string",
   "plannedSite": "string",
   "actualDevice": "string",
   "actualSite": "string"
 }
]
```

Response:

```
[
  {
    "deviceData": {
      "plannedDevice": "B",
      "plannedSite": "Site-2",
      "actualDevice": "vxrSite_1",
      "actualSite": "site_vxrSite_1"
    },
    "response": {
      "httpCode": 200,
      "message": "Associated device successfully"
    }
  }
]
```

Error conditions:

- If actual device is not present in the onboarded devices list, the user will the following error : Device Not found
- If the equipment type of actual device and planned device are not matched, the user will get the following error:

```
[
 {
   "deviceData": {
     "plannedDevice": "A",
     "plannedSite": "Site-1",
```

```
      "actualDevice": "vxrSite_1",
      "actualSite": "site1"
    },
    "response": {
      "httpCode": 500,
      "message": "Equipment mismatched"
    }
  }
]
```


Bulk Provision

Description:
This API is used to bulk Provisioning of devices for different Config types such as Equipment, Connection, OpticalAttribute.

URL: onc-nodesetup-service/api/ans/provision
Type : POST
Inputs :
Request Body:


```
{
  "type": "string",
  "fileId": "string",
  "site": [
    {
      "plannedSite": "Site-2",
      "device": [
        {
          "plannedDevice": "A",
          "configType": [
            "EQUIPMENT","CONNECTION"
          ]
        }
      ]
    }
  ]
}
```


Response:


```
{
  "httpCode": 200,
  "message": "Push Config Initiated Successfully for: Site-2"
```

}

Error conditions:

- If the planned device is not already associated, then the user will get the error : Device not associated.

Get Planned Devices

Description:

This API is used to fetch the list of planned devices along with the details such as NetworkName, plannedDevice, plannedSite, actualDevice, actualSite, lastPushStatus, lastPushTimestamp, lastPlannedTimestamp.

URL: onc-nodesetup-service/api/ans/devices

Type : GET

Input:

Request Body :

Response:


```
{
   "plannedDevice": "A",
   "plannedSite": "Site-2",
   "actualDevice": "vxrSite_1",
   "actualSite": "site1",
   "lastPushStatus": "EQUIPMENT-FAILED,CONNECTION-FAILED",
   "lastPushTimestamp": 1682072631753,
   "auditLog": [
    {
      "id": "5cc649de-bd44-4303-9b4e-b9131aefa767",
      "info": "Equipment-0-NCS1K-OLT-C",
      "status": "SUCCESS",
      "errorMsg": null,
      "logTimestamp": 1682072367358
    },
    {
      "id": "c38ac446-643e-451a-9990-0a813541360e",
      "info": "Passive-0/1/0-NCS1K-BRK-8->0-Deg",
      "status": "FAILED",
      "errorMsg": "Slot provided in the request is not valid.",
      "logTimestamp": 1682072367358
    },
    {
      "id": "49b0b386-85eb-4e75-87b0-2ae6365199a1",
```

```
      "info": "Connection-0/0/0/28-NCS1K-OLT-C-0/1/0/8-NCS1K-BRK-8",
      "status": "FAILED",
      "errorMsg": "OCH port type is not allowed for non degree IPC ,Invalid port :0/1/0/8",
      "logTimestamp": 1682072367358
    },
    {
      "id": "8abeb800-e863-4b39-a184-2c8f3c196854",
      "info": "Equipment-0-NCS1K-OLT-C",
      "status": "SUCCESS",
      "errorMsg": null,
      "logTimestamp": 1682072634341
    },
    {
      "id": "0458572a-a2dd-43d1-9427-c6ad7624ec6c",
      "info": "Passive-0/1/0-NCS1K-BRK-8->0-Deg",
      "status": "FAILED",
      "errorMsg": "Slot provided in the request is not valid.",
      "logTimestamp": 1682072634341
    },
    {
      "id": "c7917ca1-2f66-4e13-be0e-9d18655dcb41",
      "info": "Connection-0/0/0/28-NCS1K-OLT-C-0/1/0/8-NCS1K-BRK-8",
      "status": "FAILED",
      "errorMsg": "OCH port type is not allowed for non-degree IPC ,Invalid port :0/1/0/8",
      "logTimestamp": 1682072634341
    }
  ],
  "networkName": "Network-21",
  "lastPlannedTimeStamp": 1682058735504
},
{
  "plannedDevice": "A",
  "plannedSite": "Site-3",
  "actualDevice": null,
  "actualSite": null,
  "lastPushStatus": null,
  "lastPushTimestamp": null,
  "auditLog": [],
  "networkName": "Network-21",
  "lastPlannedTimeStamp": 1682058735660
}]
```

Inventory

Rack view edit

Description: This api is used to relocate the equipments under the rack
URL: onc-inventory-service/api/v2/inventory/rack
Type: PUT
Inputs:
RequestBody:
site_vxrSite_1


```
[
   {
      "deviceNm": "vxrSite_1",
      "equipmentType": "Virtual Shelf",
      "currentRackNum": "21bfbd0e-90c6-3843-b107-c16ccaa72169",
      "newRackNum": "21bfbd0e-90c6-3843-b107-c16ccaa72169",
      "currentRuPos": 9,
      "newRuPos": 16,
      "shelfNumber": "3"
   }
]
```


Response:
Changes Updated Successfully

Error Condition:
i) In case of invalid data


```
{
  "errorCode": "400",
  "errorMessage": "Invalid Data ",
  "path": "/inventoryservice/api/v2/inventory/rack",
  "httpStatus": "BAD_REQUEST"
}
```


ii) In case of invalid SiteName


```
{
  "errorCode": "404",
  "errorMessage": "NetworkEntity not present for given siteName",
  "path": "/inventoryservice/api/v2/inventory/rack",
  "httpStatus": "NOT_FOUND"
}
```


Get inventory

Description: get inventory using deviced
URL: onc-inventory-service/api/v1/inventory/device/{deviceId}

Type: GET
Inputs:
RequestBody:
deviceId : "06ecb8d9-970b-3663-86fb-286a0d2c28fd"
Response:
success/failure

Error Condition
i) In case of invalid deviceId


```
{
  "errorCode": "404",
  "errorMessage": "Device ID is not in the correct format.",
  "path": "/inventoryservice/api/v1/inventory/device/5007bec0-9f84-345a-b8b8-
c8fa609644fc",
  "httpStatus": "NOT_FOUND"
}
```


ii) In case of null deviceId


```
{
  "errorCode": "404",
  "errorMessage": "Device ID cannot be null or empty.",
  "path": "/inventoryservice/api/v1/inventory/device/null",
  "httpStatus": "NOT_FOUND"
}
```


iii)


```
{
  "errorCode": "500",
  "errorMessage": "Exception while getting DeviceManager response : {}",
  "path": "/inventoryservice/api/v1/inventory/device/device",
  "httpStatus": "INTERNAL_SERVER_ERROR"
}
```


iv)


```
{
  "errorCode": "500",
  "errorMessage": "Response from collector service is null",
  "path": "/inventoryservice/api/v1/inventory/device/device",
  "httpStatus": "INTERNAL_SERVER_ERROR"
}
```


v)


```
{
```

```
  "errorCode": “404”,
  "errorMessage": "Protocol not Supported",
  "path": "/inventoryservice/api/v1/inventory/device/device",
  "httpStatus": "NOT_FOUND"
}
```


vi)


```
 {
  "errorCode": “500”,
  "errorMessage": "Response from Device Manager service is null",
  "path": "/inventoryservice/api/v1/inventory/device/device",
  "httpStatus": "INTERNAL_SERVER_ERROR"
}
```


vii)


```
{
  "errorCode": “404”,
  "errorMessage": "DeviceId is invalid. It cannot be null or empty in DeviceManager_Listen().",
  "path": "/inventoryservice/api/v1/inventory/device/device",
  "httpStatus": "NOT_FOUND"
}
```


Fetch Internal Patch Cord List

Description:

Api is used to fetch all l0PhysicalSite link object for the provided site

URL: onc-inventory-service/api/v2/inventory/ipc

Type : GET

Input:

```
{
        SiteName /*Mandatory*/
}
```

Ex: site_vxrSite_5

Request Body:

Resposne:

IPC List


```
[
{
"fromDevice": "vxrSite_5",
"fromPort": "Ots0/0/0/4",
"toDevice": "vxrSite_5",
```

"toPort": "Oms0/2/0/8",
"ipcId": null,
"cvStatus": "Not Found",
"cvForwardStatus": "Not Found",
"cvReverseStatus": "Not Found",
"lastVerifyTime": "Not Found",
"lossStatus": "Not Found",
"lossLastRun": "Not Found",
"forwardPatchLoss": "Not Found",
"reversePatchLoss": "Not Found",
"highPowerThreshold": null,
"lowPowerThreshold": null,
"fromDeviceUID": "cc27220c-744d-35db-9f3d-ddbb1f2a2cf2",
"toDeviceUID": "cc27220c-744d-35db-9f3d-ddbb1f2a2cf2",
"cvForwardReason": "Not Found",
"cvReverseReason": "Not Found",
"plReason": "Not Found"
}
]

Error conditions:

1. When the siteName is not present in the inventory database.

{
"errorCode": "404",
"errorMessage": "No device is present with given Site Name",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "NOT_FOUND"
}

2. When the siteName is given as "null" or an empty string.

{
"errorCode": "400",
"errorMessage": "SiteName cannot be null or empty.",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

3. When the site is present in DB, but no IPC is present on the given site.

{
"errorCode": "404",
"errorMessage": "Site link not found in DB for given Site",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "NOT_FOUND"

```
}
```

4. When an internal server error occurs in case of IPC GET

```
{
  "errorCode": "500",
  "errorMessage": "Internal server error has occurred",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "INTERNAL_SERVER_ERROR"
}
```

Add Internal Patch Cord

Description:

This api is used to create new l0PhysicalSite link object for the provided src and dst ports.

URL: onc-inventory-service/api/v2/inventory/ipc

Type : *POST*

Inputs :

Request Body:

Sample Input:

```
{
    "dstDeviceId": "cc27220c-744d-35db-9f3d-ddbb1f2a2cf2", /*Mandatory*/
    "dstPort": "0/2/0/8", /*Mandatory*/
    "srcDeviceId": "cc27220c-744d-35db-9f3d-ddbb1f2a2cf2", /*Mandatory*/
    "srcPort": "0/0/0/4", /*Mandatory*/
    "tonePattern": [] /*Optional*/
}
```

Response :

Success scenario output:

IPC created successfully with UID: {L0PhysicalSiteLink UIDs}

Error Scenarios:

1. When source or destination deviceID is not in correct format.

