MicroClimate™ Management System Netconf Server (R4.0.0)

User Guide

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089
support

Phone: 888.JUNIPER (888.586.4737)
Web https://support.juniper.net/support/requesting-

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

The information in this document is current as of the date on the title page.

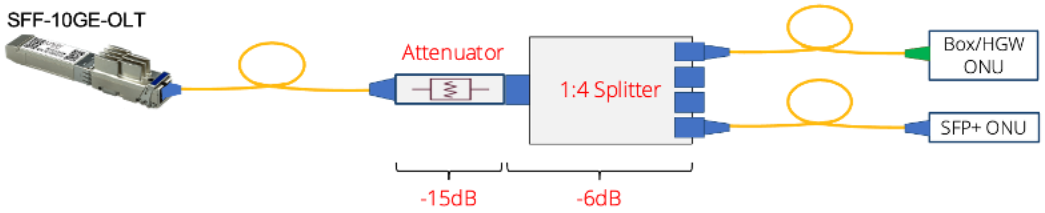
WARNING !

Refer to Install Guide before Installation

Warranty Notice: Device Attenuation Required

Do not connect OLT directly to ONUs without proper attenuation. PON transceivers will be **permanently damaged** unless connected with minimum 16dB attenuation (20dB recommended)
Damage from optical overload will void warranty.

Combination of attenuator and splitters can provide required attenuation -- see example:



Revision History

Revision	Date	Comments
R4.0.0	August 1, 2023	
R3.2.0	November 19, 2022	
R3.1.0	July 26, 2022	
R3.0.0	April 7, 2022	
R2.3.0	November 19, 2021	
R2.2.0	July 1, 2021	
R2.1.1	April 2, 2021	
R2.1.0	February 12, 2021	
R2.0.0	October 6, 2020	
R1.3.1	July 07, 2020	
R1.3.0	May 29, 2020	
R1.2.0	February 28, 2020	
R1.1.0	November 20, 2019	
A1.1.0	September 30, 2019	Initial version.

References

ID	Document Description
BBF YANG	Broadband Forum YANG Modules, <https://github.com/BroadbandForum/yang>.
MCMS Installation Guide	TN051,Tibit MCMS Installation Guide
MCMS Netconf Server	TN037, MicroClimate™ Management System Netconf Server User Guide & Release Notes
MCMS PON Controller	TN026, MicroClimate™ Management System PON Controller User Guide & Release Notes
MCMS PON Manager	TN035, MicroClimate™ Management System PON Manager User Guide & Release Notes
MongoDB	MongoDB 4.0 Manual, <https://docs.mongodb.com/v4.0/>.
RFC 8343	Bjorklund, M., "A YANG Data Model for Interface Management", RFC 8343, DOI 10.17487/RFC8343, March 2018, <https://www.rfc-editor.org/info/rfc8343>.
Tibit MicroPlug™ OLT	TN002, Tibit MicroPlug™ OLT Release Notes
TR-383	TR-383 Common YANG Modules for Access Networks, Issue 1, Amendment 3, October 2020, <https://www.broadband-forum.org/technical/download/TR-383_Amendment-3.pdf>.
TR-385	TR-385 ITU-T PON YANG Modules, Issue 2, October 2020, <https://www.broadband-forum.org/technical/download/TR-385_Issue-2.pdf.pdf>.

Introduction

The MicroClimate™ Management System (MCMS) is the management solution for PON networks. The MCMS architecture is shown in Figure 1 and consists of the MCMS PON Manager graphical user interface, MCMS Netconf Server, and MCMS PON Controller. Together these components provide a complete network management solution for provisioning and monitoring MicroPlug™ OLT devices, as well as the subtended MicroPlug™ ONU devices and third-party ONUs compliant with the XGS-PON and 10G-EPON standards.

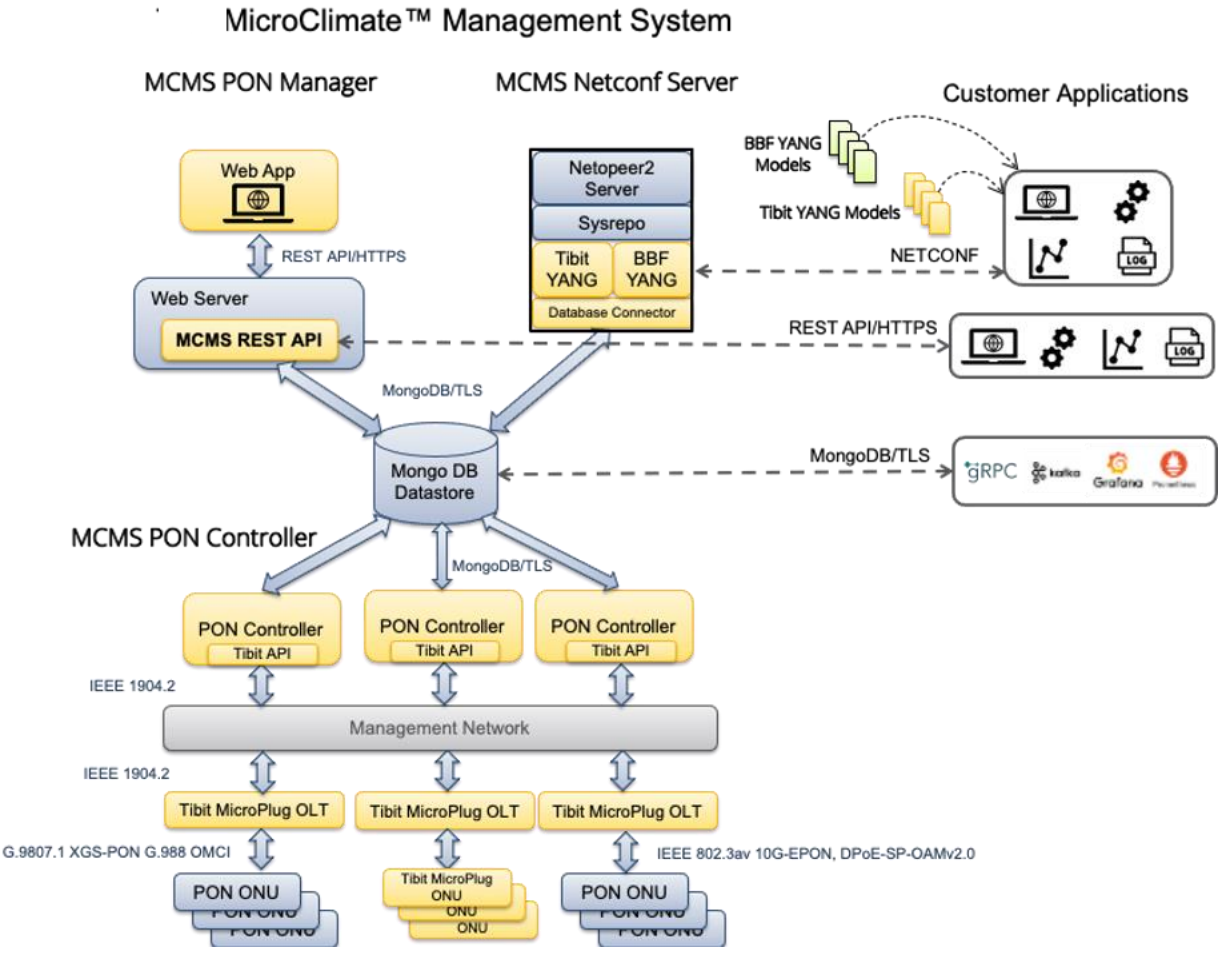


Figure 1 - Tibit MicroClimate™ Management System Architecture

NETCONF/YANG Interface

The MCMS Netconf Server provides an interface for managing the PON Controller, MicroPlug OLTs, and ONU devices using standard NETCONF protocols and tools.

The NETCONF/YANG interface supports the following IETF standards:

- RFC 4742 NETCONF Protocol over Secure Shell (SSH)
- RFC 5277 NETCONF Event Notifications
- RFC 6241 Network Configuration Protocol (NETCONF)
- RFC 6020 YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)
- RFC 7950 The YANG 1.1 Data Modeling Language

The NETCONF/YANG interface supports the following Broadband Forum standards:

- TR-383, Issue 1, Amendment 3 Common YANG Modules for Access Networks
- TR-385, Issue 2 ITU-T PON YANG Modules

Other Software Components

This User Guide focuses on the MicroClimate Management System NETCONF/YANG interface. See [MCMS PON Manager] for more information regarding the MCMS PON Manager graphical user interface, REST API, and MongoDB datastore. See [MCMS PON Controller] for more information regarding the MCMS PON Controller stateless management controller and driver application.

Architecture Overview

The MCMS Netconf Server is built on the Netopeer2 and Sysrepo open source NETCONF management framework. The MCMS Netconf Database Connector, in conjunction with Netopeer2 and Sysrepo, make up the components of the Netconf Server. Together these components provide the NETCONF/YANG interface for managing a MicroPlug OLT PON network. The software architecture is shown in the figure below.

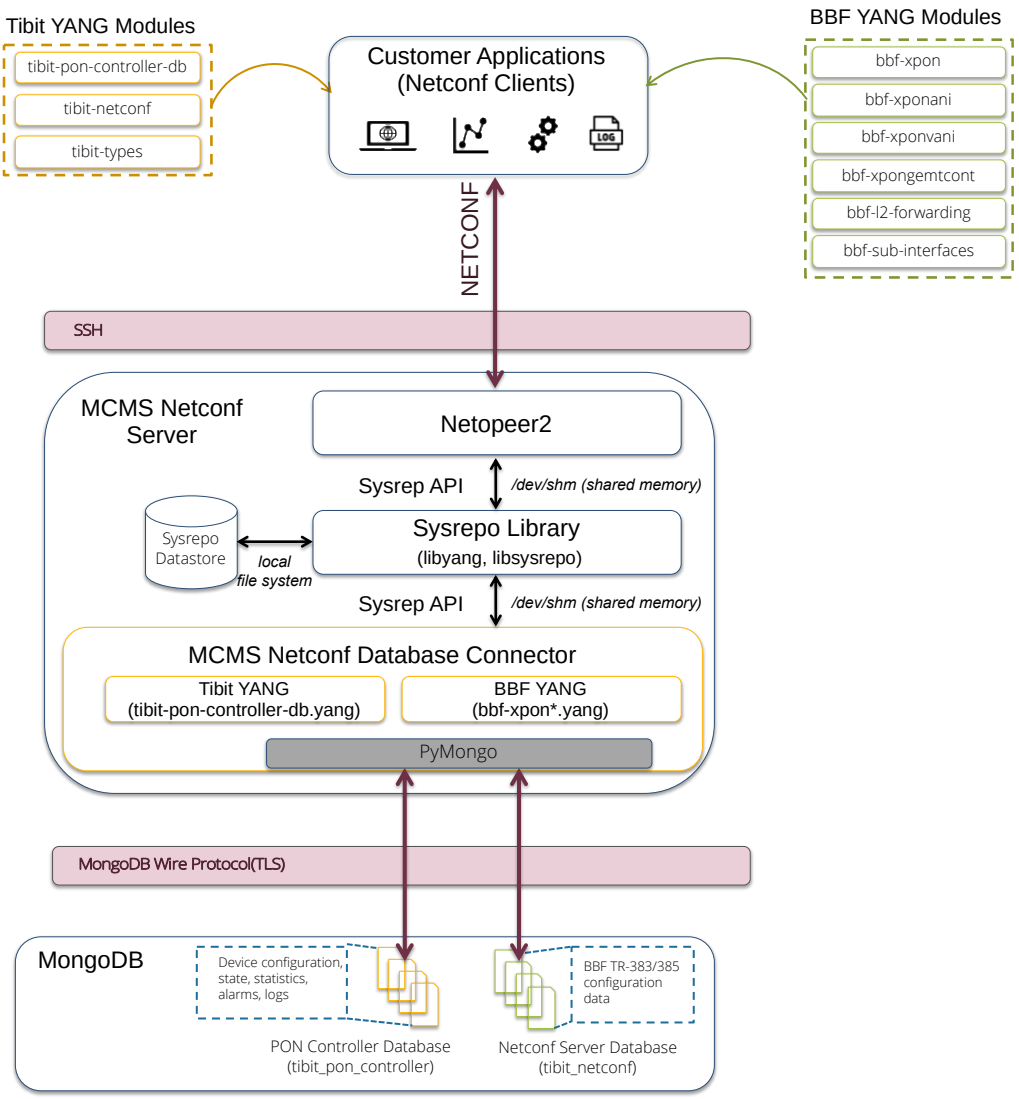


Figure 2 - MCMS Netconf Server Architecture

Netopeer2 Server

Netopeer2 is a component of the open source NETCONF framework used in the Netconf Server. The Netopeer2 Server implements the server side of the RFC 6241 NETCONF Protocol. The MCMS Netconf Server supports Secure Shell (SSH) transport and is accessible on TCP port 830 by default. Netopeer2 terminates the RPC requests from NETCONF client applications and calls the Sysrepo API to service requests for specific YANG models.

Sysrepo (libsysrepo)

Sysrepo is a component of the open source NETCONF framework used in the Netconf Server. It provides tools and functions for managing YANG datastores, including management for the startup, running, and candidate configuration. Netopeer2 and the Database Connector use the Sysrepo API to interface with the datastores using Linux Shared Memory (/dev/shm).

Database Connector

The MCMS Netconf Database Connector translates and synchronizes data between Sysrepo and the Mongo Database. The Database Connector provides two-way synchronization. Configuration changes made through the NETCONF/YANG interface are synchronized to MongoDB. Likewise, changes made to MongoDB (e.g., through the MCMS PON Manager Web App) are synchronized to the Sysrepo running configuration datastore.

YANG Models

provides YANG models for the configuration and monitoring of the PON Controller, MicroPlug OLTs, MicroPlug ONUs, and other third-party ONUs compliant with the DPoE and XGS PON standards. See Section YANG Models for more information on the YANG models supported by the MCMS Netconf Server.

Broadband Forum YANG Models

The MCMS Netconf Server supports service configuration using Broadband Forum YANG Models, including support for TR-383 and TR-385. See Section Broadband Forum YANG Models for more information on the BBF YANG models supported by the Netconf Server.

MongoDB

The Mongo database provides the datastore for the MicroClimate Management System. See [MCMS PON Controller] for a description of the data model used to manage the PON network. MongoDB is an open source, secure database (www.mongodb.com) which employs a NoSQL

architecture. See section MongoDB Configuration for information on installation and configuration for use in management solutions.

Although MongoDB is shown as part of the MCMS architecture, MongoDB is not provided as part of any MCMS installation package. MongoDB is a dependency of the MCMS PON Manager and Netconf Server.

Installation

This section describes how to install, configure, and run the MCMS Netconf Server software. This setup guide assumes that a MongoDB server has been installed and configured as described in [MCMS Installation Guide].