```
{
  "errorCode": "400",
  "errorMessage": "Device ID is not in correct format",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```

2. When source device is not present in db

```
{
  "errorCode": "404",
  "errorMessage": "IPC creation not possible, src device details not found.",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "NOT_FOUND"
```

}

3. When destination device is not present in db
{
"errorCode": "404",
"errorMessage": "IPC creation not possible, dst device details not found.",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "NOT_FOUND"
}

4. When null is given as input in place of ipcData
{
"errorCode": "500",
"errorMessage": "Required request body is missing: public
org.springframework.http.ResponseEntity<java.lang.String>
com.cisco.inventory.controller.InventoryController.createIPC(com.cisco.inventory.model.IPC
RequestData)",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "INTERNAL_SERVER_ERROR"
}

5.When srcDeviceId, dstDeviceId, srcPort, dstPort is null or empty.
{
"errorCode": "400",
"errorMessage": "IPC data parameters should not be null or empty.",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

6. When srcPort or dstPort is not in correct format.
{
"errorCode": "400",
"errorMessage": "Port is not in correct format",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

7. When the source and destination devices are under different sites.
{
"errorCode": "400",
"errorMessage": "IPC creation failed, as src and dst devices are under different sites.",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"

```
}
```

8. Device is not present in the db.

```
{
  "errorCode": "404",
  "errorMessage": "Network object not present for the given site UUID.",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "NOT_FOUND"
}
```

9. If source or destination port is not present in the source or destination device respectively

```
{
  "errorCode": "400",
  "errorMessage": "Invalid input port of device: {srcPort/dstPort}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```

10. When is IPC is created on ILA device

```
{
  "errorCode": "400",
  "errorMessage": "IPC creation failed, as src and dst are both same ILA devices",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```

11. When OCH port is given as input in non degree IPC scenario

```
{
  "errorCode": "400",
  "errorMessage": "OCH port type is not allowed for non-degree IPC ,Invalid port : {port-
Name}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```

12. When flex passive IPC creation is triggered with destination OMS port as a port other than the first OMS port.

```
{
  "errorCode": "400",
  "errorMessage": "IPC creation is allowed only on first OMS port of the MPO passive type
,Invalid port :{portName}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
```

}

13. When flex passive IPC creation is triggered with source OTS port as a port other than the first OTS port.
{
"errorCode": "400",
"errorMessage": "IPC creation is allowed only on first OTS port of the MPO type on OLT ,Invalid port :{portName}",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

14. For fixed passive IPC, if a port other than 2 or 3 is chosen as source port. (PortName - R/S/I/P -- If P value is other than 2/3)
{
"errorCode": "400",
"errorMessage": "Invalid port selected on OLT for a Fixed passive,Invalid port: {port-Name}",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

15. For flex passive IPC, if a port is 0,1,2,3 is chosen as source port. (PortName - R/S/I/P -- If P value is 0, 1, 2, 3)
{
"errorCode": "400",
"errorMessage": "Invalid port selected on OLT for a MPO passive type,Invalid port: {port-Name}",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

16. If the port mentioned is not present in the device in any card
{
"errorCode": "400",
"errorMessage": "Port is not valid ,port: {portName}",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

17. When both source and destination is given as OTS port
{
"errorCode": "400",

```
  "errorMessage": "Both ports cannot be on Same Equipment",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


18. When both source and destination is given as OMS port


```
{
  "errorCode": "400",
  "errorMessage": "Both ports cannot be of OMS type",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


19. During degree IPC creation, if source or destination port is not from a NCS1K-BRK-8 card


```
{
  "errorCode": "400",
  "errorMessage": "Src card or dst card are not of type BRK-8, degree IPC creation not possible",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


20. During degree IPC creation, if source or destination physical port is not of OCH-PORT type


```
{
  "errorCode": "400",
  "errorMessage": "Src or dst physical port are not of type OCH-PORT, degree IPC creation not possible",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


21. During degree IPC creation, if source or destination logical port is not present for the given port


```
{
  "errorCode": "400",
  "errorMessage": "Src or dst logical port are absent, degree IPC creation not possible",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```

22. During degree IPC creation, if source or destination logical port is not of INTERNAL port role


```
{
  "errorCode": "400",
  "errorMessage": "Src or dst logical port are not of role Internal, degree IPC creation not possible",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


23. During IPC creation, if tone pattern is given, and the tone pattern length is less than 8


```
{
  "errorCode": "400",
  "errorMessage": "Tone pattern {tonePattern} should be equal or greater than 8 characters",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


24. During degree IPC creation, if tone pattern is given, and the number of tone patterns is more than 1


```
{
  "errorCode": "400",
  "errorMessage": "Please provide only one tone pattern for degree IPC",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


25. During fixed passive IPC creation, if tone pattern is given, and the number of tone patterns given is more than 1


```
{
  "errorCode": "400",
  "errorMessage": "Please provide only one tone pattern for Fixed passive types",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


26. During IPC creation, if tone pattern is given, and the tone pattern has non hex characters.


```
{
  "errorCode": "400",
  "errorMessage": "Invalid hex value,valid value is 0-9 , a-f , A-F are allowed,wrong pattern :{tonePattern}",
```

"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

27. During flex passive IPC creation, if the passive type is NCS1K-BRK-16 or NCS1K-BRK-24 and if tone pattern is given.

{
"errorCode": "405",
"errorMessage": "Tone Pattern configuration is not allowed yet for BRK-24 and BRK-16",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "METHOD_NOT_ALLOWED"
}

27. During a flex passive on NCS1K-BRK-8 is created, tone pattern is mentioned, OTS port is 0/0/0/28 and less than 6 valid tone patterns are provided

{
"errorCode": "400",
"errorMessage": "IPC creation not possible, for Flex this port pair provide either no patterns or 6 patterns",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

28. During a flex passive on NCS1K-BRK-8 is created, tone pattern is mentioned, OTS port is NOT 0/0/0/28 and less than 8 valid tone patterns are provided

{
"errorCode": "400",
"errorMessage": "IPC creation not possible, for Flex this port pair provide either no patterns or 6 patterns",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

29. During a flex passive on NCS1K-BRK-8 is created, tone pattern is mentioned, OTS port is 0/0/0/28 and more than 6 valid tone patterns are provided

{
"errorCode": "400",
"errorMessage": "IPC creation not possible, for this port pair provide only 6 patterns",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

30. During a flex passive on NCS1K-BRK-8 is created, tone pattern is mentioned, OTS port is NOT 0/0/0/28 and more than 8 valid tone patterns are provided


```
{
  "errorCode": "400",
  "errorMessage": "IPC creation not possible, for this port pair provide only   8 patterns",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


31. During Degree IPC creation with tonePattern, if the existing tone doesnt match with requested tone


```
{
  "errorCode": "400",
  "errorMessage": "Existing tone: {Existing l0PhysicalSiteLink.linkId} is not matching with requested tone: {tonePattern}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


32. During IPC creation, if the device on which the port is present is not in site fetched from network object


```
{
  "errorCode": "400",
  "errorMessage": "Device: {deviceID} not present in site {siteUID}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


33. If an IPC already exists on either or both the given ports


```
{
  "errorCode": "400",
  "errorMessage": "IPC already present on one or both of the input port(s)",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


34. If source or destination device's hostname is not set


```
{
  "errorCode": "400",
  "errorMessage": "src or dst location hostname is not set",
  "path": "/inventoryservice/api/v2/inventory/ipc",
```

"httpStatus": "BAD_REQUEST"
}

35. During NonDegree IPC creation with tonePattern, if the existing tone doesnt match with requested tone

{
"errorCode": "404",
"errorMessage": "Existing tone: {Existing l0PhysicalSiteLink.linkId} is not matching with requested tone: {tonePattern}",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "NOT_FOUND"
}

Delete Internal Patch Cord

Description:

This api is used to delete the l0PhysicalSite link object for the provided src and dst ports

URL: onc-inventory-service/api/v2/inventory/ipc

Type : DELETE

Inputs :