Requirements and Dependencies

Supported Operating Systems

Operating System	Version(s)
Ubuntu, 64-bit	18.04, 20.04

Software and Firmware Dependencies

Package	Version
PON Controller	R4.0.0
OLT Firmware	R4.0.0
ONU Firmware	R4.0.0

MongoDB Dependencies

NOTE: MongoDB is not installed by the Netconf .deb package. See [MCMS Installation Guide] for information on MongoDB installation, setup, and operation.

Database Package	Ubuntu 18.04	Ubuntu 20.04
MongoDB	4.0	4.4

Required APT Packages

The following Ubuntu Linux apt packages are automatically installed by the Netconf .deb package. See section Installation for installation instructions for the .deb package.

Ubuntu 18.04		Ubuntu 20.04	
Package	**Version**	**Package**	**Version**
libc6	2.27	libc6	2.27
libcurl3-gnutls	7.16.2	libcrypt1	1:4.1.0
libgcc1	1:3.0	libcurl3-gnutls	7.16.2
libpcre3	8.39	libgcc-s1	3.0
libpython3.6	3.6.5	libpcre3	8.39
libssl1.1	1.1.1	libpython3.8	3.8.2
libstdc++6	5.2	libssl1.1	1.1.1
zlib1g	1:1.1.4	libstdc++6	5.2
coreutils	8.28	zlib1g	1:1.1.4
gawk	1:4.1.4	coreutils	8.30
jq	1.5	gawk	1:5.0.1
openssl	1.1.1	jq	1.6
		openssl	1.1.1

TCP/IP Ports and Network Services

The MCMS Netconf Server is deployed over the SSH protocol. By default, the Netconf Server is configured to listen on the following TCP/IP ports:

Network Service	TCP Port	Description
NETCONF	830	Listen TCP port for the SSH interface provided by the Netconf Server.

In addition to providing network services, the MCMS Netconf Server requires network services from external systems to operate. The Netconf Server requires client access to the following network services:

Network Service	TCP Port	Description
MongoDB	27017	Default TCP port of the MongoDB server that the Netconf Server connects to.

Package Contents

The MCMS Netconf Server software is provided as a compressed .zip file. The contents of the package are described in the table below.

File/Directory	Description
docker/	Directory containing scripts for building and running the MCMS Netconf Server in a docker container.
examples/	Directory containing example scripts for configuring and managing the MicroPlug OLT PON network using NETCONF/YANG.
examples/bbf	Directory containing examples for using Broadband Forum TR-383 and TR385 YANG models for configuring OLT and ONU devices. See Examples for descriptions of the examples contained in this directory.
examples/nacm	Directory containing examples for configuring the Network Configuration Access Control Model (NACM) for the MCMS Netconf Server.
examples/sdn	Directory containing examples for using the MCMS Netconf Server with an SDN controller supporting a RESTCONF interface.
INSTALL.txt	Instructions for installing and quick start guide for running the MCMS Netconf Server.
LICENSE.txt	Text file that lists the licensing information for third party software used by the MCMS Netconf Server.
tibit-netconf.deb	Debian installer package containing the MCMS Netconf Server application binaries, default configuration, and supporting files.
yang/	Directory containing the YANG modules and other YANG modules supported by the server.

MongoDB Configuration

These instructions assume the MongoDB server has been installed and configured as described in [MCMS Installation Guide].

The MCMS Netconf Server requires a change to the default MongoDB configuration to enable replica sets. The MCMS Netconf Server uses MongoDB change streams to watch for changes in the database and synchronize updates with the Sysrepo datastore. Change Streams require Replica Sets to be configured in the MongoDB server.

Note: A Replica Set needs to be configured to support Change Streams, but multiple MongoDB servers are not required. Change Streams are supported using a Replica Set containing a single MongoDB server.

Installation

This section describes the steps to install MCMS Netconf Server software using the Debian (.deb) package. The Linux 'apt' utility is used to manage the .deb package, which contains the server binaries, configuration, and supporting files. The table below describes where the files are installed on the target system.

Directory	Description
/etc/tibit/netconf/	Configuration files for the MCMS Netconf Server.
/opt/tibit/netconf/	MCMS Netconf Server binaries, examples, YANG modules, and other supporting files.
/var/log/tibit/	Log files generated by the Netconf server.

Prerequisites

- Ubuntu Linux should be running with the latest updates.
- An active internet connection is required to install Ubuntu Linux apt packages.
- MongoDB must be configured with a Replica Set as described in Section MongoDB Configuration.
- Root level access is required to install the software.

Installation Steps

1. From a Linux shell, unpack the .zip file and change to the new unpacked directory.

```
unzip R4.0.0-Netconf.zip
cd R4.0.0-Netconf
```

2. Use 'apt' to install the Netconf .deb package. Note: the installation requires root level privileges.

```
sudo apt-get install ./tibit-netconf_R4.0.0_amd64.deb
```

3. Configure the MCMS Netconf Server's MongoDB connection information. Edit the file '/etc/tibit/netconf/NetconfInit.json' and modify the IP address, port number, and database name as required.

```
cat /etc/tibit/netconf/NetconfInit.json
    {
        "Logging": {
            "Filename" : "/var/log/tibit/netconf.log",
            "FileCount" : 3,
            "FileSize" : 1024000,
            "Netconf" : {
                "Console" : "INFO",
                "File" : "INFO",
                "Syslog" : "INFO"
            }
        },
        "MongoDB": {
            "auth_db": "tibit_users",
            "auth_enable": false,
            "ca_cert_path": "/etc/tibit/ca.pem",
            "host": "127.0.0.1",
            "name": "tibit_pon_controller",
            "password": "",
            "port": "27017",
            "tls_enable": false,
            "username": ""
        }
    }
```

4. Restart the MCMS Netconf Server using Linux systemd scripts.

```
sudo systemctl restart tibit-netconf
```

Validation Steps

From a shell, run the '/opt/tibit/netconf/examples/nc_get_tibitnc_version.sh' script and verify the MCMS Netconf Server version is reported in the output as highlighted below.

The script prompts for a password as shown below. When prompted, enter the password for the current user.

```
$ cd /opt/tibit/netconf/examples
$ ./nc_get_tibitnc_version.sh
Interactive SSH Authentication
Type your password:
Password:
DATA
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <netconf-state xmlns="urn:com:tibitcom:ns:yang:netconf">
    <version>
      <build-date>2023-08-01T14:57:18-07:00</build-date>
      <build-sha>d234fa7</build-sha>
      <version>R4.0.0</version>
    </version>
    <database>
      <ip-address>127.0.0.1</ip-address>
      <port>27017</port>
      <name>tibit_pon_controller</name>
      <status>online</status>
    </database>
  </netconf-state>
</data>
```

Uninstall

The uninstall process deletes the MCMS Netconf Server binaries and configuration files. This script completely removes the /etc/tibit/netconf and /opt/tibit/netconf directories from the file system.

Uninstall Steps

1. Use 'apt-get' to uninstall the Netconf service. Note: uninstall requires root level privileges. Note: this will completely remove the /etc/tibit/netconf and /opt/tibit/netconf directories.

   ```
   sudo apt-get purge tibit-netconf
   ```

2. If the following warning occurs during 'apt-get purge', use 'rm -rf /opt/tibit/netconf' to completely remove the /opt/tibit/netconf directory.

   ```
   dpkg: warning: while removing tibit-netconf, directory
   '/opt/tibit/netconf' not empty so not removed
   ```

```
rm -rf /opt/tibit/netconf
```

NETCONF Features

This section describes the RFC 6241 NETCONF protocol features and capabilities supported by the MCMS Netconf Server.

Protocol Operations

The following protocol operations are supported:

- <get>
- <get-config>
- <edit-config>
- <copy-config>
- <lock>
- <unlock>
- <close-session>
- <kill-session>
- <delete-config>

Datastores

The following datastores are supported:

- candidate
- running

The following datastores are ***not*** supported ***for Tibit and BBF YANG modules***:

- startup

Capabilities

The following NETCONF capabilities are supported:

- :writable-running
- :candidate
- :rollback-on-error
- :validate
- :xpath
- :url

The following NETCONF capabilities are ***not*** supported:

- :confirmed-commit
- :startup

Notifications *(Beta)*

The following notification features are supported:

- <create-subscription>
- <notification>

YANG Models

YANG models available for the Tibit Profile-based Management solution are listed in the table below.

YANG Model	Description
tibit-bbf-interfaces@ 2023-06-30	YANG for mapping BBF configuration to OLT MicroPlug devices.
tibit-netconf@2023-06-30	MCMS Netconf Server version and status information.
tibit-pon-controller-db@ 2023-06-30	Configuration and monitoring for the Tibit PON Controller, Tibit MicroPlug OLTs, Tibit MicroPlug ONUs, and third-party ONUs compliant with the XGS-PON and 10G-EPON standards. These YANG definitions represent the data model defined by [MCMS PON Controller].
tibit-types@2023-06-30	Common data types shared across all Tibit YANG models.

tibit-bbf-interfaces

The tibit-bbf-interfaces YANG model is used to associate BBF YANG configuration to specific OLT MicroPlug devices. See Broadband Forum YANG Models for more information on using BBF YANG to configure OLT and ONU devices. The tibit-bbf-interfaces configuration attributes are described in the table below.

Attribute	Description
/tibit-bbf-interfaces:interfaces/tibit-bbf-interfaces:olt-interface-map	
device-id	MAC address identifying the MicroPlug OLT device.
pon/channel-group-ref	Reference to an existing PON Channel Group interface used to configure the OLT device referenced by device-id.
pon/channel-partition-ref	Reference to an existing PON Channel Partition interface used to configure the OLT device referenced by device-id.
pon/channel-pair-ref	Reference to an existing PON Channel Pair interface used to configure the OLT device referenced by device-id.

pon/channel-termination-ref	Reference to an existing PON Channel Termination interface used to configure the OLT device referenced by device-id.
nni/interface	Reference to an existing NNI or Uplink interface used to configure the OLT device referenced by device-id.

tibit-netconf

The tibit-netconf YANG model reports state and operational data for the Database Connector, including database connection status, diagnostic information, logging configuration, command statistics, system and version information, and system logs. The tibit-netconf attributes are described in the table below.

Attribute	Description
/tibit-netconf:netconf	
logging	Logging configuration.
console	Severity level for log messages sent to the console.
file	Severity level for log messages sent to the log file.
syslog	Severity level for log messages sent to the Syslog.
database	Severity level for log messages sent to the database.
maximum-log-size	The maximum size of the Netconf Server log collection in MongoDB. When the maximum size is reached, the oldest entries are removed to make space for new entries.
/tibit-netconf:netconf-state	
database	MongoDB connection status information.
ip-address	IP Address or hostname used to connect to the MongoDB server.
port	TCP port number used to connect to the MongoDB server.

name	Name of the database used as the datastore for managing PON devices.
replica-set	Replica set information. (enabled, disabled, name)
servers	List of all known MongoDB servers. (ip address, port, latency, status, error status)
status	MongoDB connection status. (init, online, and error).
diagnostics	Netconf server database connector diagnostic information.
netconf-sr-change	Count of NETCONF change request callbacks.
netconf-sr-get	Count of NETCONF get operational data request callbacks.
netconf-sr-rpc	Count of NETCONF RPC request callbacks.
poncntl-db-init	Count of PON Controller database (re)initializations.
poncntl-db-update	Count of PON Controller database updates.
poncntl-db-drop	Count of PON Controller database drop requests.
netconf-db-update	Count of netconf-server database updates.
netconf-db-drop	Count of netconf-server database drop requests.
logging	Netconf Database Connector logging status.
console	Severity level for log messages sent to the console.
file	Severity level for log messages sent to the log file.
syslog	Severity level for log messages sent to the Syslog.
database	Severity level for log messages sent to the database.
maximum-log-size	The maximum size of the Netconf Server log collection in MongoDB. When the maximum size is reached, the oldest entries are removed to make space for new entries.
statistics	Collection of statistics.

db-commands	MongoDB command statistics. (min/max/average latency, success/fail count)
db-connections	MongoDB connection statistics. (current/max/average count)
system	Netconf Database Connector system information.
hostname	Fully qualified domain name of the system hosting the NETCONF server.
name	Name of the system hosting the NETCONF server.
version	Database Connector version information.
build-date	Build date.
build-sha	Build SHA hash.
version	Version string (e.g., R4.0.0).
/tibit-netconf:netconf-log	
log-table	NETCONF Database Connector Log.
id	Log number.
system-name	System name.
message	Log text.
severity	Log severity.
timestamp	Timestamp when the message was logged.
/tibit-netconf:netconf-clear	
netconf-clear	Clear log data fot Netconf from the database.

tibit-pon-controller-db

The tibit-pon-controller-db YANG model represents the Mongo database model that the PON Controller uses to manage the PON network. See [MCMS PON Controller] for a description of

the data model defined by the PON Controller. A mapping between the tibit-pon-controller-db YANG model and MongoDB is shown in the table below.

tibit-pon-controller-db YANG Container	MongoDB Collection	Description
:controller/	CNTL-CFG	PON Controller configuration
:controller-alarm-history/	CNTL-ALARM-HIST-STATE	PON Controller alarm history
:controller-alarm-profile/	CNTL-ALARM-CFG	PON Controller alarm profile configuration
:controller-auth-state/	CNTL-AUTH-STATE	MCMS Authenticator service state
:controller-engine-state/	CNTL-ENGINE-STATE	MCMS UMT Relay service state
:controller-log/	SYSLOG-CNTL	PON Controller log table
:controller-state/	CNTL-STATE	PON Controller state
:controller-stats/	STATS-CNTL	PON Controller statistics
:cpe-state/	CPE-STATE	CPE state (802.1X, DHCP, etc.)
:downstream-qos-map	DS-MAP-CFG	Downstream CoS/Pbit mapping to XGem
:olt/	OLT-CFG	OLT device configuration
:olt-alarm-history/	OLT-ALARM-HIST-STATE	OLT alarm history
:olt-alarm-profile/	OLT-ALARM-CFG	OLT alarm profile configuration
:olt-log/	SYSLOG-OLT	OLT log table
:olt-state/	OLT-STATE	OLT state
:olt-stats/	STATS-OLT	OLT statistics
:onu/	ONU-CFG	ONU configuration
:onu-alarm-history/	ONU-ALARM-HIST-STATE	ONU alarm history

:onu-alarm-profile/	ONU-ALARM-CFG	ONU alarm profile configuration
:onu-cpe-state/	ONU-CPE-STATE	CPE state per ONU (802.1X, DHCP, etc.)
:onu-log/	SYSLOG-ONU	ONU log table
:onu-state/	ONU-STATE	ONU state
:onu-stats/	STATS-ONU	ONU statistics
:sla-profile/	SLA-CFG	SLA profile configuration
:switch/	SWI-CFG	Switch configuration
:controller-acknowledge-alarms/	n/a	RPC: Acknowledge one or more alarms for a PON Controller
:controller-clear/	n/a	RPC: Clear statistics and logs for a PON Controller.
:controller-purge-alarms/	n/a	RPC: Purge one or more alarms from the PON Controller alarm history
:controller-set-status/	n/a	RPC: Set PON Controller status
:olt-acknowledge-alarms/	n/a	RPC: Acknowledge one or more alarms for an OLT device.
:olt-allow-onu-registration/	n/a	RPC: Allow ONU registration
:olt-clear/	n/a	RPC: Clear statistics and logs for an OLT
:olt-disable-onu/	n/a	RPC: Send Disable Serial Number to ONU
:olt-protection-arm/	n/a	RPC: Enable automatic PON protection
:olt-protection-switchover/	n/a	RPC: Force PON protection switchover
:olt-purge-alarms/	n/a	RPC: Purge one or more alarms from the OLT device alarm history
:olt-reset/	n/a	RPC: Reset OLT

:onu-acknowledge-alarms/	n/a	RPC: Acknowledge one or more alarms for an ONU device.
:onu-clear/	n/a	RPC: Clear statistics and logs for an ONU
:onu-purge-alarms/	n/a	RPC: Purge one or more alarms from the ONU device alarm history
:onu-reset/	n/a	RPC: Reset ONU

Broadband Forum YANG Models

Overview

Broadband Forum has a series of technical reports (TRs) that define standard YANG models for managing ITU access networks, including xPON, xDSL, and G.fast. TR-355 defines YANG models for managing xDSL and G.fast DPUs. TR-383 defines YANG models for managing network functions that are common to all access network types, including xPON, xDSL, and G.fast. TR-385 defines YANG models for configuring services for several variants of ITU PON including, ITU-T G.984.x G-PON, ITU-T G.989 NG-PON2, ITU-T G.987 XG-PON, and ITU-T-G.9807 XGS-PON devices. This section provides an overview of the BBF and other standard YANG models implemented by the MCMS Netconf Server for managing XGS-PON devices.

RFC 8343 A YANG Data Model for Interface Management

RFC 8343 ietf-interfaces is a generalized YANG model for managing network interfaces. The model is designed to be augmented (or extended) with interface type specific YANG models. For example, ieee802-ethernet-interface is a YANG model for configuring Ethernet interfaces, which augments ietf-interfaces with Ethernet specific attributes such as auto-negotiation, duplex, speed, and flow control. The ietf-interfaces model is leveraged by the BBF YANG models for managing OLT PON and uplink interfaces, ONU PON and UNI interfaces, and VLAN sub-interfaces.

TR-383 Common YANG Modules for Access Networks

TR-383 defines YANG models that are common across all access network types, including xPON, xDSL, and G.fast. TR-383 is used to manage common networking functions such as VLAN tagging and switching, MAC learning and forwarding, quality of service, multicast, DHCP Relay, and PPPoE. In the PON solution, TR-383 is used to configure VLAN matching and tagging operations on the OLT and ONU. TR-383 is also used to configure L2 Forwarders which are configured in conjunction with VLAN Subinterfaces to define the networks and datapath on the OLT.

TR-385 ITU-T PON YANG Modules

TR-385 defines YANG models for configuring services on ITU-T G.984.x G-PON, ITU-T G.989 NG-PON2, ITU-T G.987 XG-PON, and ITU-T-G.9807 XGS-PON devices. TR-385 is used to manage xPON related functions such as the OLT PON interfaces, T-CONTs, GEM Ports, Traffic Descriptor Profiles (upstream SLA) and xPON ONUs. In the PON solution, TR-385 is used to configure OLT PON Port attributes such as Discovery Period, encryption, FEC, PON ID and

other XGS-PON attributes. TR-385 is also used to manage OLT Service Ports (i.e., Links) through T-CONTs and GEM Port configuration. SLAs are configured through Traffic Descriptor Profiles in TR-385.

BBF Modules Overview

The MCMS Netconf Server supports TR-383 and TR-385 for configuring subscriber services on MicroPlug OLT devices and XGS-PON ONUs, including the MicroPlug ONU device and third party ONUs. An overview of the TR-383 and TR-385 configuration objects is shown in Figure 3 and the table below.

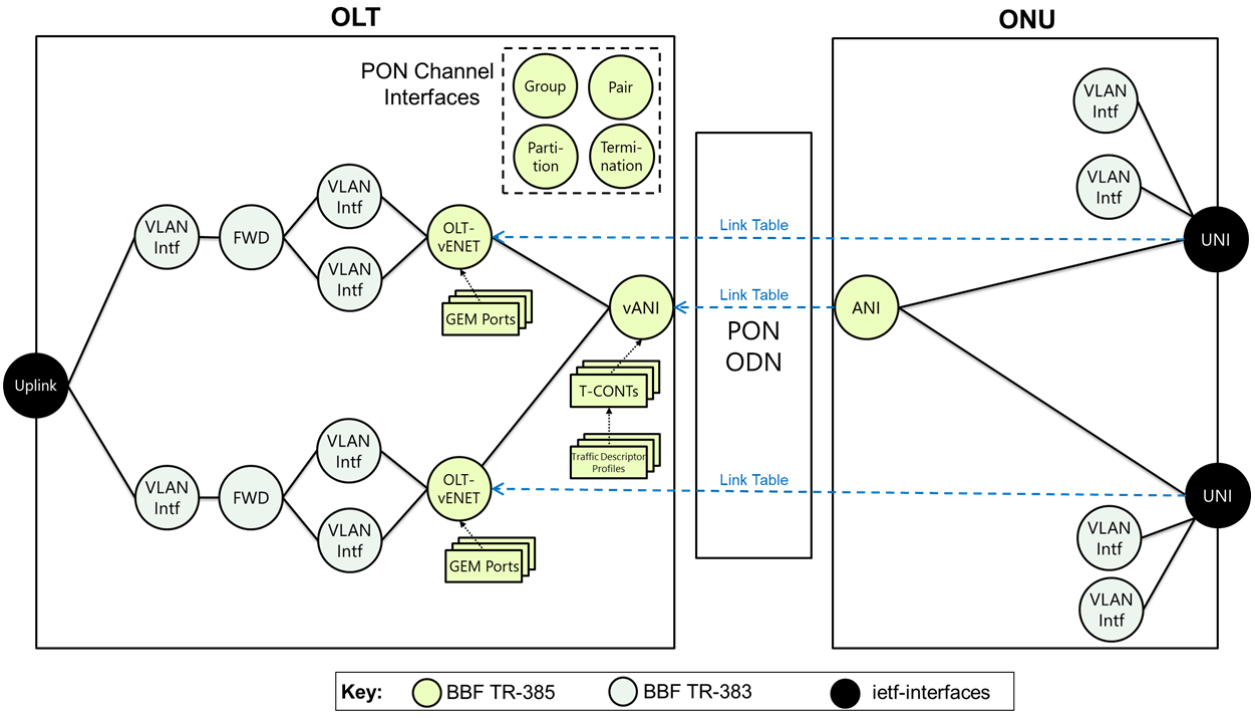


Figure 3 - BBF Modules Overview

Object	Description
ANI	Access Network Interface represents the PON interface on the ONU. The ANI is also used to configure the management channel for the ONU (e.g., ONU ID).
FWD	Forwarders represent a virtual switch instance within a device that connects two or more VLAN sub-interfaces. L2 forwarding and MAC learning are configured on forwarders. A forwarder indirectly represents a MicroPlug

	OLT L2 Switching Domain (L2SD).
GEM Ports	Management for GEM Port resources on OLT and ONU devices.
Link Table	Link Table entries define the relationship between OLT and ONU configuration objects. See Management Mode for more information.
OLT-vENET	OLT Virtual Ethernet Interface is an OLT configuration object that represents an ONU's UNI or VEIP configuration at an OLT.
ONU-vENET	ONU Virtual Ethernet Interface is an ONU configuration object for common VLAN and switching configuration that applies to all UNIs connected with a bridging function on the ONU.
PON Channel Interfaces	PON Channel Interfaces represents the PON port on a MicroPlug OLT. There are four logical channels configured for each PON.
T-CONTs	Management for T-CONT resources on OLT and ONU devices.
Traffic Descriptor Profile	Traffic descriptor profiles define the service level agreements for upstream traffic.
UNI	User Network Interface represents an Ethernet port or Virtual Ethernet Interface Port (VEIP) on an ONU.
Uplink	Uplink or Network-to-Network Interface (NNI) is the northbound connection from the OLT to the operator's network.
vANI	Virtual Access Network Interface is an OLT configuration object that represents an ONU's ANI at the OLT.
VLAN	VLAN subinterfaces define the VLAN matching and tagging operations (e.g., push and pop) on the OLT and ONU. On the OLT, VLAN sub-interfaces are connected to forwarding objects to define networks within the OLT.

Supported Modules

This section lists the BBF YANG and related IETF models supported by the MCMS Netconf Server.

The following BBF TR-383 YANG Models are supported:

- bbf-ethernet-performance-management@2017-05-08
  - ethernet
- bbf-hardware@2020-10-13
  - reset
- bbf-hardware-transceivers@2020-10-13
  - transceiver
  - transceiver-link
- bbf-interfaces-performance-management@2020-10-13
  - performance
- bbf-l2-dhcpv4-relay@2017-05-08
  - l2-dhcpv4-relay-profiles
- bbf-l2-forwarding@2020-10-13
  - forwarders
  - forwarding-databases
  - mac-learning-control-profiles
- bbf-qos-classifiers@2020-10-13
  - classifiers
- bbf-qos-policies@2020-10-13
  - policies
  - os-policy-profiles
- bbf-sub-interfaces@2020-10-13
  - vlan-sub-interface
- bbf-subscriber-profiles@2020-10-13
  - subscriber-profiles

The following BBF TR-385 YANG Models are supported:

- bbf-hardware-transceivers-xpon@2020-10-13
  - rssi-onu
- bbf-link-table@2020-10-13
  - link-table
- bbf-xpon@2020-10-13
  - channel-group
  - channel-pair
  - channel-partition
  - channel-termination
- bbf-xpon-defects@2020-10-13
- bbf-xpon-performance-management@2020-10-13
  - xpon
- bbf-xponani@2020-10-13
  - ani
  - onu-v-enet
- bbf-xponvani@2020-10-13
  - v-ani
  - olt-v-enet
- bbf-xpongemtcont@2020-10-13
  - traffic-descriptor-profiles
  - tconts
  - gemports
- bbf-xpongemtcont-gemport-performance-management@2020-10-13
  - performance
- bbf-xpon-onu-state@2020-10-13
  - ous-present-on-local-channel-termination

The following IETF YANG Models are supported:

- RFC 8343 ietf-interfaces@2018-02-20
  - interfaces
  - Interfaces-state
- RFC 8348 ietf-hardware@2018-03-13
  - hardware

BBF YANG, Tibit YANG, and Other Management Interfaces

The BBF YANG interface operates in parallel with the Tibit YANG interface and the [MCMS PON Manager] Web Client. Configuration changes though BBF YANG update configuration records in the PON Controller database (OLT-CFG, ONU-CFG, and SLA-CFG) in MongoDB,

and are reflected in both the Tibit YANG and PON Manager Web interfaces. Likewise, configuration changes made through Tibit YANG and PON Manager are reflected in the BBF YANG interface.

BBF Configuration Persistence

The MCMS Netconf Server uses MongoDB for persistent storage for BBF YANG configuration. By default, BBF YANG related configuration is stored in the 'tibit_netconf' database in MongoDB. See Database Connection Configuration for more information on configuring the 'tibit_netconf' database. The list of MongoDB collections in the NETCONF database is shown in Figure 4.

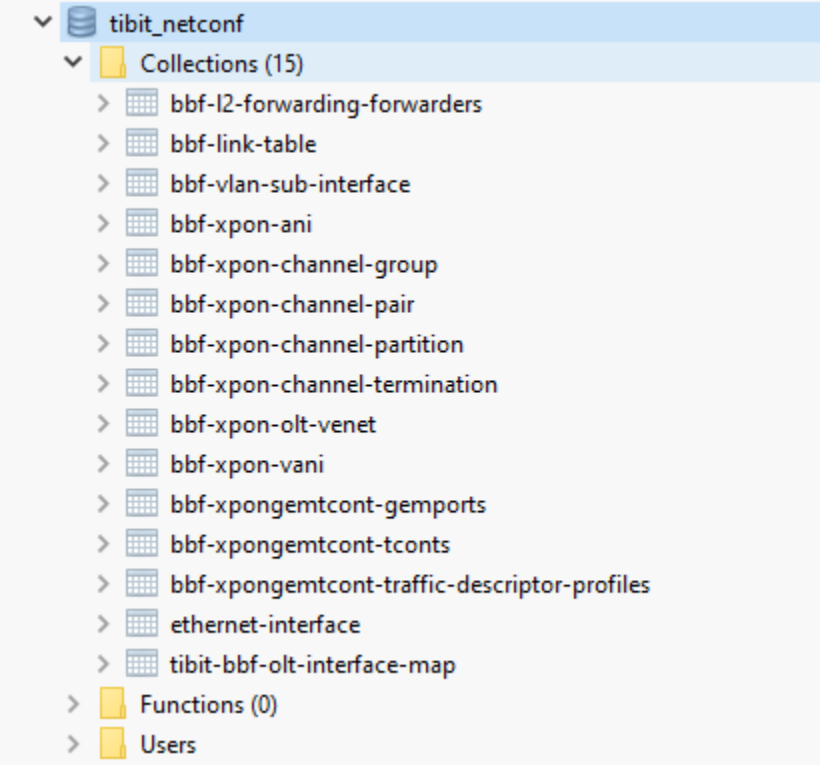


Figure 4 - NETCONF Database

The NETCONF database is managed exclusively by the MCMS Netconf Server. Unlike the PON Controller database (tibit_pon_contrller), the NETCONF database is not intended to be managed externally. The database is not manageable through PON Manager and must not be modified directly through MongoDB. Changes to the NETCONF database through MongoDB will not be recognized by the NETCONF interface and will be ignored and overwritten by subsequent NETCONF <edit-config> requests.

Management Mode

TR-385 defines two management modes for representing OLT and ONU devices as network elements to higher layer applications. In Combined NE-Mode, the OLT and subtended ONUs connected to the OLT appear as a single network element to applications. There is a single NETCONF server for managing the OLT and ONUs. ONUs are managed through the OLT's NETCONF server.

In Separated NE-Mode, the OLT and ONUs are managed as separate network elements. There is a NETCONF server for the OLT and one or more NETCONF servers for the ONUs, where the granularity ranges from one NETCONF server for all ONUs to one NETCONF server per ONU. A use case for Separated NE-Mode is to support Cloud Central Office (CloudCO) where ONU management functions are virtualized in the cloud. A Distribution Point Unit (DPU) is another use case for Separated NE-Mode, which is a larger device deployed at or near the premise that aggregates multiple customers.

The NETCONF/YANG solution supports Combined NE-Mode. Separated NE-Mode is not supported.

BBF YANG Configuration Details

This section describes the BBF YANG configuration tables and attributes that are supported by the MCMS Netconf Server in more detail. Additional details are also provided for the PON implementation and how the NETCONF server applies the configuration to OLT and ONU devices.