Request Body:

{
"dstDeviceId": "cc27220c-744d-35db-9f3d-ddbb1f2a2cf2", /*Mandatory*/
"dstPort": "0/2/0/8", /*Mandatory*/
"srcDeviceId": "cc27220c-744d-35db-9f3d-ddbb1f2a2cf2", /*Mandatory*/
"srcPort": "0/0/0/4", /*Mandatory*/
"tonePattern": [] /*Optional*/
}

Response :

Success scenario output:

IPCs with UIDs 1408fedc-ff00-37f3-a026-f2f672d4a0fe,62a2b81c-b3ab-37f3-8e3e-8f9ae795c549,c676f6a8-f1ae-312d-87c3-8537a04345ee,627a068f-6155-3828-b65f-90b154e5a134,cd1af5e0-3835-38c0-8a71-72838d93f377,2b18c2c0-46af-3cf2-9f53-d525807413c9,c6981b74-ad18-3e03-aaa3-a7feb6ea3309,4474c4a5-3e6d-33eb-8279-8a737ead16c5 have been deleted

Error Scenarios:

1. When the IPCList is given as null or an empty IPC list is given as input

{
"errorCode": "400",
"errorMessage": "IpcList cannot be null or empty",
"path": "/inventoryservice/api/v2/inventory/ipc",
"httpStatus": "BAD_REQUEST"
}

2. If the source or destination device is not present in DB


```
{
  "errorCode": "404",
  "errorMessage": "No device found with given UID:  {srcDeviceId/dstDeviceId}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "NOT_FOUND"
}
```


3. If the source and destination are of different types


```
{
  "errorCode": "400",
  "errorMessage": "IPC deletion failed, as src and dst devices are of different type.",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


4. If the destination device is not present in the same site as source device


```
{
  "errorCode": "400",
  "errorMessage": "Device: {deviceId} not present in site {siteUID}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


5. If source or destination port is not present in the source or destination device respectively


```
{
  "errorCode": "400",
  "errorMessage": "Invalid input port of device: {srcPort/dstPort}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


6. When an invalid port is given


```
{
  "errorCode": "400",
  "errorMessage": "Port is not  valid ,port: {portName}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}
```


7. When both source and destination port are both of OMS type or both are of OTS type


```
{
  "errorCode": "400",
  "errorMessage": "IPC not supported on selected ports,cannot proceed with deletion",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
```

}

8. During fixed IPC deletion, if the port is not a valid fixed IPC OTS port
{
  "errorCode": "400",
  "errorMessage": "IPC not supported on selected ports,cannot proceed with deletion",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}

9. During flex IPC deletion, if the port is not a valid flex IPC OTS port
{
  "errorCode": "400",
  "errorMessage": "IPC not supported on selected ports,cannot proceed with deletion",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}

10. When flex passive IPC deletion is triggered with destination OMS port as a port other than the first OMS port.
{
  "errorCode": "400",
  "errorMessage": "IPC Deletion is allowed only on first OMS port of the MPO passive type ,Invalid port :{portName}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}

11. When flex passive IPC deletion is triggered with source OTS port as a port other than the first OTS port.
{
  "errorCode": "400",
  "errorMessage": "IPC Deletion is allowed only on first OTS port of the MPO type on OLT ,Invalid port :{portName}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "BAD_REQUEST"
}

12. If a circuit spans across the devices through the mentioned ports of the IPC
{
  "errorCode": "405",
  "errorMessage": "IPC Deletion not possible as Crossconnect exist on IPC, Crossconnect UID: {crossConnectUID}",
  "path": "/inventoryservice/api/v2/inventory/ipc",
  "httpStatus": "METHOD_NOT_ALLOWED"
}

Trigger connection verification

Description:

This API is used to Trigger connection verification on an Internal Patch Card.
URL: inventoryservice/api/v1/siteconnectionslib/connectionverification
Type: POST
Inputs:

```
{
   "InputType": "IPC",
   "Inputs": [
      {
         "srcSiteName": "site_vxrSite_1",
         "srcDeviceName": "r1",
         "srcPortName": "Ots0/0/0/2",
         "dstSiteName": "site_vxrSite_1",
         "dstDeviceName": "r1",
         "dstPortName": "Oms0/1/0/32"
      }
   ]
}
```

Request Body: NA
Response:
"success"
Error conditions:

- If device or site name,port name is wrong, then we will get error as below:

```
{
  "errorCode": "412",
  "errorMessage": "Input not correct",
  "path": "/inventoryservice/api/v1/siteconnectionslib/connectionverification",
  "httpStatus": "PRECONDITION_FAILED"
}
```

Trigger Patch loss

Description:

This API is used to Trigger Patch loss on an IPC .
URL: /api/v1/siteconnectionslib/patchloss
Type: POST
Inputs:

```
{
   "InputType": "IPC",
   "Inputs": [
      {
         "srcSiteName": "site_vxrSite_1",
         "srcDeviceName": "r1",
         "srcPortName": "Ots0/0/0/2",
         "dstSiteName": "site_vxrSite_1",
         "dstDeviceName": "r1",
         "dstPortName": "Oms0/1/0/32"
      }
   ]
}
```


Request Body: NA

Response: "success"

Error conditions:

- If device or site name,port name is wrong, then we will get error as below:


```
{
  "errorCode": "412",
  "errorMessage": "Input not correct",
  "path": "/inventoryservice/api/v1/siteconnectionslib/connectionverification",
  "httpStatus": "PRECONDITION_FAILED"
}
```


Get SiteType

Description: Api is used to fetch all site's type details
URL: onc-inventory-service/api/v2/inventory/siteType
Type: GET
Inputs:
RequestBody: NA
Response:


```
[
 {
   "siteName": "<site name>",
   "siteType": "<roadm/ola>",
   "devices": [
     "<device name1>",
     “<device name2>
   ],
   "extendedTypeDescription": {
     "degrees": <number of degrees the site consist of>,
```

```
    "omnidegrees": "",
    "transponder": "",
    "aggregationType": ""
   }
  }
]
```


Error Conditions:

If no site is present


```
{
 "errorCode": "404",
 "errorMessage": "SiteTypeDeviceDetails not present for given siteName",
 "path": "/inventoryservice/api/v2/inventory/siteType",
 "httpStatus": "NOT_FOUND"
}
```


Get Site Details by site name

Description: Api is used to fetch site details of the input site name
URL: onc-inventory-service/api/v2/inventory/site/{siteName}
Type: GET
Inputs:
RequestBody: site_vxrSite_1
Response:


```
{
 "id": null,
 "entityType": "Site",
 "UID": "<site unique identifier>",
 "hierarchicalUID": "",
 "label": "<site name>",
 "type": "<roadm/ola>",
 "preEquipDegree": null,
 "systemLabel": "<site name>",
 "xCoordinate": null,
 "yCoordinate": null,
 "businessStatus": "FUTURE",
 "siteAddress": null,
 "mask": null,
 "gateway": null,
 "softwareVersion": null,
```

```
  "CLLICode": null,
  "networkStatus": "DISCOVERED",
  "sitePositionLock": null,
  "equipment": {
    "location": [
      {"_comments": "<location details for all the devices under the site. This includes rack >
shelf, passive > card > port>"}
    ],
    "unlocked": [],
    "forced": []
  },
  "logicalSites": {
    "L0Site": [
      {
        "id": null,
        "entityType": "L0Site",
        "UID": "<L0Site unique identifier>",
        "hierarchicalUID": "l0Site-928e70d0-c7cc-367f-80cd-20ae56423cb7",
        "label": "<<roadm/ola>_<sitename>>",
        "L0IP": "",
        "L0SiteType": "ola",
        "siteAddress": null,
        "businessStatus": "FUTURE",
        "networkStatus": "DISCOVERED",
        "additionalProperties": null,
        "edgesSides": [
          {"_comments": "degree details of the device under the site”}
        ],
        "DWDMBOM": [],
        "regenGroups": [],
        "terminationPoints": [],
        "nodeTermPoints": [
          {"_comments": "service interface point details of the device under the site"}
        ],
        "L0DesignData": null,
        "L0OutputData": null,
        "L0Parameters": [
          {"_comments": "port optical parameters of the device under the site"}
        ],
        "unlocked": [],
        "forced": [],
        "total": null,
        "isSSON": null
      }
```

```
    ],
    "OTN": [],
    "NCS1K": [],
    "SONETSDH": [],
    "L3": [],
    "additionalProperties": null,
    "pktPDH": [],
    "pktEthernet": [],
    "pktSONET": [],
    "unlocked": [],
    "forced": []
  },
  "unlocked": [],
  "forced": [],
  "total": null,
  "isSSON": null
}
```


Error Conditions:
i). Incorrect siteName


```
 {
  "errorCode": "404",
  "errorMessage": "NetworkEntity not present for given siteName",
  "path": "/inventoryservice/api/v2/inventory/site/R1",
  "httpStatus": "NOT_FOUND"
}
```


ii). Null or empty siteName


```
{
  "errorCode": "400",
  "errorMessage": "SiteName cannot be null or empty.",
  "path": "/inventoryservice/api/v2/inventory/site/null",
  "httpStatus": "BAD_REQUEST"
}
```


Add passive

Description: This api is used to pre -provision a passive with a mentioned type in the specified slot for a given device
URL: onc-inventory-service/api/v2/inventory/passive/provision
Type: POST
Inputs:
RequestBody:


```
{
```

```
    "deviceId": "39fe18c5-aa81-31c1-80f5-1da1865be23c",
    "type": "NCS1K-BRK-8",
    "slot": "0/1",
    "requestId": "39fe18c5-aa81-31c1-80f5-1da1865be23c"
}
```


Response:

Passive pre provision request received, a passive unit of the given type will be provisioned in the mentioned slot

Error Condition:

i) In case of invalid deviceId


```
{
  "errorCode": "400",
  "errorMessage": "[Invalid UID]",
  "path": "/inventoryservice/api/v2/inventory/passive/provision",
  "httpStatus": "BAD_REQUEST"
}
```


ii). In case of invalid slot


```
{
  "errorCode": "400",
  "errorMessage": "Slot provided in the request is not valid.",
  "path": "/inventoryservice/api/v2/inventory/passive/provision",
  "httpStatus": "BAD_REQUEST"
}
```


iii). In case of invalid passive type


```
{
  "errorCode": "400",
  "errorMessage": "Passive type provided in the request is not valid.",
  "path": "/inventoryservice/api/v2/inventory/passive/provision",
  "httpStatus": "BAD_REQUEST"
}
```


iv). In case of missing passive type


```
{
  "errorCode": "400",
  "errorMessage": "[Passive type cannot be blank]",
  "path": "/inventoryservice/api/v2/inventory/passive/provision",
  "httpStatus": "BAD_REQUEST"
}
```


v). In case of missing slot


```
{
```

```
  "errorCode": "400",
  "errorMessage": "[Passive slot cannot be blank]",
  "path": "/inventoryservice/api/v2/inventory/passive/provision",
  "httpStatus": "BAD_REQUEST"
}
```


vi). In case of slot already being occupied


```
{
  "errorCode": "400",
  "errorMessage": "The slot provided for passive pre-provision is not empty",
  "path": "/inventoryservice/api/v2/inventory/passive/provision",
  "httpStatus": "BAD_REQUEST"
}
```


vii). In case of provisioning a same passive


```
 {
  "errorCode": "409”,
  "errorMessage": "Same type of passive is already present in given slot",
  "path": "/inventoryservice/api/v2/inventory/passive/provision",
  "httpStatus": “CONFLICT”
}
```


Delete passive

Description: Api is used to delete a pre provisioned passive if there is no real passive plugged in the given slot. Only pre-provisioned passive will be deleted.
URL: onc-inventory-service/api/v2/inventory/passive/delete
Type: DELETE
Inputs:
RequestBody:


```
{
    "deviceID": "cc27220c-744d-35db-9f3d-ddbb1f2a2cf2",
    "passiveSlot": "0/3"
}
```


Response:
Pre provision delete accepted

Error Conditions:
i). In case of invalid deviceId


```
 {
  "errorCode": "400",
  "errorMessage": "[Invalid UID]",
  "path": "/inventoryservice/api/v2/inventory/passive/delete",
  "httpStatus": "BAD_REQUEST"
```

}

ii). In case of SVO DeviceId

{
"errorCode": "400",
"errorMessage": "Passive Delete not supported for a given device type SVO",
"path": "/inventoryservice/api/v2/inventory/passive/delete",
"httpStatus": "BAD_REQUEST"
}

iii). In case of incorrect deviceId

{
"errorCode": "404",
"errorMessage": "Device details doesn't exist 8bc01229-246d-37c9-aaf8-5dbba7ebdb0c",
"path": "/inventoryservice/api/v2/inventory/passive/delete",
"httpStatus": "NOT_FOUND"
}

iv). In case of invalid slot

{
"errorCode": "400",
"errorMessage": "Invalid slot",
"path": "/inventoryservice/api/v2/inventory/passive/delete",
"httpStatus": "BAD_REQUEST"
}

v). Trying to delete a non-pre-provisioned passive

{
"errorCode": "400",
"errorMessage": "Only pre-provisioned passives can be deleted",
"path": "/inventoryservice/api/v2/inventory/passive/delete",
"httpStatus": "BAD_REQUEST"
}

vi). Trying to delete a flex passive under holder

{
"errorCode": "400",
"errorMessage": "Flex passive under holder isn't supported",
"path": "/inventoryservice/api/v2/inventory/passive/delete",
"httpStatus": "BAD_REQUEST"
}

Performance Monitoring

Fetching PM information

Description:

This API is used to Fetch the power data for the list of devices with sub list of specified interfaces in api

URL: onc-pm-service/v1/current-pm/interfaces

Type: POST

Request Body:


```
[
{
   "deviceNm": "vxrSite_1",
   "interfaceList": [
     "Och0/1/0/0"
   ]
 }
]
```


Success Response:-


```
[
 {
   "deviceNm": "vxrSite_1",
   "pmResponses": [
    {
      "interfaceName": "0/1/0/0",
      "pmValues": [
       {
         "attrName": "receivedPower",
         "attrValue": "0.0",
         "attrUnit": "dB"
       },
       {
         "attrName": "transmittedPower",
         "attrValue": "-50.0",
         "attrUnit": "dB"
       }
      ],
      "errorMessage": null
    }
   ]
```

```
  }
]
```


Failure Response: -


```
[
  {
    "deviceNm": "vxrSite_1",
    "pmResponses": [
      {
        "interfaceName": "0/1/0/0",
        "pmValues": [],
        "errorMessage": "Internal Server Error"
      }
    ]
  }
]
```


Error Conditions:-

1.Pm Current Interface will support only Ots,Och,Ots-Och entites for fetching the power.

2. If th interface name does not contains without the entity name than it will throw the **“**Internal Server Error**”** with error code as 200 with empty power data.

3.If the device name does not exist, then it will throw the **“**DeviceName not found**”** and device name is empty then it will throw the “EntityType not found**”** with error code as 200 with empty power data.

Alarms

Fetch List Of Alarms

Description:
This API fetches list of active alarms for a given Site. The response will contain alarm description, severity, location etc.
URL: onc-alarm-service/v1/alarms?siteName=Site_1
Type: GET
Inputs:
Query Param: SiteName
Request Body: NA
Response:

[
{
"id": "dff9aeee-7fef-468d-9e72-490d9a82c66b",
"deviceId": "dca671ee-17d1-33df-97c2-e97d60d59ae3",
"deviceName": "Site_1",
"severity": "critical",
"alarmDescription": "Loss of Signal - Payload",
"location": "0/1",
"alarmTime": "2023-04-05 07:59:01.0",
"alarmInterface": "Och0/1/0/0",
"hostName": null,
"alarmName": "RX-LOS-P"
}]

Audit Log

Fetch Filtered Audit Log

Description:
This API fetches Audit Log based on the Log Category.
URL: onc-torch-service/v1/filteredAuditLog
Type: GET
Inputs:
Query Param:
Request :
torchservice/v1/filteredAuditLog?page=0&size=1&sort=asc&logCategory=SECURITY'
Pageable : {
"page": 0,
"size": 1,
"sort": ["asc"]
}
LogCategory : [SECURITY, DEVICES, INVENTORY, POWER_METRICS, PLANNING_IMPORT, ALIEN_IMPORT, XC_MANAGEMENT]
Response :


```
{ "content": [
   {
    "eventType": "LOGIN",
    "description": "User logged in successfully",
    "clientIp": "10.42.0.1",
    "date": "2023-04-06T11:40:00.700356",
    "userName": "ADMIN",
    "logInfo": "SUCCESS",
    "id": "77864c99-26d0-4861-885e-bca5285068ed",
    "srcService": "usermanagementservice"
   }
  ]
}
```


Fetch Audit Log

Description:
This API fetched Audit Log for all Log Category.
URL: onc-torch-service/v1/auditLog
Type: GET

Inputs:
Query Param:
Request Body : torchservice/v1/auditLog?page=0&size=1&sort=asc
Pageable : {
"page": 0,
"size": 1,
"sort": ["asc"]
}
Response :

{ "content": [
{
"eventType": "LOGIN",
"description": "User logged in successfully",
"clientIp": "10.42.0.1",
"date": "2023-04-06T11:40:00.700356",
"userName": "ADMIN",
"logInfo": "SUCCESS",
"id": "77864c99-26d0-4861-885e-bca5285068ed",
"srcService": "usermanagementservice"
}
]
}

Circuits

Fetch Cross Connects

Description: The API is used to get the list of Cross connects available in Provided Site.
URL: circuitservice/v1/circuit/siteName/{siteName}/crossConnects
Type: GET
Input: vxrSite_1
Request Body: NA
Response:


```
[
 {
  "hierarchicalUID": "CROSSCONNECT_6391ac1a-b43c-3114-be55-
8f798a2df3cf_7e2ce432ff833d31a88776b567a8edf71bea6159d1c2ac3a3d48fb05d
88994c1_OCH-CHANNEL",
  "connectionCircuitId": null,
  "protocolLayer": "OCH-CHANNEL",
  "srcDeviceUID": "6391ac1a-b43c-3114-be55-8f798a2df3cf",
  "dstDeviceUID": "6391ac1a-b43c-3114-be55-8f798a2df3cf",
  "path": [
   "{\"srcIf\":\"0/0/0/0\"}"
  ],
  "status": null,
  "type": null,
  "oduCrossConnectProps": null,
  "ochCrossConnectProps": {
   "connectionName": null,
   "circuitId": null,
   "adminState": "UNLOCKED",
   "serviceState": "ENABLED",
   "lifeCycleState": "INSTALLED",
   "centralFrequency": 194.875,
   "allocationWidth": 75,
   "signalWidth": null,
   "channelOxc": null,
   "carrierOxcs": [
    "89baf9dd-011e-37a0-9038-c99b82f0c971"
   ],
   "internalPaths": [],
   "channelPorts": [],
```