ANI

The Access Network Interface (ANI) represents the physical ONU device in the TR-385 YANG model. An ANI interface must be created for each ONU device. However, the ANI configuration attributes (onu-id, management-gemport-id, etc.) are ignored because the NETCONF Server is operating in Combined NE-Mode. Instead, the ONU PON and management Channel (OMCC) attributes are configured using the vANI. See Management Mode for information on Combined NE-Mode. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xponani:ani	
name	There are no restrictions on the interface configuration name.
type	The type must be set to bbf-xpon-if-type:ani for ANI interfaces.

The following XML is an example of configuring an ANI interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <interfaces
          xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
          xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type">
          <interface>
            <name>ani-ALPHe30cadcf</name>
            <type>bbf-xponift:ani</type>
          </interface>
        </interfaces>
      </config>
   </edit-config>
```

```
</rpc>
```

GEM Port

GEM Port configuration maps to an OLT Service Port in the PON Solution (i.e., per-Link/T-CONT configuration). The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-xpongemtcont:xpongemtcont/bbf-xpongemtcont:gemports	
name	There are no restrictions on the GEM Port name.
gemport-id	The GEM Port ID to use for this XGem Port (i.e., OLT Service Port). If the value is not specified, the NETCONF Server automatically assigns a Port ID for this XGem Port, and updates the ONU Inventory in the OLT based on the value specified for this attribute. Note that the OLT MicroPlug implements a shared number space between T-CONT Alloc-IDs and GEM Port IDs. Configurations require that at least one GEM Port ID value matches a corresponding T-CONT Alloc-ID value. See T-CONT and GEM Port Mapping for more information on how T-CONT and GEM Port configuration is mapped to hardware resources. **MongoDB Ref:** OLT-CFG.ONUs.<ONU SSN>.OLT-Service **MongoDB Ref:** ONU-CFG.OLT-Service.Service Reference
interface	Must reference an existing OLT-vENET interface.
traffic-class	The traffic class is used in conjunction with bbf-qos-classifiers and bbf-qos-policies to map traffic flows (e.g., VLANs) to specific T-CONTs and GEM Ports.
downstream-aes-indicator	The PON Solution currently does not support configuring AES encryption on a per-ONU basis. Moreover, the PON Solution does not support configuring upstream encryption separately from downstream. Encryption can be configured on a per OLT PON Port basis using Tibit YANG.
upstream-aes-indicator	See description for downstream-aes-indicator.
tcont-ref	Must reference an existing T-CONT configuration.

The following XML is an example of configuring a GEM Port.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <xpongemtcont xmlns="urn:bbf:yang:bbf-xpongemtcont">
          <gemports>
            <gemport>
              <name>gem-ALPHe30cadcf-eth.5-tc0</name>
              <interface>olt-v-enet-ALPHe30cadcf.5</interface>
              <tcont-ref>vani-ALPHe30cadcf-tcont.1</tcont-ref>
            </gemport>
          </gemports>
        </xpongemtcont>
      </config>
   </edit-config>
</rpc>
```

L2 DHCPv4 Relay

Note: the L2 DHCPv4 Relay and Subscriber Profile configuration only apply when the DHCPv4 Host Processing function is enabled in the PON Controller.

TR-383 supports configuration for an L2 DHCPv4 Relay Agent that inserts and removes Option 82 as defined by RFC 3046 DHCP Relay Agent Information Option. The L2 DHCPv4 Relay supports Option 82 only, and does not provide the “ip-helper” broadcast to unicast IP function that is typically available with a full L3 DHCPv4 Relay Agent. When the PON Controller DHCPv4 Host Processing function is being used, the Option 82 Circuit-ID and Remote-ID can be configured using TR-383.

TR-383 divides the DHCPv4 Relay Agent configuration into multiple sections. A L2 DHCPv4 Relay Profile defines the Option 82 format that determines which Option 82 suboptions to insert in DHCP messages received from the client. Typically, there is one L2 DHCPv4 Relay Profile configured for the system. A Subscriber Profile defines the Option 82 Circuit ID and Remote values for a particular ONU configuration. A Subscriber Profile must be configured for each ONU that requires DHCPv4 Relay Option 82. The L2 DHCPv4 Relay Profile and Subscriber Profile are associated to a specific ONU through PON Network OLT VLAN Subinterface configured for the ONU.

L2 DHCPv4 Relay Profile

The L2 DHCPv4 Relay Profile defines the format of Option 82 that is inserted into DHCP messages originating from the DHCP Client. This profile defines the Option 82 format only. Values for Circuit ID and Remote ID are defined in a Subscriber Profile. The L2 DHCPv4 Relay Profile is referenced by the PON Network OLT VLAN Subinterface configured for the subscriber. This reference associates the Option 82 format to a specific ONU configuration. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/l2-dhcpv4-relay-profile:l2-dhcpv4-relay-profiles	
name	There are no restrictions on the DHCPv4 Relay Profile configuration name.
option82-format	List of Option 82 sub-options to insert into DHCP messages originating from the client. The Netconf Server supports ‘circuit-id’ and ‘remote-id’ sub-options only. Other options are ignored by the server.

The following XML is an example of configuring an L2 DHCP Relay Profile.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
         <l2-dhcpv4-relay-profiles xmlns="urn:bbf:yang:bbf-l2-dhcpv4-relay">
            <l2-dhcpv4-relay-profile>
               <name>dhcp-relay-prof1</name>
               <option82-format>
                  <suboptions>circuit-id</suboptions>
                  <suboptions>remote-id</suboptions>
               </option82-format>
            </l2-dhcpv4-relay-profile>
         </l2-dhcpv4-relay-profiles>
      </config>
   </edit-config>
</rpc>
```

Subscriber Profile

The Option 82 Circuit ID and Remote ID values for a specific subscriber are configured using a Subscriber Profile. One Subscriber Profile is configured for each subscriber configuration that requires DHCPv4 Relay Option 82. The Subscriber Profile is referenced by the PON Network

OLT VLAN Subinterface configured for the subscriber. This reference associates the Option 82 values to a specific ONU configuration. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/l2-dhcpv4-relay-profile:l2-dhcpv4-relay-profiles	
name	There are no restrictions on the Subscriber Profile name.
circuit-id	Option 82 Circuit ID as defined by RFC 3046 DHCP Relay Agent Information Option. **MongoDB Ref:** ONU-CFG.OLT-Service.DHCP.Circuit ID
remote-id	Option 82 Remote ID as defined by RFC 3046 DHCP Relay Agent Information Option. **MongoDB Ref:** ONU-CFG.OLT-Service.DHCP.Remote ID

The following XML is an example of configuring an L2 DHCP Relay Profile.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
         <subscriber-profiles xmlns="urn:bbf:yang:bbf-subscriber-profiles">
            <subscriber-profile>
               <name>ALPHe30cadcf</name>
               <circuit-id>tibit pon 1/0/1:vlan200.25</circuit-id>
               <remote-id>ALPHe30cadcf</remote-id>
            </subscriber-profile>
         </subscriber-profiles>
      </config>
   </edit-config>
</rpc>
```

L2 Forwarder

A TR-383 Forwarder represents a virtual switch or virtual LAN instance within the OLT that provides network bridging functions, including MAC learning, L2 forwarding, broadcast flooding and multicast replication. A Forwarder represents an L2 Switch Domain (L2SD) in a PON OLT MicroPlug. VLAN Subinterfaces are connected through forwarders to establish the datapath through the OLT. Point-to-point services (1:1) are configured with exactly two VLAN

Subinterfaces (one NNI and one PON) connected to a single Forwarder. Multipoint services (1:N) are configured with exactly one NNI VLAN Subinterface, one or more PON VLAN Subinterfaces, all connected to a single Forwarder. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-l2-forwarding:forwarding/bbf-l2-forwarding:forwarders	
name	There are no restrictions on the forwarder configuration name.
ports	List of VLAN Subinterfaces connected through this forwarder. There must be exactly one NNI VLAN Subinterface configured in each forwarder. For point-to-point (1:1) services, there must be exactly one PON VLAN Subinterface configured in the forwarder. For multipoint (1:N) services, there should be one or more PON VLAN Subinterface configured in the forwarder.
l2-dhcpv4-relay	Presence of this node enables DHCPv4 Protocol Filtering over UMT for the NNI Network (L2SD). **MongoDB Ref:** OLT-CFG.NNI Networks.Filter.DHCPv4

The following XML is an example of configuring an L2 Forwarder.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <forwarding
               xmlns="urn:bbf:yang:bbf-l2-forwarding"
               xmlns:bbf-l2-d4r-fwd="urn:bbf:yang:bbf-l2-dhcpv4-relay-forwarding">
          <forwarders>
            <forwarder>
              <name>ALPHe30cadcf_s200.c25.c0</name>
              <ports>
                <!-- NNI VLAN Subinterface (ONU-CFG.OLT-Service.NNI Network) -->
                <port>
                  <name>network1.s200.c25.c0</name>
                  <sub-interface>network1.s200.c25.c0</sub-interface>
                </port>
                <!-- PON-size VLAN Subinterface (ONU-CFG.OLT-Service.PON Network) -->
                <port>
                  <name>olt-ALPHe30cadcf-eth.5.25</name>
                  <sub-interface>olt-ALPHe30cadcf-eth.5.25</sub-interface>
```

```
              </port>
            </ports>
            <!-- Enable DHCP Relay -->
            <bbf-l2-d4r-fwd:l2-dhcpv4-relay/>
          </forwarder>
        </forwarders>
      </forwarding>
    </config>
  </edit-config>
</rpc>
```

L2 Forwarding Database

A TR-383 Forwarding Database, in conjunction with a Forwarder and MAC Learning Control Profile, is used to configure a virtual switch or virtual LAN instance within the OLT that provides network bridging functions, including MAC learning, L2 forwarding, broadcast flooding and multicast replication. A Forwarding Database represents the MAC address learning function of an L2 Switch Domain (L2SD) in a PON OLT MicroPlug, which is used to support multipoint services (1:N). The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-l2-forwarding:forwarding-databases/bbf-l2-forwarding:forwarding-database	
name	There are no restrictions on the forwarding database configuration name.
aging-timer	Timeout value used to age out and remove inactive CPE MAC addresses from the learning table. **MongoDB Ref:** OLT-CFG.MAC Learning.Age Limit
max-number-mac-addresses	MAC address table size, which is the maximum number of CPE MAC addresses able to be learned for an NNI Network (L2SD). **MongoDB Ref:** OLT-CFG.NNI Networks.Learning Limit

The following XML is an example of configuring an L2 Forwarding Database.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
```

```
        <forwarding xmlns="urn:bbf:yang:bbf-l2-forwarding">
          <forwarding-databases>
            <forwarding-database>
              <name>ALPHe30cadcf_s200.c25.c0</name>
              <max-number-mac-addresses>2046</max-number-mac-addresses>
              <aging-timer>300</bbf-l2-fwd:aging-timer>
              <mac-learning-control>
                <mac-learning-control-profile>learning-profile</mac-learning-control-profile>
              </mac-learning-control>
            </forwarding-database>
          </forwarding-databases>
        </forwarding>
      </config>
   </edit-config>
</rpc>
```

L2 MAC Learning Control Profile

A TR-383 MAC Learning Control Profile, in conjunction with a Forwarder and Forwarding Database, is used to configure a virtual switch or virtual LAN instance within the OLT that provides network bridging functions, including MAC learning, L2 forwarding, broadcast flooding and multicast replication. A MAC Leaning Control Profile is used to configure MAC Move for an L2 Switch Domain (L2SD) in a PON OLT MicroPlug. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-l2-forwarding:mac-learning-control-profiles/mac-learning-control-profile	
name	There are no restrictions on the MAC Learning Control Profile name.
mac-learning-rule/ receiving-interface-usage, mac-can-not-move-to	Configures MAC Move for an OLT device. The receiving-interface-usage field selects the type of receive port UNI vs. NNI. The only valid value for the PON solution is ‘user-port’. ***NOTE:*** *BBF YANG configures MAC Move per NNI Network. However, the PON solution supports configuring MAC Move per OLT device. This configuration applies to all forwarders configured on an OLT device. Therefore, the same MAC Learning Control Profile must be used for all forwarders on a specific OLT device.* **MongoDB Ref:** OLT-CFG.MAC Learning.Allow CPEs To Move

The following XML is an example of configuring an L2 MAC Learning Control Profile.

```
<rpc
```

```
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <forwarding xmlns="urn:bbf:yang:bbf-l2-forwarding">
          <mac-learning-control-profiles>
            <mac-learning-control-profile>
              <name>learning-profile/bbf-l2-fwd:name>
                  <mac-learning-rule>
                     <receiving-interface-usage>user-port</receiving-interface-usage>
                     <mac-can-not-move-to>user-port</mac-can-not-move-to>
                  </mac-learning-rule>
               </mac-learning-control-profile>
            </mac-learning-control-profiles>
        </forwarding>
      </config>
   </edit-config>
</rpc>
```

Link Table

OLT and ONUs are managed as separate network elements when operating in Separated NE-Mode. This requires applications to configure XGS-PON resources such as T-CONTs and GEM Ports on both the OLT and the ONU separately. However, the OLT configures these resources on behalf of the ONU when operating in Combined NE-Mode. In order to share the T-CONT and GEM Port configuration, the OLT configuration objects must be associated with ONU objects. The bbf-link-table YANG model is used to define the relationship between OLT and ONU interfaces, where entries are created to associate OLT vANIs with ONU ANIs and to associate OLT-vENETs with ONU UNIs or VEIPs. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-link-table:link-table	
from-interface	Reference to an ONU interface, such as an ANI, UNI, or ONU-vENET.
to-interface	Reference to the virtual interface on the OLT, such as a vANI or OLT-vENET interface.

The following XML is an example of creating an entry in the Link Table.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <link-table xmlns="urn:bbf:yang:bbf-link-table">
          <link-table>
            <from-interface>ani-ALPHe30cadcf</from-interface>
            <to-interface>vani-ALPHe30cadcf</to-interface>
          </link-table>
          <link-table>
            <from-interface>onu-ALPHe30cadcf-eth.5</from-interface>
            <to-interface>olt-v-enet-ALPHe30cadcf.5</to-interface>
          </link-table>
      </config>
   </edit-config>
</rpc>
```

OLT NNI Interface

The MicroPlug OLT NNI interface is not directly defined by the BBF YANG models. The MCMS Netconf Server represents each OLT NNI as a standard 10G Ethernet interface with ietf-interfaces type iana-if-type:ethernetCsmacd.

In the current MCMS Netconf Server implementation, there are no configurable attributes for an NNI. However, the NNI must be created to and connected to VLAN Subinterfaces and Forwarders to establish a forwarding path through the OLT. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface	
name	There are no restrictions on the NNI interface configuration name.
type	The type must be set to iana-if-type:ethernetCsmacd for NNI interfaces.

There is no explicit mapping from an NNI configuration to a OLT MicroPlug device using standard YANG models. An NNI interface configuration is associated with an OLT through the Tibit YANG olt-interface-map entry. See OLT Device Mapping for more information on how NNI interface configuration is mapped to OLT devices.

The following XML is an example of configuring an NNI interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <if:interfaces
            xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
            xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
            <!-- Create the NNI uplink interface -->
            <if:interface>
              <if:name>uplink.1/0/1</if:name>
              <if:type>ianaift:ethernetCsmacd</if:type>
            </if:interface>
        </if:interfaces>
      </config>
   </edit-config>
</rpc>
```

OLT PON Interfaces

The TR-385 OLT PON interface configuration model is based on the NG-PON2 architecture and is compatible with G-PON, XG-PON, and XGS-PON. The NETCONF/YANG Solution only supports XGS-PON, which requires less configuration than the more complex NG-PON2. Therefore, several attributes do not apply for XGS-PON. TR-385 defines specific hard-coded values for attributes that do not apply to XGS-PON.