```
      "oxctype": "bidirectional"
    },
    "id": null,
    "entityPath": "networks[0]/links/siteLinks[e26017d1-b740-330f-93dd-
ca642f3373f4]/logical/crossConnect[51578891-46da-3068-a7d6-
f7e235b8e186]",
    "entityType": "CrossConnect",
    "siteUID": "297e3704-e69e-3580-89c6-36a4b79ce7e6",
    "deviceId": "6391ac1a-b43c-3114-be55-8f798a2df3cf",
    "connectionServiceId": null,
    "portDescriptor": null,
    "isMainCrossConnect": true,
    "version": 0,
    "isLogical": true,
    "subCrossConnect": [],
    "srcDeviceName": "vxrSite_1",
    "dstDeviceName": "vxrSite_1",
    "passiveType": null,
    "crossConnectType": "TERMINAL",
    "deleteStatus": null,
    "upgradable": false,
    "uid": "51578891-46da-3068-a7d6-f7e235b8e186"
  },
  {
    "hierarchicalUID": "CROSSCONNECT_6391ac1a-b43c-3114-be55-
8f798a2df3cf_194.875",
    "connectionCircuitId": null,
    "protocolLayer": "OCH-CARRIER",
    "srcDeviceUID": "6391ac1a-b43c-3114-be55-8f798a2df3cf",
    "dstDeviceUID": "6391ac1a-b43c-3114-be55-8f798a2df3cf",
    "path": [
      "{\"srcIf\":\"0/0/0/0\",\"dstIf\":\"0/2/1/0\"}"
    ],
    "status": "discovered",
    "type": null,
    "oduCrossConnectProps": null,
    "ochCrossConnectProps": {
      "connectionName": "onc_TqeZk281YMtDp83s9ywYVxwDm",
      "circuitId": null,
      "adminState": "UNLOCKED",
```

```
  "serviceState": "ENABLED",
  "lifeCycleState": "INSTALLED",
  "centralFrequency": 194.875,
  "allocationWidth": 75,
  "signalWidth": 75,
  "channelOxc": "51578891-46da-3068-a7d6-f7e235b8e186",
  "carrierOxcs": [],
  "internalPaths": [
   {
    "directionType": null,
    "path": [
     "port-6391ac1a-b43c-3114-be55-8f798a2df3cf-0/0/0/0",
     "port-6391ac1a-b43c-3114-be55-8f798a2df3cf-0/0/0/12",
     "port-6391ac1a-b43c-3114-be55-8f798a2df3cf-0/2/1/24",
     "port-6391ac1a-b43c-3114-be55-8f798a2df3cf-0/2/1/0"
    ]
   }
  ],
  "channelPorts": [
   {
    "name": "Ots-Och0/0/0/0/17",
    "type": "LINECHANNEL",
    "adminState": "UNLOCKED",
    "serviceState": "ENABLED",
    "channelId": 17,
    "physicalPortUid": "33745933-31d1-320c-9a8e-4ba91630d1e7"
   },
   {
    "name": "Ots-Och0/0/0/12/17",
    "type": "ADDDROPCHANNEL",
    "adminState": "UNLOCKED",
    "serviceState": "ENABLED",
    "channelId": 17,
    "physicalPortUid": "e5519329-0094-31fe-9ecb-dc08a8cbd75b"
   }
  ],
  "oxctype": "bidirectional"
 },
 "id": null,
```

```
    "entityPath": "networks[0]/links/siteLinks[e26017d1-b740-330f-93dd-
ca642f3373f4]/logical/crossConnect[89baf9dd-011e-37a0-9038-c99b82f0c971]",
    "entityType": "CrossConnect",
    "siteUID": "297e3704-e69e-3580-89c6-36a4b79ce7e6",
    "deviceId": "6391ac1a-b43c-3114-be55-8f798a2df3cf",
    "connectionServiceId": null,
    "portDescriptor": null,
    "isMainCrossConnect": true,
    "version": 2,
    "isLogical": false,
    "subCrossConnect": [],
    "srcDeviceName": "vxrSite_1",
    "dstDeviceName": "vxrSite_1",
    "passiveType": "NCS1K-BRK-24",
    "crossConnectType": "TERMINAL",
    "deleteStatus": null,
    "upgradable": true,
    "uid": "89baf9dd-011e-37a0-9038-c99b82f0c971"
  }
]
```


Input: Site_2
Error Condition: When provided siteName is not found.
Error Response:


```
{
  "errorCode": "400",
  "errorMessage": "Site not found ",
  "path": "/circuitservice/v1/circuit/siteName/vxrSite/crossConnects",
  "httpStatus": "BAD_REQUEST"
}
```


Input: Site_5
Response: [] -> If there are no cross connects for given Site

Deletion of Cross Connect

Description: The API is used to delete the Single Cross connect which are in Partial or PendingRemoval state using Crossconnect Uid.
URL: circuitservice/v1/crossConnects/{crossConnectUID}

Type: DELETE
Input: 89baf9dd-011e-37a0-9038-c99b82f0c971
Request Body: NA
Response:
Case-1: Cross connect Uid delete success scenario.
CROSSCONNECT-89baf9dd-011e-37a0-9038-c99b82f0c971 deleted successfully

Error Condition: Provided cross connect Uid is spanning under circuit, so cross connect deletion is not possible.
Error Response:

```
{
  "errorCode": "500",
  "errorMessage": "CROSSCONNECT-48771f09-4957-3602-bb46-5a2a8b1b065c cannot be deleted as it is part of Circuit",
  "path": "/circuitservice/v1/crossConnects/48771f09-4957-3602-bb46-5a2a8b1b065c",
  "httpStatus": "INTERNAL_SERVER_ERROR"
}
```

Error Condition: Provided cross connect Uid is not available under Site.
Error Response:

```
{
  "errorCode": "404",
  "errorMessage": "No crossconnects found!",
  "path": "/circuitservice/v1/crossConnects/48771f09-4957-3602-bb46-5a2a8b1b065d",
  "httpStatus": "NOT_FOUND"
}
```

Mapping Ingress Passive to Egress Passive

Description: The API is used to Map Ingress to Egress for circuit creation of Brownfield.
URL: circuitservice/v1/circuits/PassiveIngressToEgressMapping
Type: POST
Input:
Condition: For cross connect after internal patch card creation success and while trying to map Ingress passive to Egress Passive in Success scenario
Request Body:

```
{
  "crossconnectId": "463c2c7c-1bf9-3b08-9e62-1247be35a872",
```

"omsPort": "0/1/0/32",
"ochPort": "0/1/0/0",
"passiveTypes": "NCS1K_BRK_24"
}
Response:
Response Body: Success

Error Condition: When provided Passive ports are not valid mapping.
Request Body:
{
"crossconnectId": "89baf9dd-011e-37a0-9038-c99b82f0c972",
"omsPort": "0/2/1/24",
"ochPort": "0/2/1/21",
"passiveTypes": "NCS1K_BRK_24"
}
Error Responses:
{
"errorCode": "400",
"errorMessage": "Wrong Mapping between OMS port to OCH port for passive breakout failed",
"path": "/circuitservice/v1/circuits/PassiveIngressToEgressMapping",
"httpStatus": "BAD_REQUEST"
}
Error Condition: When Cross connect UID is in Discovered State or Provided cross connect UID not available under Device/Site
Request Body:
{
"crossconnectId": "89baf9dd-011e-37a0-9038-c99b82f0c972",
"omsPort": "0/2/1/24",
"ochPort": "0/2/1/0",
"passiveTypes": "NCS1K_BRK_24"
}
Error Responses:
{
"errorCode": "400",
"errorMessage": "Crossconnect not found or is in discovered state. Provide a valid CrossConnect Id",
"path": "/circuitservice/v1/circuits/PassiveIngressToEgressMapping",
"httpStatus": "BAD_REQUEST"
}

Error Condition: Without IPC creation trying to Map Igress passive to Egress Passive
Request Body:


```
{
  "crossconnectId": "b4052793-7dae-34e5-9305-e00c4ccd1d0d",
  "omsPort": "0/2/1/24",
  "ochPort": "0/2/1/0",
  "passiveTypes": "NCS1K_BRK_24"
}
```


Error Responses:


```
{
  "errorCode": "400",
  "errorMessage": "Ipc Not provisioned yet. Mapping failed",
  "path": "/circuitservice/v1/circuits/PassiveIngressToEgressMapping",
  "httpStatus": "BAD_REQUEST"
}
```

Topology

Get Topology Details

Description : Get All OTS and OMS Link from topology
URL: onc-topology-service/v1/topology?status=discovered
Type : GET
Inputs : discovered/partial/* Based on status will return the ots and oms link */
Request Body:
Response:


```
{
  "networks": [
   {
     "site": [
      {
        "deviceId": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
        "hostName": "AMPLIFIER",
        "type": "NCS1010",
        "protocol": "NETCONF",
        "errorInfo": null,
        "port": 63524,
        "deviceVersion": "0",
        "siteName": "site_vxrSite_4",
        "siteUID": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
        "frequencyBand": "C",
        "ip": "10.64.98.20",
        "uid": "dd79d3c5-2847-43e6-bc3e-64e512f0b643"
      }

    ],
    "links": {
      "networkLinks": {
        "logical": null,
        "physical": {
          "fiberSpanSection": [
           {
             "hierarchicalUID": "ots_port-bac1fa2c-fafd-3e91-a713-02d14caf8847-0/0/0/0",
             "label": "Fiber-1",
             "srcSite": "ceec2bfb-3fca-305b-a180-19d0de7ff127",
             "sourceEdge": "c4c55edc-9f94-32f8-b36a-0dcdd20565b1",
             "hierarchicalSourceEdge": "edgeSide-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
             "dstSite": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
             "destinationEdge": "2ee53fb6-a7c1-356c-9d76-903c4e1e9209",
             "hierarchicalDestinationEdge": "edgeSide-2775a5a1-048b-3582-b1a0-
e540c9639bc6-0/0/0/0",
```

```
            "l0DesignData": {
              "lengthBasedLoss": null,
              "isLOGOEnabled": null,
              "connectorLossIn": null,
              "connectorLossOut": null,
              "rdFactor": null,
              "cbandRule": null,
              "qdcband": null,
              "cdcband": null
            },
            "bidirectional": true,
            "businessStatus": "FUTURE",
            "networkStatus": "DISCOVERED",
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "fiberType": "SMF",
            "length": 0,
            "loss": null,
            "ageingLoss": null,
            "ageingFactor": null,
            "attenuationCoeff": null,
            "pmd": null,
            "dcnExtension": null,
            "oscFrameType": null,
            "ramanAmplified": null,
            "measurementUnits": "km",
            "srcDeviceUID": "bac1fa2c-fafd-3e91-a713-02d14caf8847",
            "dstDeviceUID": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
            "forward": {
              "hierarchicalUID": "ots_forward_port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
              "srcDeviceUID": "bac1fa2c-fafd-3e91-a713-02d14caf8847",
              "dstDeviceUID": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
              "srcSite": "ceec2bfb-3fca-305b-a180-19d0de7ff127",
              "sourceEdge": "c4c55edc-9f94-32f8-b36a-0dcdd20565b1",
              "sourcePort": "60733daf-901a-3b73-8ede-ab2e84ef3127",
              "hierarchicalSourceEdge": "edgeSide-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
              "hierarchicalSourcePort": "port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
              "dstSite": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
              "destinationEdge": "2ee53fb6-a7c1-356c-9d76-903c4e1e9209",
              "destinationPort": "d4cccf7e-e1b4-3cc4-a3d7-b2b98beb46bb",
              "hierarchicalDestinationEdge": "edgeSide-2775a5a1-048b-3582-b1a0-
e540c9639bc6-0/0/0/0",
```

```
            "hierarchicalDestinationPort": "port-2775a5a1-048b-3582-b1a0-e540c9639bc6-
0/0/0/0",
            "bidirectional": null,
            "businessStatus": "FUTURE",
            "networkStatus": "DISCOVERED",
            "label": "Fiber-1-COUPLE-AZ",
            "length": 0,
            "loss": null,
            "ageingLoss": null,
            "ageingFactor": null,
            "attenuationCoeff": null,
            "pmd": null,
            "measurementUnits": "km",
            "fiberType": "SMF",
            "isOsnrDataValid": false,
            "noise": null,
            "sigma": null,
            "unlocked": [],
            "forced": [],
            "opticalParams": null,
            "channelNumber": null,
            "pmdCoefficient": null,
            "connectorLossIn": null,
            "connectorLossOut": null,
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "uid": "0d2ce68d-2642-339c-8fe5-34fae71528ef"
          },
          "reverse": {
            "hierarchicalUID": "ots_reverse_port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
            "srcDeviceUID": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
            "dstDeviceUID": "bac1fa2c-fafd-3e91-a713-02d14caf8847",
            "srcSite": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
            "sourceEdge": "2ee53fb6-a7c1-356c-9d76-903c4e1e9209",
            "sourcePort": "d4cccf7e-e1b4-3cc4-a3d7-b2b98beb46bb",
            "hierarchicalSourceEdge": "edgeSide-2775a5a1-048b-3582-b1a0-
e540c9639bc6-0/0/0/0",
            "hierarchicalSourcePort": "port-2775a5a1-048b-3582-b1a0-e540c9639bc6-
0/0/0/0",
            "dstSite": "ceec2bfb-3fca-305b-a180-19d0de7ff127",
            "destinationEdge": "c4c55edc-9f94-32f8-b36a-0dcdd20565b1",
            "destinationPort": "60733daf-901a-3b73-8ede-ab2e84ef3127",
            "hierarchicalDestinationEdge": "edgeSide-bac1fa2c-fafd-3e91-a713-
02d14caf8847-0/0/0/0",
```

```
            "hierarchicalDestinationPort": "port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
            "bidirectional": null,
            "businessStatus": "FUTURE",
            "networkStatus": "DISCOVERED",
            "label": "Fiber-1-COUPLE-ZA",
            "length": 0,
            "loss": 300,
            "ageingLoss": null,
            "ageingFactor": null,
            "attenuationCoeff": null,
            "pmd": null,
            "measurementUnits": "km",
            "fiberType": "SMF",
            "isOsnrDataValid": false,
            "noise": null,
            "sigma": null,
            "unlocked": [],
            "forced": [],
            "opticalParams": null,
            "channelNumber": null,
            "pmdCoefficient": null,
            "connectorLossIn": null,
            "connectorLossOut": null,
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "uid": "b30ccec9-f234-34c6-8a68-68094462b5e1"
          },
          "unlocked": [],
          "forced": [],
          "omsLink": [],
          "status": "discovered",
          "dstIp": "10.62.1.4",
          "srcIp": "10.62.1.5",
          "srcSnmpIndex": "190",
          "dstSnmpIndex": "16",
          "version": 1,
          "coddesignData": null,
          "UID": "572bf81f-6426-307d-8228-6e61d934cbb0"
        },
        ],
      "omsLink": [
      {
        "hierarchicalUID": "oms_port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-0/0/0/0",
        "label": "OMS-2",
        "srcSite": "6438743c-6133-3093-816b-e9ba6f53bb61",
```

```
            "sourceEdge": "2df52fd5-7769-3b0d-8a1d-30742595792e",
            "hierarchicalSourceEdge": "edgeSide-0e95ec61-a756-3280-bbe0-984ea8c39235-
0/0/0/0",
            "dstSite": "78642254-4dd9-3ecf-a124-f053d77757a9",
            "destinationEdge": "5ffe9234-c3f5-3914-a74e-92f51657fe2e",
            "hierarchicalDestinationEdge": "edgeSide-a3b0da8a-a64d-32da-abd9-
1f26ac9eec93-0/0/0/0",
            "operState": null,
            "bidirectional": true,
            "businessStatus": "FUTURE",
            "networkStatus": "DISCOVERED",
            "status": "discovered",
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "forward": {
              "hierarchicalUID": "oms_forward_port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
              "label": "OMS-2-COUPLE-ZA",
              "srcDeviceUID": "0e95ec61-a756-3280-bbe0-984ea8c39235",
              "dstDeviceUID": "a3b0da8a-a64d-32da-abd9-1f26ac9eec93",
              "srcSite": "6438743c-6133-3093-816b-e9ba6f53bb61",
              "sourceEdge": "2df52fd5-7769-3b0d-8a1d-30742595792e",
              "hierarchicalSourceEdge": "edgeSide-0e95ec61-a756-3280-bbe0-
984ea8c39235-0/0/0/0",
              "sourcePort": "a2ee13f2-0ec8-3637-b17c-a195bf12d626",
              "hierarchicalSourcePort": "port-0e95ec61-a756-3280-bbe0-984ea8c39235-
0/0/0/0",
              "dstSite": "78642254-4dd9-3ecf-a124-f053d77757a9",
              "destinationEdge": "5ffe9234-c3f5-3914-a74e-92f51657fe2e",
              "destinationPort": "0a2c0029-5baa-3ddf-b7c7-7d526d6424a1",
              "hierarchicalDestinationEdge": "edgeSide-a3b0da8a-a64d-32da-abd9-
1f26ac9eec93-0/0/0/0",
              "hierarchicalDestinationPort": "port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
              "opticalParams": null,
              "channelNumber": null,
              "adminState": "UNLOCKED",
              "serviceState": "ENABLED",
              "isLOGO": true,
              "freqGrid": "FLEX",
              "channelSpacing": null,
              "spectralDensity": 81,
              "uid": "87cf9c61-082e-3c6a-bdd8-9532a1f0295d"
            },
            "reverse": {
```