PON port configuration is split across four logical interfaces, as defined in ITU-T G.989.

Logical Interface	Description
Channel Group	A set of channel pairs carried over a common fiber.
Channel Pair	A set of one downstream wavelength channel and one upstream wavelength channel that provides connectivity between an OLT and one or more ONUs.
Channel Partition	Any of the operator-specified non-overlapping subsets of TWDM or PtP WDM channels in an NG-PON2 system.
Channel Termination	A logical reference to an OLT PON interface that terminates a channel pair in an NG-PON2 system. For XGS-PON, the channel termination refers to a logical function associated

	with an OLT port that terminates an XGS-PON.

There is no explicit mapping from PON interface configuration to a MicroPlug OLT device using standard YANG models. PON interface configuration is associated with an OLT through the Tibit YANG olt-interface-map entry. See OLT Device Mapping for more information on how PON interface configuration is mapped to OLT devices.

Configuration for each of the logical interfaces are described in the sections that follow. Note that the examples in these sections show each logical interface configured independently from the other logical interfaces. TR-385 defines each of the logical interfaces such that the channel interfaces cross reference one or more of the other logical channel interfaces, which makes it impossible to create them independently. In practice, all four logical interfaces must be created in the same NETCONF request.

Channel Group

A Channel Group interface must be created for each OLT MicroPlug. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xpon:channel-group	
name	There are no restrictions on the interface configuration name.
type	The type must be set to bbf-xpon-if-type:channel-group for PON Channel Group interfaces.
polling-period	Amount of time between discovery slots on the PON. A setting of '0' will disable discovery. **MongoDB Ref:** OLT-CFG.GPON.Discovery Period
pon-pool	A PON pool must be created for a channel group. However, NETCONF/YANG does not support configurable ranges for ONU IDs, T-CONT IDs, and GEM Port IDs. These attributes are ignored.

The following XML is an example of configuring a PON Channel Group interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
```

```
      <target>
         <running/>
      </target>
      <config>
        <if:interfaces
            xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
            xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type"
            xmlns:bbf-xpon="urn:bbf:yang:bbf-xpon">
            <!-- Create the PON Channel Group interface -->
            <if:interface>
              <if:name>channelgroup.1/0/1</if:name>
              <if:type>bbf-xponift:channel-group</if:type>
              <bbf-xpon:channel-group>
              <bbf-xpon:pon-pools>
                <bbf-xpon:pon-pool>
                    <bbf-xpon:name>pool1</bbf-xpon:name>
                    <bbf-xpon:channel-termination-ref>
                        channeltermination.1/0/1
                    </bbf-xpon:channel-termination-ref>
                </bbf-xpon:pon-pool>
                </bbf-xpon:pon-pools>
              </bbf-xpon:channel-group>
            </if:interface>
        </if:interfaces>
      </config>
   </edit-config>
</rpc>
```

Channel Partition

A Channel Partition interface must be created for each MicroPlug OLT. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xpon:channel-partition	
name	There are no restrictions on the interface configuration name.
type	The type must be set to bbf-xpon-if-type:channel-partition for PON Channel Partition interfaces.
closest-onu-distance	Configures the fiber reach for an OLT device. In conjunction with maximum-differential-xpon-distance, closest-onu-distance is used to configure the fiber reach for an OLT device. The only supported configurations are 0..20km and 20..40km. BBF YANG does not support configuration for 0..10km.

	For 0..20km: closest-onu-distance = 0 maximum-differential- xpon-distance = 20 For 20..40km: closest-onu-distance = 20 maximum-differential- xpon-distance = 20 **MongoDB Ref:** OLT-CFG.OLT.Max Round Trip Time
downstream-fec	Enable FEC on a PON port. Note that the BBF YANG models assume that disabling FEC does not apply to XGS-PON and that FEC is always enabled. However, the OLT MicroPlug supports disabling FEC on the PON. The MCMS Netconf Server applies this attribute to both Downstream and Upstream FEC on the OLT. **MongoDB Ref:** OLT-CFG.GPON.Downstream Fec, OLT-CFG.GPON.Upstream FEC 0
maximum-differential-xpon-distance	See closest-onu-distance above.

The following XML is an example of configuring a PON Channel Partition interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <if:interfaces
            xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
            xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type"
            xmlns:bbf-xpon="urn:bbf:yang:bbf-xpon">
            <!-- Create the PON Channel Partition interface -->
            <if:interface>
              <if:name>channelpartition.1/0/1</if:name>
              <if:type>bbf-xponift:channel-partition</if:type>
              <bbf-xpon:channel-partition>
                <bbf-xpon:channel-group-ref>
                    channelgroup.1/0/1
                </bbf-xpon:channel-group-ref>
                <bbf-xpon:channel-partition-index>0</bbf-xpon:channel-partition-index>
                <bbf-xpon:closest-onu-distance>0</bbf-xpon:closest-onu-distance>
                <bbf-xpon:maximum-differential-xpon-distance>
                    20
                </bbf-xpon:maximum-differential-xpon-distance>
                <bbf-xpon:authentication-method>serial-number</bbf-xpon:authentication-method>
              </bbf-xpon:channel-partition>
```

```
            </if:interface>
          </if:interfaces>
        </config>
    </edit-config>
</rpc>
```

Channel Pair

A Channel Pair interface must be created for each MicroPlug OLT. However, none of the Channel Pair configuration attributes apply to the OLT. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xpon:channel-pair	
name	There are no restrictions on the interface configuration name.
type	The type must be set to bbf-xpon-if-type:channel-pair for PON Channel Pair interfaces.

The following XML is an example of configuring a PON Channel Pair interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <if:interfaces
            xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
            xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type"
            xmlns:bbf-xpon="urn:bbf:yang:bbf-xpon"
            xmlns:bbf-xpon-types="urn:bbf:yang:bbf-xpon-types">
            <!-- Create the PON Channel Pair interface -->
            <if:interface>
              <if:name>channelpair.1/0/1</if:name>
              <if:type >bbf-xponift:channel-pair</if:type>
              <bbf-xpon:channel-pair>
                <bbf-xpon:channel-partition-ref>
                    channelpartition.1/0/1
                </bbf-xpon:channel-partition-ref>
                <bbf-xpon:channel-group-ref>channelgroup.1/0/1</bbf-xpon:channel-group-ref>
                <bbf-xpon:channel-pair-type>bbf-xpon-types:xgs</bbf-xpon:channel-pair-type>
              </bbf-xpon:channel-pair>
            </if:interface>
```

```
        </if:interfaces>
      </config>
   </edit-config>
</rpc>
```

Channel Termination

A Channel Termination interface must be created for each OLT MicroPlug. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xpon:channel-termination	
name	There are no restrictions on the interface configuration name.
type	The type must be set to bbf-xpon-if-type:channel-termination for PON Channel Termination interfaces.
enable	Enables the PON interface on the MicroPlug OLT. **MongoDB Ref:** OLT-CFG.OLT.PON Enable
xgs-pon-id	Unique 32-bit value assigned to each PON. **MongoDB Ref:** OLT-CFG.GPON.PON ID

The following XML is an example of configuring a PON Channel Termination interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <if:interfaces
            xmlns:if="urn:ietf:params:xml:ns:yang:ietf-interfaces"
            xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type"
            xmlns:bbf-xpon="urn:bbf:yang:bbf-xpon"
            xmlns:bbf-xpon-types="urn:bbf:yang:bbf-xpon-types">
            <!-- Create the PON Channel Termination interface -->
            <if:interface>
              <if:name>channeltermination.1/0/1</if:name>
              <if:type>bbf-xponift:channel-termination</if:type>
              <if:enabled>true</if:enabled>
              <bbf-xpon:channel-termination>
                <bbf-xpon:channel-pair-ref>channelpair.1/0/1</bbf-xpon:channel-pair-ref>
                <bbf-xpon:channel-termination-type>
```

```
                    bbf-xpon-types:xgs
                  </bbf-xpon:channel-termination-type>
                </bbf-xpon:channel-termination>
              </if:interface>
            </if:interfaces>
          </config>
      </edit-config>
</rpc>
```

OLT-vENET

The OLT Virtual Ethernet Interface is an OLT management object that represents an ONU-vENET, UNI, or VEIP on an ONU. An OLT-vENET interface is associated with an ONU interface through an entry in the Link Table.

The OLT-vENET must be created and connected to VLAN subinterfaces and forwarders to establish a forwarding path through the OLT. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xponvani:olt-v-enet	
name	There are no restrictions on the OLT-vENET interface configuration name.
type	The type must be set to bbf-xpon-if-type:olt-v-enet for OLT-vENET interfaces.
lower-layer-interface	Must reference an existing vANI configuration.

The following XML is an example of configuring an OLT-vENET interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <interfaces
          xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
          xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type">
          <interface>
            <name>olt-v-enet-ALPHe30cadcf.5</name>
            <type>bbf-xponift:olt-v-enet</type>
```

```
            <olt-v-enet xmlns="urn:bbf:yang:bbf-xponvani">
              <lower-layer-interface>vani-ALPHe30cadcf</lower-layer-interface>
            </olt-v-enet>
          </interface>
        </interfaces>
      </config>
   </edit-config>
</rpc>
```

ONU-vENET

The ONU Virtual Ethernet Interface is an ONU management object for common VLAN and switching configuration that applies to all UNIs connected through a bridging function on an ONU. An ONU-vENET interface is associated with an ONU through an ANI interface reference. An ONU-vENET interface is associated with an OLT-vENET through an entry in the Link Table.

VLAN Subinterfaces are configured off of a ONU-vENET interface, which define the VLAN matching and modification rules applied to frames forwarded through the ONU. The ONU-vENET VLAN configuration is automatically applied to all UNIs that exist on the ONU. There is no additional configuration to associate ONU-vENET configuration to specific UNIs or a sub-set of UNIs on the ONU. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xponani:onu-v-enet	
name	There are no restrictions on the ONU-vENET interface configuration name.
type	The type must be set to bbf-xpon-if-type:onu-v-enet for ONU-vENET interfaces.
ani	Must reference an existing ANI configuration.

The following XML is an example of configuring an ONU-vENET interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
```

```
        <interfaces
          xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
          xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type">
          <interface>
            <name>onu-v-enet-ALPHe30cadcf</name>
            <type>bbf-xponift:onu-v-enet</type>
            <onu-v-enet xmlns="urn:bbf:yang:bbf-xponani">
              <ani>ani-ALPHe30cadcf</ani>
            </onu-v-enet>
          </interface>
        </interfaces>
      </config>
   </edit-config>
</rpc>
```

QoS Policy Profiles

QoS Policy Profile configuration classifies frames by matching on header elements such as VLAN priority bits, DSCP, and addresses to actions such as color marking, filters/dropping, and setting scheduling traffic class. The MCMS Netconf Server only supports QoS Policy Profiles for the purposes of classifying traffic to GEM Ports, where frames are classified by VLAN priority bits to scheduling traffic classes (i.e., GEM Ports). Classification matching on DSCP and other frame header elements is not currently supported. Other classification actions such as color marking and filtering are not currently supported.

The Netconf Server applies QoS Policy Profile configuration depending on the type of device. For OLT devices, a QoS Policy Profile is used to configure DS-MAP-CFG in MongoDB, which defines the mapping between VLAN Tag priority bits and OLT-Service Ports for downstream frames. For ONU devices, a QoS Policy Profile is used to configure the IEEE 802.1p Mapper Service Profile, which defines the mapping between priority and GEM Ports for frames forwarded through the ONU. The following sections describe the tables that make up the QoS Policy Profile and how the Netconf Server applies the QoS Policy Profile to configuring the PON solution.

Classifiers

A Classifier defines the set of rules that match frames to specific QoS actions. Classifiers are configured individually and referenced by a QoS Policy which is further referenced by a QoS Policy Profile. The match-criteria types ‘in-pbit-list’ and ‘match-all’ are the only types of classifier match supported by the Netconf Server. The action-type ‘scheduling-traffic-class’ is the only classifier action supported by the Netconf Server. The MCMS Netconf Server applies the configuration attributes as described in the table below. Note that classifier rules are applied differently for OLT and ONU.