```
            "hierarchicalUID": "oms_reverse_port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
            "label": "OMS-2-COUPLE-ZA",
            "srcDeviceUID": "a3b0da8a-a64d-32da-abd9-1f26ac9eec93",
            "dstDeviceUID": "0e95ec61-a756-3280-bbe0-984ea8c39235",
            "srcSite": "78642254-4dd9-3ecf-a124-f053d77757a9",
            "sourceEdge": "5ffe9234-c3f5-3914-a74e-92f51657fe2e",
            "hierarchicalSourceEdge": "edgeSide-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
            "sourcePort": "0a2c0029-5baa-3ddf-b7c7-7d526d6424a1",
            "hierarchicalSourcePort": "port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
            "dstSite": "6438743c-6133-3093-816b-e9ba6f53bb61",
            "destinationEdge": "2df52fd5-7769-3b0d-8a1d-30742595792e",
            "destinationPort": "a2ee13f2-0ec8-3637-b17c-a195bf12d626",
            "hierarchicalDestinationEdge": "edgeSide-0e95ec61-a756-3280-bbe0-
984ea8c39235-0/0/0/0",
            "hierarchicalDestinationPort": "port-0e95ec61-a756-3280-bbe0-984ea8c39235-
0/0/0/0",
            "opticalParams": null,
            "channelNumber": null,
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "isLOGO": true,
            "freqGrid": "FLEX",
            "channelSpacing": null,
            "spectralDensity": 81,
            "uid": "cf4c75d0-52d1-3c9c-9c47-d44a3aae7a86"
          },
          "srcDeviceUID": "0e95ec61-a756-3280-bbe0-984ea8c39235",
          "dstDeviceUID": "a3b0da8a-a64d-32da-abd9-1f26ac9eec93",
          "fiberSpanSection": [
            "5f829f6a-ddc7-30b9-b270-3d72fed1453b"
          ],
          "dstIp": "192.168.227.105",
          "srcIp": "192.168.227.106",
          "path": [
            "a3b0da8a-a64d-32da-abd9-1f26ac9eec93-0/0/0/0",
            "0e95ec61-a756-3280-bbe0-984ea8c39235-0/0/0/0"
          ],
          "version": 0,
          "UID": "726a3c14-11c6-3b52-bc9e-30b3d9748caa"
        }
      ],
      "interconnectLinks": []
    }
```

```
        },
        "siteLinks": [
          {
            "hierarchicalUID": null,
            "site": null,
            "logical": null,
            "physical": {
              "l0PhysicalSiteLink": [],
              "l2PhysicalSiteLink": [],
              "sonetphysicalSiteLink": []
            },
            "uid": null
          }
        ]
      }
    }
  ]
}
```


Get Topology For Device

Description : Get All OTS and OMS Link from topology for given device id
URL: onc-topology-service/v1/topology/{deviceId}
Type : GET
Inputs : device /* Based on device id will retrun the ots and oms link */
Request Body:
Response:


```
{
  "networks": [
    {
      "site": [
        {
          "deviceId": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
          "hostName": "AMPLIFIER",
          "type": "NCS1010",
          "protocol": "NETCONF",
          "errorInfo": null,
          "port": 63524,
          "deviceVersion": "0",
          "siteName": "site_vxrSite_4",
          "siteUID": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
          "frequencyBand": "C",
          "ip": "10.64.98.20",
          "uid": "dd79d3c5-2847-43e6-bc3e-64e512f0b643"
        }
```

```
    ],
    "links": {
      "networkLinks": {
        "logical": null,
        "physical": {
          "fiberSpanSection": [
            {
              "hierarchicalUID": "ots_port-bac1fa2c-fafd-3e91-a713-02d14caf8847-0/0/0/0",
              "label": "Fiber-1",
              "srcSite": "ceec2bfb-3fca-305b-a180-19d0de7ff127",
              "sourceEdge": "c4c55edc-9f94-32f8-b36a-0dcdd20565b1",
              "hierarchicalSourceEdge": "edgeSide-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
              "dstSite": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
              "destinationEdge": "2ee53fb6-a7c1-356c-9d76-903c4e1e9209",
              "hierarchicalDestinationEdge": "edgeSide-2775a5a1-048b-3582-b1a0-
e540c9639bc6-0/0/0/0",
              "l0DesignData": {
                "lengthBasedLoss": null,
                "isLOGOEnabled": null,
                "connectorLossIn": null,
                "connectorLossOut": null,
                "rdFactor": null,
                "cbandRule": null,
                "qdcband": null,
                "cdcband": null
              },
              "bidirectional": true,
              "businessStatus": "FUTURE",
              "networkStatus": "DISCOVERED",
              "adminState": "UNLOCKED",
              "serviceState": "ENABLED",
              "fiberType": "SMF",
              "length": 0,
              "loss": null,
              "ageingLoss": null,
              "ageingFactor": null,
              "attenuationCoeff": null,
              "pmd": null,
              "dcnExtension": null,
              "oscFrameType": null,
              "ramanAmplified": null,
              "measurementUnits": "km",
              "srcDeviceUID": "bac1fa2c-fafd-3e91-a713-02d14caf8847",
              "dstDeviceUID": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
```

```
            "forward": {
              "hierarchicalUID": "ots_forward_port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
              "srcDeviceUID": "bac1fa2c-fafd-3e91-a713-02d14caf8847",
              "dstDeviceUID": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
              "srcSite": "ceec2bfb-3fca-305b-a180-19d0de7ff127",
              "sourceEdge": "c4c55edc-9f94-32f8-b36a-0dcdd20565b1",
              "sourcePort": "60733daf-901a-3b73-8ede-ab2e84ef3127",
              "hierarchicalSourceEdge": "edgeSide-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
              "hierarchicalSourcePort": "port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
              "dstSite": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
              "destinationEdge": "2ee53fb6-a7c1-356c-9d76-903c4e1e9209",
              "destinationPort": "d4cccf7e-e1b4-3cc4-a3d7-b2b98beb46bb",
              "hierarchicalDestinationEdge": "edgeSide-2775a5a1-048b-3582-b1a0-
e540c9639bc6-0/0/0/0",
              "hierarchicalDestinationPort": "port-2775a5a1-048b-3582-b1a0-e540c9639bc6-
0/0/0/0",
              "bidirectional": null,
              "businessStatus": "FUTURE",
              "networkStatus": "DISCOVERED",
              "label": "Fiber-1-COUPLE-AZ",
              "length": 0,
              "loss": null,
              "ageingLoss": null,
              "ageingFactor": null,
              "attenuationCoeff": null,
              "pmd": null,
              "measurementUnits": "km",
              "fiberType": "SMF",
              "isOsnrDataValid": false,
              "noise": null,
              "sigma": null,
              "unlocked": [],
              "forced": [],
              "opticalParams": null,
              "channelNumber": null,
              "pmdCoefficient": null,
              "connectorLossIn": null,
              "connectorLossOut": null,
              "adminState": "UNLOCKED",
              "serviceState": "ENABLED",
              "uid": "0d2ce68d-2642-339c-8fe5-34fae71528ef"
            },
            "reverse": {
```

```
            "hierarchicalUID": "ots_reverse_port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
            "srcDeviceUID": "2775a5a1-048b-3582-b1a0-e540c9639bc6",
            "dstDeviceUID": "bac1fa2c-fafd-3e91-a713-02d14caf8847",
            "srcSite": "843b26ba-52b0-31a1-87c7-286adadf6f4f",
            "sourceEdge": "2ee53fb6-a7c1-356c-9d76-903c4e1e9209",
            "sourcePort": "d4cccf7e-e1b4-3cc4-a3d7-b2b98beb46bb",
            "hierarchicalSourceEdge": "edgeSide-2775a5a1-048b-3582-b1a0-
e540c9639bc6-0/0/0/0",
            "hierarchicalSourcePort": "port-2775a5a1-048b-3582-b1a0-e540c9639bc6-
0/0/0/0",
            "dstSite": "ceec2bfb-3fca-305b-a180-19d0de7ff127",
            "destinationEdge": "c4c55edc-9f94-32f8-b36a-0dcdd20565b1",
            "destinationPort": "60733daf-901a-3b73-8ede-ab2e84ef3127",
            "hierarchicalDestinationEdge": "edgeSide-bac1fa2c-fafd-3e91-a713-
02d14caf8847-0/0/0/0",
            "hierarchicalDestinationPort": "port-bac1fa2c-fafd-3e91-a713-02d14caf8847-
0/0/0/0",
            "bidirectional": null,
            "businessStatus": "FUTURE",
            "networkStatus": "DISCOVERED",
            "label": "Fiber-1-COUPLE-ZA",
            "length": 0,
            "loss": 300,
            "ageingLoss": null,
            "ageingFactor": null,
            "attenuationCoeff": null,
            "pmd": null,
            "measurementUnits": "km",
            "fiberType": "SMF",
            "isOsnrDataValid": false,
            "noise": null,
            "sigma": null,
            "unlocked": [],
            "forced": [],
            "opticalParams": null,
            "channelNumber": null,
            "pmdCoefficient": null,
            "connectorLossIn": null,
            "connectorLossOut": null,
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "uid": "b30ccec9-f234-34c6-8a68-68094462b5e1"
          },
          "unlocked": [],
          "forced": [],
```