Attribute	Description
/bbf-qos-classifiers:classifiers/bbf-qos-classifiers:classifier-entry	
name	Name identifying the classifier entry.
match-criteria	
match-all	Classifier matches all frames.
tag/in-pbit-list	Classifier matches specific VLAN priority bit values. **OLT MongoDB Ref:** DS-MAP-CFG.Mapping.COS; ONU-CFG.OLT-Service.DS-MAP-CFG **ONU MongoDB Ref:** SRV-CFG.OMCI.Ieee8021pMapperServiceProfile.Interwork_tp_pointer_for_p_bit_priority
classifier-action-entry-cfg	
action-type	Must be set to ‘scheduling-traffic-class’.
scheduling-traffic-class	Traffic class value that matches a GEM Port traffic-class. See GEM Ports. **OLT MongoDB Ref:** DS-MAP-CFG.Mapping.OLT-Service Offset; ONU-CFG.OLT-Service.DS-MAP-CFG **ONU MongoDB Ref:** SRV-CFG.OMCI.Ieee8021pMapperServiceProfile.Interwork_tp_pointer_for_p_bit_priority

The following XML is an example of configuring a Classifier entry.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
         <bbf-qos-cls:classifiers xmlns:bbf-qos-cls="urn:bbf:yang:bbf-qos-classifiers">
            <!-- QoS Classifier for tc0 -->
            <bbf-qos-cls:classifier-entry>
               <bbf-qos-cls:name>onu-ingress-classifier-tc-0</bbf-qos-cls:name>
               <bbf-qos-cls:description>ONU ingress classifier for tc0</bbf-qos-cls:description>
               <bbf-qos-cls:filter-operation>
                      bbf-qos-cls:match-any-filter
```

```
            </bbf-qos-cls:filter-operation>
            <bbf-qos-cls:match-criteria>
               <bbf-qos-cls:tag>
                  <bbf-qos-cls:index>0</bbf-qos-cls:index>
                  <bbf-qos-cls:in-pbit-list>0,1</bbf-qos-cls:in-pbit-list>
               </bbf-qos-cls:tag>
            </bbf-qos-cls:match-criteria>
            <bbf-qos-cls:classifier-action-entry-cfg>
                   <bbf-qos-cls:action-type>
                          bbf-qos-cls:scheduling-traffic-class
                   </bbf-qos-cls:action-type>
                   <bbf-qos-cls:scheduling-traffic-class>
                          0
                   </bbf-qos-cls:scheduling-traffic-class>
            </bbf-qos-cls:classifier-action-entry-cfg>
         </bbf-qos-cls:classifier-entry>
      </bbf-qos-cls:classifiers>
   </config>
</edit-config>
</rpc>
```

Policy Profiles

A QoS Policy Profile defines the mapping of frames to GEM Ports. It defines a set of classification rules that match on VLAN Tag p-bits and select a destination traffic-class (or GEM Port). Other uses for QoS Policy Profiles defined by TR-383 are not supported, such as color marking and filtering.

A QoS Policy associates a set of classifiers into a group. A Policy is an intermediate management object that associates a set of classifiers with a profile. Each Policy contains a set of Classifiers and is referenced by QoS Policy Profile.

The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-qos-policies:policies/bbf-qos-classifiers:policy	
name	Name identifying the policy.
classifiers	List of classifiers associated with this policy
/bbf-qos-policies:qos-policy-profiles/bbf-qos-classifiers:policy-profile	
name	Name identifying the policy profile.

policy-list	List of policies associated with this profile. Currently, the Netconf Server only supports one policy per profile.

The following XML is an example of configuring a QoS Policy and QoS Policy Profile.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
         <!-- ONU QoS Policy -->
         <bbf-qos-pol:policies xmlns:bbf-qos-pol="urn:bbf:yang:bbf-qos-policies">
            <bbf-qos-pol:policy>
               <bbf-qos-pol:name>onu-ingress-qos-policy</bbf-qos-pol:name>
               <bbf-qos-pol:description>
                      ONU QoS policy mapping pbits to traffic classes
               </bbf-qos-pol:description>
               <bbf-qos-pol:classifiers>
                  <bbf-qos-pol:name>onu-ingress-classifier-tc-0</bbf-qos-pol:name>
               </bbf-qos-pol:classifiers>
               <bbf-qos-pol:classifiers>
                  <bbf-qos-pol:name>onu-ingress-classifier-tc-1</bbf-qos-pol:name>
               </bbf-qos-pol:classifiers>
               <bbf-qos-pol:classifiers>
                  <bbf-qos-pol:name>onu-ingress-classifier-tc-2</bbf-qos-pol:name>
               </bbf-qos-pol:classifiers>
               <bbf-qos-pol:classifiers>
                  <bbf-qos-pol:name>onu-ingress-classifier-tc-3</bbf-qos-pol:name>
               </bbf-qos-pol:classifiers>
            </bbf-qos-pol:policy>
         </bbf-qos-pol:policies>

         <!-- ONU QoS Policy Profile -->
         <bbf-qos-pol:qos-policy-profiles xmlns:bbf-qos-pol="urn:bbf:yang:bbf-qos-policies">
            <bbf-qos-pol:policy-profile>
               <bbf-qos-pol:name>onu-ingress-qos-profile</bbf-qos-pol:name>
               <bbf-qos-pol:policy-list>
                  <bbf-qos-pol:name>onu-ingress-qos-policy</bbf-qos-pol:name>
               </bbf-qos-pol:policy-list>
            </bbf-qos-pol:policy-profile>
         </bbf-qos-pol:qos-policy-profiles>
      </config>
   </edit-config>
</rpc>
```

T-CONTs

T-CONT configuration maps to an OLT Service Port in the PON Solution (i.e., per-Link/T-CONT/GEM Port configuration). The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-xpongemtcont:xpongemtcont/bbf-xpongemtcont:tconts	
name	There are no restrictions on the T-CONT name.
alloc-id	The Alloc-ID to use for this T-CONT (i.e., OLT Service Port). If the value is not specified, the Netconf Server automatically assigns an Alloc-ID value for this T-CONT, and updates the ONU Inventory in the OLT based on the value specified for this attribute. See T-CONT and GEM Port Mapping for more information on how T-CONT configuration is mapped to hardware resources. **MongoDB Ref:** OLT-CFG.ONUs.<ONU SSN>.OLT-Service 0
traffic-descriptor-profile-ref	Must reference an existing Traffic Descriptor Profile.

The following XML is an example of configuring a T-CONT.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <xpongemtcont xmlns="urn:bbf:yang:bbf-xpongemtcont">
          <tconts>
            <tcont>
              <name>vani-ALPHe30cadcf-tcont.1</name>
              <interface-reference>vani-ALPHe30cadcf</interface-reference>
              <traffic-descriptor-profile-ref>sla-add-ctag</traffic-descriptor-profile-ref>
            </tcont>
          </tconts>
        </xpongemtcont>
      </config>
   </edit-config>
</rpc>
```

Traffic Descriptor Profiles

A Traffic Descriptor Profile is referenced by a T-CONT and defines the *upstream* Service Level Agreement (SLA) for a T-CONT. In the current release, downstream SLAs cannot be configured using BBF YANG. However, downstream SLAs can be managed through Tibit YANG, MCMS PON Manager, or directly in MongoDB. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/bbf-xpongemtcont:xpongemtcont/bbf-xpongemtcont:traffic-descriptor-profiles	
name	There are no restrictions on the profile name.
fixed-bandwidth	Fixed grant rate in units of bits per second. This is an unsolicited grant by the DBA regardless of need. **MongoDB Ref:** SLA-CFG.Up Fixed Rate
assured-bandwidth	Guaranteed (high priority) rate in units of bits per second. **MongoDB Ref:** SLA-CFG.Up Guaranteed Rate
maximum-bandwidth	Total bandwidth allowed for this SLA in units of bits per second, including the Fixed, Assured (or Guaranteed), and Best Effort portions of the SLA. The Best Effort portion of the SLA is the remaining bandwidth after taking into account Fixed and Assured portions. The Best Effort bandwidth is calculated as follows: Best Effort = maximum-bandwitdh - (fixed bandwidth + assured-bandwidth). **MongoDB Ref:** SLA-CFG.Up Best Effort Rate
priority	Priority level for the Assured (or Guaranteed) portion of the SLA. **MongoDB Ref:** SLA-CFG.Up Priority
Additional-bw-eligibility-indicator	Must be set to a value of ‘non-assured-sharing’.

The following XML is an example of configuring a Traffic Descriptor Profile.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running />
      </target>
```

```
      <config>
      <xpongemtcont xmlns="urn:bbf:yang:bbf-xpongemtcont">
        <traffic-descriptor-profiles>
        <traffic-descriptor-profile>
          <name>sla-add-ctag</name>
          <fixed-bandwidth>0</fixed-bandwidth>
          <assured-bandwidth>0</assured-bandwidth>
          <maximum-bandwidth>10000000000</maximum-bandwidth>
          <additional-bw-eligibility-indicator>
            non-assured-sharing
          </additional-bw-eligibility-indicator>
        </traffic-descriptor-profile>
        </traffic-descriptor-profiles>
      </xpongemtcont>
      </config>
   </edit-config>
</rpc>
```

UNI

The User Network Interface (UNI) represents a physical Ethernet interface on the ONU that the customer connects to in order to access service from the operator. The MCMS Netconf Server represents each ONU UNI as a standard Ethernet interface with ietf-interfaces type iana-if-type:ethernetCsmacd. For ONUs that have Virtual Ethernet Interface Points (VEIPs), NETCONF represents each VEIP as a standard Ethernet interface.

In the current MCMS Netconf Server implementation, there are no configurable attributes for a UNI. However, the UNI must be created and connected to VLAN subinterfaces to establish a forwarding path through the ONU. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface	
name	**There are restrictions on the UNI interface name.** The name encodes an index that identifies a specific UNI or VEIP port number on the ONU. See ONU UNI Port Mapping for more information on how UNI configuration is mapped to ONU ports.
type	The type must be set to iana-if-type:ethernetCsmacd for UNI interfaces.

The following XML is an example of configuring a UNI.

```
<rpc
```

```
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
        <interfaces
          xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
          xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">
          <interface>
            <name>onu-ALPHe30cadcf-eth.5</name>
            <type>ianaift:ethernetCsmacd</type>
          </interface>
        </interfaces>
      </config>
   </edit-config>
</rpc>
```

vANI

The Virtual Access Network Interface (vANI) is an OLT management object that represents an ONU device attached to the OLT. A vANI interface must be created for each ONU device. The ONU's PON and management channel attributes are configured using the vANI. The vANI is associated with an ANI through an entry in the Link Table.

The vANI associates BBF YANG configuration to a specific ONU device by serial number. The expected-serial-number attribute specifies the Vendor-Specific Serial Number of the XGS-PON ONU. In the case where the ONU is pre-provisioned (i.e., not discovered by the OLT at the time the ONU is configured), the vANI channel-partition must be specified when creating the vANI to associate the ONU to a specific OLT Port.

The current MCMS Netconf Server implementation does not allow the expected-serial-number field to be modified after creating the vANI. To work-around this limitation, delete the vANI with the old serial number and recreate the vANI with the new serial number. A future MCMS Netconf Server implementation will remove this limitation.

The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/ietf-interfaces:interface/bbf-xponani:v-ani	
name	There are no restrictions on the interface configuration name.

type	The type must be set to bbf-xpon-if-type:v-ani for vANI interfaces.
onu-id	The ONU ID (1..128) to use for this ONU device. If the value is not specified, the PON Controller automatically assigns an ONU ID value for this ONU. NETCONF updates the ONU Inventory in the OLT based on the value specified for this attribute. **MongoDB Ref:** OLT-CFG.ONUs.<ONU SSN>.ALLOC ID (OMCC)
channel-partition	A reference to an existing OLT PON Channel Partition interface which associates the ONU to a specific OLT Port (i.e., OLT MicroPlug). This attribute is optional in TR-385, but the MCMS Netconf Server requires this attribute in the create request when pre-provisioning an ONU.
expected-registration-id	The expected-registration-id value is used by the OLT for an ONU authorization feature only. The expected-registration-id cannot be used to associate BBF configuration to a device. The configuration must include expected-serial-number, regardless of the expected-registration-id configuration. The Registration ID is an operator-defined string programmed into ONU's non-volatile memory. The 'expected-registration-id' is used by the OLT's Registration ID authorization feature, where this value is matched against the Registration ID reported by the ONU. If the values do not match, the ONU is not authorized and not permitted to complete registration with the OLT. Setting the value to 'ANY' or a blank string disables this feature in the OLT. **MongoDB Ref:** ONU-CFG.ONU.Expected Registration ID
expected-serial-number	The Vendor-Specific Serial Number identifying the ONU device the vANI configuration is to be applied to. **MongoDB Ref:** ONU-CFG._id
upstream-fec	The PON Solution does not support configuring FEC for a specific ONU. Note that the BBF YANG models assume that disabling FEC does not apply to XGS-PON and that FEC is always enabled. However, the PON MicroPlug OLT supports disabling FEC per OLT PON Port. See Channel Partition for information on configuring FEC for an OLT PON port.
management-gemport-id	The PON Solution requires the ONU ID and Management GEM Port ID to be set to the same value (1..128).
management-gemport-aes-indicator	The PON Solution currently does not support configuring AES encryption for a specific ONU. Encryption can be configured on a per OLT PON Port basis using Tibit YANG.

The following XML is an example of configuring a vANI interface.

```
<rpc
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
    message-id="34566754">
   <edit-config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <target>
         <running/>
      </target>
      <config>
      <interfaces
        xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces"
        xmlns:bbf-xponift="urn:bbf:yang:bbf-xpon-if-type">
        <interface>
          <name>vani-ALPHe30cadcf</name>
          <type>bbf-xponift:v-ani</type>
          <v-ani xmlns="urn:bbf:yang:bbf-xponvani">
            <expected-serial-number>ALPHe30cadcf</expected-serial-number>
          </v-ani>
        </interface>
      </interfaces>
      </config>
   </edit-config>
</rpc>
```

VLAN Subinterfaces

TR-383 supports configuration of VLAN tag matching rules and modifications through VLAN Subinterfaces. VLAN Subinterfaces are associated with physical or logical interfaces on the OLT or ONU. OLT VLAN Subinterfaces are associated with NNI uplinks or PON-side OLT-vENET interfaces. ONU VLAN Subinterfaces are associated with UNIs or VEIPs.

VLAN Subinterfaces support a flexible matching criteria with the ability to match on all fields of the VLAN Tag, including TPID, VLAN ID, PCP, and DEI bits. The match criteria is specified as a set of rules configured in the VLAN Subinterface, and supports matching on zero, one, or two VLAN tags at most. Although the OLT MicroPlug supports a three-tag match, TR-383 does not support matching on more than two VLAN tags. Three-tag matching can be configured using YANG if such a service is required.

TR-383 supports VLAN Tag modifications in the form of push and pop operations. Pop operations are specified as the number of tags to remove from frames as they are received. Push operations are specified as one or two tags applied as an ingress or egress operation. All VLAN Tag fields including the TPID, VLAN ID, PCP, and DEI are specified as part of the push operation.