```
            "omsLink": [],
            "status": "discovered",
            "dstIp": "10.62.1.4",
            "srcIp": "10.62.1.5",
            "srcSnmpIndex": "190",
            "dstSnmpIndex": "16",
            "version": 1,
            "coddesignData": null,
            "UID": "572bf81f-6426-307d-8228-6e61d934cbb0"
          },
           ],
         "omsLink": [
          {
            "hierarchicalUID": "oms_port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-0/0/0/0",
            "label": "OMS-2",
            "srcSite": "6438743c-6133-3093-816b-e9ba6f53bb61",
            "sourceEdge": "2df52fd5-7769-3b0d-8a1d-30742595792e",
            "hierarchicalSourceEdge": "edgeSide-0e95ec61-a756-3280-bbe0-984ea8c39235-
0/0/0/0",
            "dstSite": "78642254-4dd9-3ecf-a124-f053d77757a9",
            "destinationEdge": "5ffe9234-c3f5-3914-a74e-92f51657fe2e",
            "hierarchicalDestinationEdge": "edgeSide-a3b0da8a-a64d-32da-abd9-
1f26ac9eec93-0/0/0/0",
            "operState": null,
            "bidirectional": true,
            "businessStatus": "FUTURE",
            "networkStatus": "DISCOVERED",
            "status": "discovered",
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "forward": {
              "hierarchicalUID": "oms_forward_port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
              "label": "OMS-2-COUPLE-ZA",
              "srcDeviceUID": "0e95ec61-a756-3280-bbe0-984ea8c39235",
              "dstDeviceUID": "a3b0da8a-a64d-32da-abd9-1f26ac9eec93",
              "srcSite": "6438743c-6133-3093-816b-e9ba6f53bb61",
              "sourceEdge": "2df52fd5-7769-3b0d-8a1d-30742595792e",
              "hierarchicalSourceEdge": "edgeSide-0e95ec61-a756-3280-bbe0-
984ea8c39235-0/0/0/0",
              "sourcePort": "a2ee13f2-0ec8-3637-b17c-a195bf12d626",
              "hierarchicalSourcePort": "port-0e95ec61-a756-3280-bbe0-984ea8c39235-
0/0/0/0",
              "dstSite": "78642254-4dd9-3ecf-a124-f053d77757a9",
              "destinationEdge": "5ffe9234-c3f5-3914-a74e-92f51657fe2e",
              "destinationPort": "0a2c0029-5baa-3ddf-b7c7-7d526d6424a1",
```

```
            "hierarchicalDestinationEdge": "edgeSide-a3b0da8a-a64d-32da-abd9-
1f26ac9eec93-0/0/0/0",
            "hierarchicalDestinationPort": "port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
            "opticalParams": null,
            "channelNumber": null,
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "isLOGO": true,
            "freqGrid": "FLEX",
            "channelSpacing": null,
            "spectralDensity": 81,
            "uid": "87cf9c61-082e-3c6a-bdd8-9532a1f0295d"
          },
          "reverse": {
            "hierarchicalUID": "oms_reverse_port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
            "label": "OMS-2-COUPLE-ZA",
            "srcDeviceUID": "a3b0da8a-a64d-32da-abd9-1f26ac9eec93",
            "dstDeviceUID": "0e95ec61-a756-3280-bbe0-984ea8c39235",
            "srcSite": "78642254-4dd9-3ecf-a124-f053d77757a9",
            "sourceEdge": "5ffe9234-c3f5-3914-a74e-92f51657fe2e",
            "hierarchicalSourceEdge": "edgeSide-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
            "sourcePort": "0a2c0029-5baa-3ddf-b7c7-7d526d6424a1",
            "hierarchicalSourcePort": "port-a3b0da8a-a64d-32da-abd9-1f26ac9eec93-
0/0/0/0",
            "dstSite": "6438743c-6133-3093-816b-e9ba6f53bb61",
            "destinationEdge": "2df52fd5-7769-3b0d-8a1d-30742595792e",
            "destinationPort": "a2ee13f2-0ec8-3637-b17c-a195bf12d626",
            "hierarchicalDestinationEdge": "edgeSide-0e95ec61-a756-3280-bbe0-
984ea8c39235-0/0/0/0",
            "hierarchicalDestinationPort": "port-0e95ec61-a756-3280-bbe0-984ea8c39235-
0/0/0/0",
            "opticalParams": null,
            "channelNumber": null,
            "adminState": "UNLOCKED",
            "serviceState": "ENABLED",
            "isLOGO": true,
            "freqGrid": "FLEX",
            "channelSpacing": null,
            "spectralDensity": 81,
            "uid": "cf4c75d0-52d1-3c9c-9c47-d44a3aae7a86"
          },
          "srcDeviceUID": "0e95ec61-a756-3280-bbe0-984ea8c39235",
          "dstDeviceUID": "a3b0da8a-a64d-32da-abd9-1f26ac9eec93",
```

```
            "fiberSpanSection": [
              "5f829f6a-ddc7-30b9-b270-3d72fed1453b"
            ],
            "dstIp": "192.168.227.105",
            "srcIp": "192.168.227.106",
            "path": [
              "a3b0da8a-a64d-32da-abd9-1f26ac9eec93-0/0/0/0",
              "0e95ec61-a756-3280-bbe0-984ea8c39235-0/0/0/0"
            ],
            "version": 0,
            "UID": "726a3c14-11c6-3b52-bc9e-30b3d9748caa"
          }
        ],
        "interconnectLinks": []
      }
    },
    "siteLinks": [
      {
        "hierarchicalUID": null,
        "site": null,
        "logical": null,
        "physical": {
          "l0PhysicalSiteLink": [],
          "l2PhysicalSiteLink": [],
          "sonetphysicalSiteLink": []
        },
        "uid": null
      }
    ]
  }
 ]
}
```

Alien Configuration

Retrieve optical interfaces configuration

Description:
This API is used retrieve optical interfaces configuration in short format: only with keys to use for working mode identification. The response is in xml format.
URL: onc-pce-service/getOpticalInterfaceConfiguration
Type : GET
Inputs :
Request Body:
Response:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<controlPlane>
   <opticalInterfaces>
      <opticalInterface>
         <name>NCS1004</name>
         <equipmentType>ALIEN EQPT</equipmentType>
         <equipmentId>
            <pid>NCS1K4-1.2T-K9</pid>
            <vName>CISCO</vName>
            <vids>
               <vid>00B08E</vid>
            </vids>
            <platforms/>
         </equipmentId>
         <workingModes>
            <workingMode>
               <name>R500G#1955</name>
               <config>
                  <modulationFormat>32QAM</modulationFormat>
                  <trunkOperMode>TRK_500G</trunkOperMode>
                  <fec>SD_FEC_27_DE_OFF</fec>
                  <baudRate>69.4351003125</baudRate>
                  <bitSymb>5.0</bitSymb>
                  <datarate>R500G</datarate>
                  <openConfigCode>1955</openConfigCode>
                  <subMode/>
               </config>
            </workingMode>

         </workingModes>
      </opticalInterface>
      ...
```

```
    </opticalInterfaces>
</controlPlane>
```

Add an alien configuration

Description:

This API is used add a new optical interface configuration with related optical working modes and optical classes. It is also used to replace an existing configuration. For each entry, into JSon format response, there is a detailed result (SUCCESS/FAILURE) for each entry.

URL: onc-pce-service/file/importOpticalInterfaceConfigurationFile

Type : POST

Inputs :

Request Body:

Response:

Success


```
{
   "lastupdate": "2023-04-21T17:43:29.037+0200",
   "result": "SUCCESS",
   "details": [
      {
         "name": "OPTIF_MOCK",
         "result": "SUCCESS",
         "error": "[]"
      },
      {
         "name": "TEST_CLASS",
         "result": "SUCCESS",
         "error": "[]"
      }
   ]
}
```


Error conditions:

- Missing mandatory parameter


```
{
   "result": "FAILURE",
   "details": [
      {
         "name": "TEST_ALIEN_TAC",
         "result": "FAILURE",
         "error": "[Missing TEST_ALIEN_TAC null name]"
      }
   ]
}
```


- Input parameter not valid


```
{
   "result": "FAILURE",
   "details": [
      {
```

```
            "name": "TEST_ALIEN_TAC",
            "result": "FAILURE",
            "error": "[Working Mode TEST_ALIEN_TAC, error [PCE-MD00058] - Invalid Working Mode opticalClass
aa]"
        }
    ]
}
```


- Invalid input file


```
{
    "lastupdate": "2023-04-21T17:46:33.403+0200",
    "result": "FAILURE",
    "details": []
}
```


User Management

Login

Description: Login User
URL: v2/users/login
Type: POST
Inputs:
Request Body:
{
UserName :- admin
Password :- *******

}

Response:


```
{
  "code": 200,
  "type": "unknown",
  "message": "logged in user session:1676457725835"
}
```


Error Condition:
Code :400
Response Message: Invalid username/password supplied

Logout

Description: Logout User
URL: /v2/user/logout
Type: POST

Inputs:
Request Body:
Response:

```
{
  "code": 200,
  "type": "unknown",
  "message": "ok"
}
```

Change Password

Description : User can change their password by giving username, old password, and new password
URL: access-management/users/change-password
Type: PUT
Inputs:
Request Body:

```
{

userName: 'admin'

oldPassword: 'test'

newPassword: 'cisco123'

}
```

Response:

```
{
    "errors": [],
    "warnings": [],
    "confirmations": [],
    "valid": true
}
```

Validate Token

Description: User can validate their Authentication token
URL: v2/user/validateToken
Type : POST
Inputs :
Request Body:
Response:

```
{
    "code": 200,
    "message": "Token Validated."
```

}
Error Condition :
Code :401
Response: Unauthorized