OLT VLAN Subinterfaces

OLT VLAN Subinterfaces are associated with either an NNI or PON-side interface and are used in conjunction with Forwarders to establish the datapath for forwarding frames through the OLT. The PON Controller does not support more advanced tag matching and modifications for the OLT. As a result, VLAN Subinterface configuration support is limited for the OLT. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/interface/bbf-sub-interfaces:inline-frame-processing	
name	There are no restrictions on the VLAN subinterface configuration name.
type	The type must be set to bbf-if-type:vlan-sub-interface for VLAN subinterfaces.
egress-qos-policy-profile	Reference to a QoS Policy Profile that defines a mapping for classifying downstream frames to GEM Ports. The profile is used to configure a Downstream QoS Map (DS-MAP-CFG) on the OLT. **MongoDB Ref:** ONU-CFG.OLT-Service.DS-MAP-CFG
ingress-rule/match-criteria	
match-all	Match all untagged, priority-tagged, and VLAN tagged frames. **MongoDB Ref:** ONU-CFG.OLT-Service.NNI Network, ONU-CFG.OLT-Service.PON Network
untagged	Match untagged frames. **MongoDB Ref:** ONU-CFG.OLT-Service.NNI Network, ONU-CFG.OLT-Service.PON Network
vlan-tagged/tag-type	Match the TPID field in the VLAN Tag. Supports string values ‘bbf-dot1q-types:s-vlan’ and ‘bbf-dot1q-types:c-vlan’ only. **MongoDB Ref:** ONU-CFG.OLT-Service.NNI Network, ONU-CFG.OLT-Service.PON Network
vlan-tagged/vlan-id	Match the VLAN ID in the VLAN Tag. Supports integer values 0..4095 representing the VLAN ID or the string ‘priority-tagged’. **MongoDB Ref:** ONU-CFG.OLT-Service.NNI Network, ONU-CFG.OLT-Service.PON Network
vlan-tagged/pbit	Matching on PCP bits is not supported for the OLT.

vlan-tagged/dei	Matching on the DEI bit is not supported for the OLT.
ingress-rule/ingress-rewrite	
pop-tag	Number of VLAN tags to remove from the frame. At most, one tag can be popped for the OLT.
egress-rewrite	
push-tag/tag-type	Set the TPID field in the VLAN Tag. Supports string values 'bbf-dot1q-types:s-vlan' and 'bbf-dot1q-types:c-vlan' only. **MongoDB Ref:** ONU-CFG.OLT-Service.NNI Network, ONU-CFG.OLT-Service.PON Network
push-tag/vlan-id	Set the VLAN ID value 0..4095 in the VLAN Tag. **MongoDB Ref:** ONU-CFG.OLT-Service.NNI Network, ONU-CFG.OLT-Service.PON Network
push-tag/pbit	Must be set to one of the following values for the OLT: ● 'pbit-from-tag = 0' - copy the pbits from the inner tag. ● 'write-pbit = <value>' - set the pbits to the specified value in the tag added by the OLT. No other values are supported. The 'write-pbit' attribute is supported for upstream only. **MongoDB Ref:** ONU-CFG.OLT-Service 0.US CoS Treatment, ONU-CFG.OLT-Service 0.US CoS Value
push-tag/dei	Must be set to 'write-dei-0' for the OLT. No other values are supported.
l2-dhcpv4-relay	
enable	Enable DHCPv4 Relay Option 82 insertion for this service. Note: this configuration applies to PON Network OLT VLAN Subinterfaces only. **MongoDB Ref:** ONU-CFG.OLT-Service.Filter.DHCPv4
profile-ref	Reference to an L2 DHCPv4 Relay Profile that defines the Option 82 format and which suboptions to insert in DHCP messages received from the client. Note: this configuration applies to PON Network OLT VLAN Subinterfaces only.
subscriber-profile	
profile	Reference to a Subscriber Profile that defines the DHCPv4 Relay

	Option 82 Circuit ID and Remote ID values to configure for this service. Note: this configuration applies to PON Network OLT VLAN Subinterfaces only.

ONU VLAN Subinterface

ONU VLAN Subinterfaces are associated with a UNI or VEIP and are configured in conjunction with bbf-qos-classifiers to establish the datapath for forwarding frames through the ONU. The MCMS Netconf Server translates configuration from a VLAN Subinterface into OMCI Extended Vlan Tagging Operation Configuration Data in the ONU's SRV-CFG file. See section ONU Service Configuration Files for more information on how NETCONF manages ONU SRV-CFG files. The MCMS Netconf Server applies the configuration attributes as described in the table below.

Attribute	Description
/ietf-interfaces:interfaces/interface/bbf-sub-interfaces:inline-frame-processing	
name	There are no restrictions on the VLAN subinterface configuration name.
type	The type must be set to bbf-if-type:vlan-sub-interface for VLAN subinterfaces.
ingress-qos-policy-profile	Reference to a QoS Policy Profile that defines a mapping for classifying frames forwarded through the ONU to GEM Ports. The profile is used to configure the IEEE 802.1p Mapper Service Profile on the ONU. **MongoDB Ref:** SRV-CFG.OMCI.Ieee8021pMapperServiceProfile
ingress-rule/match-criteria	
match-all	Match all untagged, priority-tagged, and VLAN tagged frames. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.filter_inner_tpid_de
untagged	Match untagged frames. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.

	received_frame_vlan_tagging_operation_table.filter_inner_tpid_de
vlan-tagged/tag-type	Match the TPID field in the VLAN Tag. Supports string values 'bbf-dot1q-types:s-vlan' and 'bbf-dot1q-types:c-vlan' only. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.filter_inner_tpid_de
vlan-tagged/vlan-id	Match the VLAN ID in the VLAN Tag. Supports integer values 0..4095 representing the VLAN ID, the string 'priority-tagged', or the string 'any'. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData. .received_frame_vlan_tagging_operation_table.filter_inner_vid
vlan-tagged/pbit	Match the value of the PCP bits in the VLAN Tag. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.filter_inner_priority
vlan-tagged/dei	Match the value of the DEI bit in the VLAN Tag. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.filter_inner_tpid_de
ingress-rule/ingress-rewrite	
pop-tag	Number of VLAN tags to remove from the frame. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.treatment_tags_to_remove
push-tag/tag-type	Set the TPID field in the VLAN Tag. Supports string values 'bbf-dot1q-types:s-vlan' and 'bbf-dot1q-types:c-vlan' only. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.treatment_inner_tpid_de
push-tag/vlan-id	Set the value of the VLAN ID 0..4095 in the VLAN Tag. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.treatment_inner_vid

push-tag/pbit	Set the value of PCP bits in the VLAN Tag. Must be set to one of the following values for the ONU: ● 'pbit-from-tag = 0' ● 'write-pbit = <value>' ● 'write-pbit-0 No other values are supported. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.Received_frame_vlan_tagging_operation_table.treatment_inner_priority
push-tag/dei	Set the value of DEI bit in the VLAN Tag. Must be set to one of the following values for the ONU: ● 'write-dei-0 No other values are supported. **MongoDB Ref:** SRV-CFG.OMCI.ExtendedVlanTaggingOperationConfigurationData.received_frame_vlan_tagging_operation_table.treatment_inner_tpid_de
egress-rewrite	Egress rewrite is not supported for the ONU.

Mapping Configuration to Devices

This section describes how configuration through BBF YANG models are mapped to MicroPlug OLT devices and XGS-PON ONUs.

OLT Device Mapping

The tibit-bbf-interfaces YANG model used to map BBF NNI and PON interface configuration to a MicroPlug OLT device. An olt-interface-map entry must be created for each OLT device. The entry maps an OLT MicroPlug device by MAC address to a PON channel and NNI interfaces configured through the BBF YANG models. For example, the following Netconf configuration maps BBF interfaces with Switch Port ID 1/0/1 to the MicroPlug OLT device identified by the MAC address 70:b3:d5:52:37:24.

```
<interfaces xmlns="urn:com:tibitcom:ns:yang:bbf:interfaces">
    <olt-interface-map>
      <olt>
        <device-id>70:b3:d5:52:37:24</device-id>
        <pon>
          <channel-group-ref>channelgroup.1/0/1</channel-group-ref>
          <channel-partition-ref>channelpartion.1/0/1</channel-partition-ref>
```

```
        <channel-pair-ref>channelpair.1/0/1</channel-pair-ref>
        <channel-termination-ref>channeltermination.1/0/1</channel-termination-ref>
      </pon>
      <nni>
        <interface>uplink.1/0/1</interface>
      </nni>
    </olt>
  </olt-interface-map>
</interfaces>
```

ONU Device Mapping

The standard BBF TR-385 YANG attribute bbf-xponvani:expected-serial-number maps BBF ONU configuration to an ONU device by Vendor-Specific Serial Number. For example, the following configuration maps V-ANI with name ‘vani-TBITc84c00df‘ to the ONU device identified by serial number ‘TBITc84c00df’.

```
<interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
  <interface>
    <name>vani-TBITc84c00df</name>
    <v-ani xmlns="urn:bbf:yang:bbf-xponvani">
      <expected-serial-number>TBITc84c00df</expected-serial-number>
    </v-ani>
  </interface>
</interfaces>
```

T-CONT and GEM Port Mapping

The MCMS Netconf Server applies T-CONT and GEM Port configuration to both OLT and ONU devices. There is no explicit mapping from the BBF T-CONT and GEM Port configuration to specific hardware resources on the OLT or ONU. As a result, the Netconf Server automatically maps BBF configuration to T-CONT and XGEM Port resources on the device. The mapping to hardware resources depends on the type of device being programmed, which is described in the sections below.

T-CONT and GEM Port Mapping to OLT Service Ports

OLT T-CONT and GEM Port resources are configured through OLT Service Ports in PON Controller ONU-CFG files. The Netconf Server automatically maps the BBF configuration to OLT Service Ports according to the bbf-xpongemtcont configuration. The Netconf Server sets the OLT Service Port name (ONU-CFG.OLT-Service.Name) to the name specified by /xpongemtcont/gemports/gemport/name for the purposes of debugging and viewing the configuration through PON Manager.

OLT Service Ports operate in one of two modes according to the PON Controller provisioning model. When the OLT Service Port is operating in “T-CONT and XGEM” mode, the PON Controller programs both a T-CONT and XGEM Port on the OLT MicroPlug. In this mode, OLT uses the same identifier value for both the Alloc-ID and GEM Port ID for this service.

The OLT Service Port operates in “XGEM-only” mode, when the configuration references another OLT Service Port that is operating in “T-CONT and XGEM” mode. The T-CONT reference is configured by ONU-CFG.OLT-Service.Service Reference parameter in the database. In “XGEM Port” mode, the PON Controller programs an XGEM Port with a reference to the T-CONT in the OLT MicroPlug for the OLT Service Port.

The following sections describe how the Netconf Server maps BBF configuration to OLT Service Ports for single and multiple GEM Port configurations.

Single GEM Port Configurations

Single GEM Port configurations have a one-to-one mapping between T-CONT and GEM Port, where each T-CONT is referenced by a single GEM Port. For these configurations, there is also a one-to-one mapping between T-CONT/GEM Port and OLT Service Port. An example mapping is shown in the table below.

BBF T-CONT/GEM Port	ONU-CFG
vani-ALPHe30cadcf-tcont.1 gem-ALPHe30cadcf-eth.5-tc0	ONU-CFG.OLT-Service 0 (T-CONT and XGEM) Enable = true Name = “gem-ALPHe30cadcf-eth.5-tc0” Service Reference = “”
vani-ALPHe30cadcf-tcont.2 gem-ALPHe30cadcf-eth.5-tc4	ONU-CFG.OLT-Service 1 (T-CONT and XGEM) Enable = true Name = “gem-ALPHe30cadcf-eth.5-tc4” Service Reference = “”
vani-ALPHe30cadcf-tcont.3 gem-ALPHe30cadcf-eth.5-tc5	ONU-CFG.OLT-Service 2 (T-CONT and XGEM) Enable = true Name = “gem-ALPHe30cadcf-eth.5-tc5” Service Reference = “”

Multi-GEM Port Configurations

Multi-GEM Port configurations have multiple GEM Ports referencing a single T-CONT. For these configurations, there is a one-to-many relationship between T-CONT and GEM Ports and a one-to-one mapping between GEM Port and OLT Service Port. The Netconf Server automatically sets the OLT Service Port.Service Reference in the ONU-CFG based on the bbf-xpongemtcont configuration. An example mapping is shown in the table below.

BBF T-CONT/GEM Port	ONU-CFG
vani-ALPHe30cadcf-tcont.1 gem-ALPHe30cadcf-eth.5-tc0	ONU-CFG.OLT-Service 0 (T-CONT and XGEM) Enable = true Name = “gem-ALPHe30cadcf-eth.5-tc0” Service Reference = “” DS-MAP-CFG = “olt-egress-qos-profile”
gem-ALPHe30cadcf-eth.5-tc1 *(ref: vani-ALPHe30cadcf-tcont.1)*	ONU-CFG.OLT-Service 1 (XGEM-only) Enable = true Name = “gem-ALPHe30cadcf-eth.5-tc1” Service Reference = “OLT-Service 0”
gem-ALPHe30cadcf-eth.5-tc2 *(ref: vani-ALPHe30cadcf-tcont.1)*	ONU-CFG.OLT-Service 2 (XGEM-only) Enable = true Name = “gem-ALPHe30cadcf-eth.5-tc2” Service Reference = “OLT-Service 0”
gem-ALPHe30cadcf-eth.5-tc3 *(ref: vani-ALPHe30cadcf-tcont.1)*	ONU-CFG.OLT-Service 3 (XGEM-only) Enable = true Name = “gem-ALPHe30cadcf-eth.5-tc3” Service Reference = “OLT-Service 0”

Downstream Classification by Priority

The PON Controller uses a Downstream QoS Map (DS-MAP-CFG) to program the OLT to classify downstream frames by priority bits in the VLAN Tag to GEM Ports. The Downstream QoS Map in MongoDB provides a mapping between priority bits in the VLAN tag to OLT Service Ports.

The Downstream QoS Map is defined as a list of VLAN Priority Bit (CoS) values 0..7 to OLT-Service Offsets 0..7, where the OLT Services represent a T-CONT and a group of associated GEM Ports. The OLT-Service Offset is the relative location of the OLT Service Port in the group of OLT Service ports representing the T-CONT and GEM Ports. This allows downstream traffic being sent to the ONU to be carried on different GEM Ports based on the CoS values. The Downstream QoS Map is referenced by the ONU Service configuration on the OLT Service Port operating in 'T-CONT and XGEM' mode. See the example showing the DS-MAP-CFG attribute in the table above.

The Netconf Server converts the BBF YANG T-CONT, GEM Port, VLAN Subinterface and QoS Policy Profile configuration into the Downstream QoS Map configuration DS-MAP-CFG and ONU-CFG defined by the PON Controller data model for MongoDB. An example mapping from a BBF QoS Policy Profile configuration to DS-MAP-CFG is shown in the table below, which classifies 8 levels of priority to four traffic classes where each traffic class represents a GEM Port (OLT-Service Port).

BBF Qos Policy Profile	DS-MAP-CFG
In-pbit-list 0,1 => traffic-class 0	COS 0 => OLT Service Offset 0 COS 1 => OLT Service Offset 0
In-pbit-list 2,3 => traffic-class 1	COS 2 => OLT Service Offset 1 COS 3 => OLT Service Offset 1
In-pbit-list 4,5 => traffic-class 2	COS 4 => OLT Service Offset 2 COS 5 => OLT Service Offset 2
In-pbit-list 6,7 => traffic-class 3	COS 6 => OLT Service Offset 3 COS 7 => OLT Service Offset 3

Figure 5 shows how the Netconf Server converts the BBF QoS Policy Profile configuration into a PON Controller DS-MAP-CFG. The classifier definitions in the QoS Policy Profile along with the VLAN Subinterface 'egress-qos-policy-profile' attribute and 'traffic-class' attribute in each GEM Port are used by the Netconf Server to create the DS-MAP-CFG configuration and related CoS to OLT-Service Offset mappings. The QoS Policy Profile name is used as the DS-MAP-CFG document identifier (_id) in MongoDB.

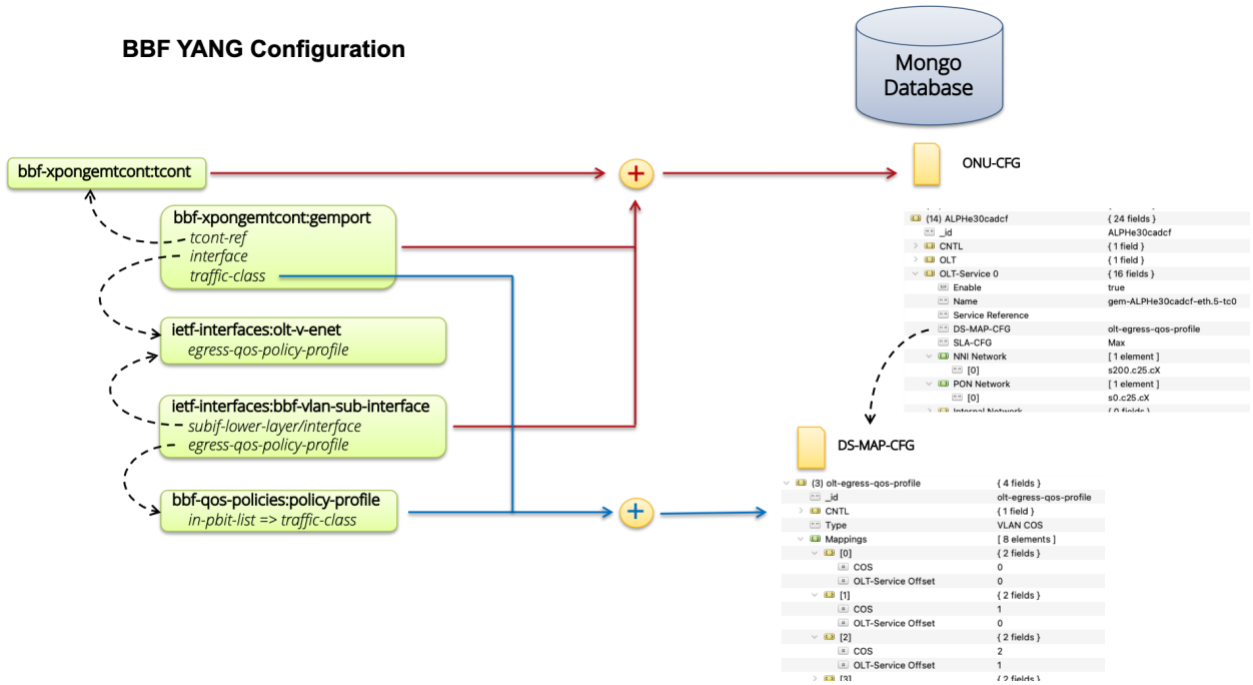


Figure 5 - Downstream QoS Map Configuration (DS-MAP-CFG)

T-CONT and GEM Port Mapping to ONU Configuration

ONU T-CONT and GEM Port resources are configured through OMCI using PON Controller SRV-CFG files. See ONU Service Configuration Files for information on how the Netconf Server manages SRV-CFG files for individual ONUs. More specifically, Netconf Server configures the Alloc-ID for the T-CONT in the ONU SRV-CFG file (e.g., SRV-CFG.OMCI.Tcont.32768.alloc_id). Likewise, the Netconf Server configures the Port ID for the GEM Port in the ONU SRV-CFG file (e.g., SRV-CFG.OMCI.GemPortNetworkCtp.256.port_id).

ONU T-CONTs and GEM Ports are referenced by OMCI Managed Entity Identifiers (ME IDs), which are defined as two byte integer values numbered in ascending order. T-CONT ME IDs. There is no direct mapping from the BBF T-CONT and GEM Port name to an ME ID value. Instead, the Netconf Server automatically maps the BBF configuration to hardware resources on the ONU. The Netconf Server discovers the T-CONT ME IDs from the ONU and applies the BBF configuration to the ONU based on the ME IDs discovered from the ONU. An example mapping is shown in the table below.

BBF T-CONT Name	SRV-CFG T-CONT ME ID
vani-ALPHe30cadcf-tcont.1	32768

vani-ALPHe30cadcf-tcont.2	32769
vani-ALPHe30cadcf-tcont.3	32770

GEM Port ME IDs are selected by the Netconf Server based on the number of GEMs from the BBF configuration. An example mapping from BBF GEM Port name to ME ID is shown in the table below.

BBF GEM Port Name	SRV-CFG GemPortNetworkCtp ME ID
gem-ALPHe30cadcf-eth.5-tc0	256
gem-ALPHe30cadcf-eth.5-tc4	257
gem-ALPHe30cadcf-eth.5-tc5	258

ONU UNI Port Mapping

The MCMS Netconf Server applies UNI configuration to UNI and VEIP interfaces on the ONU. However, there is no direct mapping from the UNI name to a specific port on the ONU. As a result, NETCONF relies on the UNI name attribute for this mapping, where a portion of the name is used to identify a specific port number. An index value is encoded in the UNI name according to the following format: <any string><integer>, where one is subtracted from the integer portion of the name to create an index that identifies a specific port on the ONU. If the UNI name does not end with an integer value, the T-CONT configuration maps to index 0. The encoding rules and mapping are the same for T-CONTs. See T-CONT and GEM Port Mapping for examples of how names map to index values.

ONU UNI ports are referenced by OMCI Managed Entity Identifiers (ME IDs), which are defined as two byte integer values numbered in ascending order. UNI ME IDs don't necessarily start at 0. For example, UNI ME IDs start with a value of 257 and VEIP ME IDs start with a value of 1025 in some ONUs. Therefore, there is no direct mapping from the BBF UNI name/index to an ME ID value. Instead, NETCONF learns the UNI ME IDs from the ONU and compiles an ordered list of ME IDs. NETCONF applies the BBF configuration using the UNI index (parsed from the name) to select the ME ID from the list. An example mapping is shown in the table below.

BBF UNI/VEIP Name	Index	SRV-CFG UNI ME ID

onu-ALPHe30cadcf-eth.1	0	257
onu-ALPHe30cadcf-eth.2	1	258
onu-ALPHe30cadcf-eth.3	2	259
onu-ALPHe30cadcf-eth.4	3	260
onu-ALPHe30cadcf-eth.5	4	261

OLT Networking and Datapath

This section details how BBF YANG is used to configure the datapath on the OLT. The datapath and networking on the OLT are configured with a combination of VLAN Subinterfaces and L2 Forwarders. VLAN Subinterfaces define the VLAN tag matching rules (e.g., any, untagged, S, C, S+C, C+C) along with push and pop operations to modify the tags in the matched frames.

VLAN Subinterfaces are attached to either an NNI interface or an OLT-vENET (i.e., PON-side) interface and are connected together through L2 Forwarders to establish the datapath between the NNI and an OLT Service Port on the PON (Link/GEM Port/T-CONT). A Forwarder represents an L2 Switching Domain (L2SD) on a OLT MicroPlug, and is used to configure L2 bridging functions such as L2 switching and forwarding, MAC learning, and flooding.

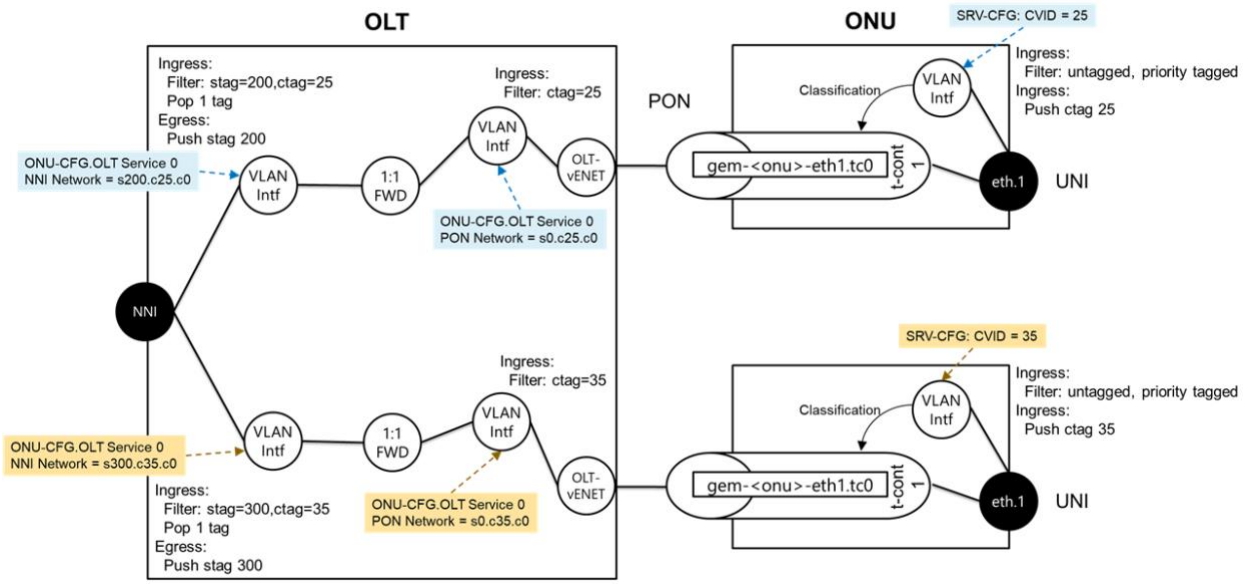


Figure 6 - Point-to-point (1:1) Service Configuration

A point-to-point (1:1) service configuration has exactly one NNI VLAN Subinterface, exactly one PON VLAN Subinterface, and exactly one L2 Forwarder. Figure 6 shows a configuration with two ONUs with a point-to-point service configured for each ONU. The top ONU is configured for a point-to-point service, where the ONU adds an inner CTAG with VID 23 and the OLT adds an outer STAG with VID 200. A second point-to-point service is configured for the bottom ONU, where the ONU adds an inner CTAG with VID 35 and the OLT adds an outer STAG with VID 300.

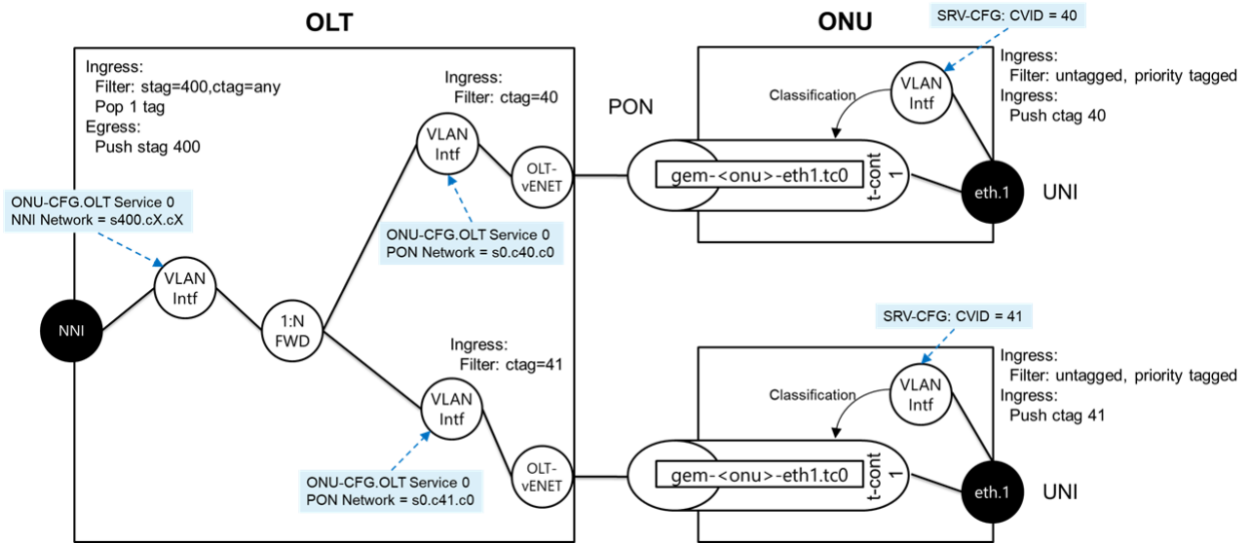


Figure 7 - Multi-point (1:N) Service Configuration

A multipoint (1:N) service configuration has exactly one NNI VLAN Subinterface, one or more PON VLAN Subinterfaces, and exactly one L2 Forwarder. Figure 7 shows a configuration with a single multipoint (or shared) VLAN service connected to two ONUs. The top ONU is configured to add an inner CTAG with VID 40, and the bottom ONU is configured to add an inner CTAG with VID 41. The OLT is configured with two PON VLAN Subinterfaces - one to match CVID 40 for the top ONU and a second to match CVID 41 for the bottom ONU. There is a single NNI VLAN Subinterface configured to add an outer STAG with VID 400. In multipoint configurations, the L2 Forwarder represents an L2SD that requires MAC learning and flooding for forwarding downstream traffic to the ONUs.

NETCONF translates the VLAN Subinterface configuration into OLT Service Port NNI and PON Network entries in the ONU OLT Service Port configuration. Figure 8 shows an example of the OLT Service Port configuration for the top and bottom ONUs that corresponds to the point-to-point service above. The top ONU has an OLT Service Port configured with NNI Network “s200.c25.c0” which programs the OLT to match an outer STAG with VID 200 and an inner CTAG with VID 25. The top ONU’s PON Network is configured with “s0.c25.c0” which programs the OLT to match a CTAG with CVID 25. Because the NNI Network value does not equal the PON Network value, the OLT is also programmed to modify the tags. In this case, the OLT is programmed to add an outer STAG with VID 200 to upstream frames and pop one tag from downstream frames. In the current implementation, NETCONF does not add networks to the OLT’s NNI Inventory (OLT-CFG.NNI Networks).

Top ONU-CFG

```
OLT-Service 0 = {
```

Bottom ONU-CFG

```
OLT-Service 0 = {
```

```
    "Enable" : true,
    "Name" : "vani-TopONU-tcont.1",
    "NNI Network" : [
        "s200.c25.c0"
    ],
    "PON Network" : [
        "s0.c25.c0"
    ],
    "DHCP" : {
        "Remote ID" : "",
        "Circuit ID" : "",
        "Sub Options" : ""
    },
    "RADIUS" : {
        "NAS Identifier" : "",
        "NAS Port ID" : "",
        "User Name Override" : ""
    },
    "Filter" : {
        "DHCPv4" : "pass",
        "DHCPv6" : "pass",
        "EAPOL" : "pass"
    },
    "SLA-CFG" : "Max"
}
```

```
    "Enable" : true,
    "Name": "vani-BottomONU-tcont.1",
    "NNI Network" : [
        "s300.c35.c0"
    ],
    "PON Network" : [
        "s0.c35.c0"
    ],
    "DHCP" : {
        "Remote ID" : "",
        "Circuit ID" : "",
        "Sub Options" : ""
    },
    "RADIUS" : {
        "NAS Identifier" : "",
        "NAS Port ID" : "",
        "User Name Override" : ""
    },
    "Filter" : {
        "DHCPv4" : "pass",
        "DHCPv6" : "pass",
        "EAPOL" : "pass"
    },
    "SLA-CFG" : "Max"
}
```

Figure 8 - Service Configuration Applied to ONU-CFG in MongoDB

ONU Service Configuration Files

The PON Controller database defines a MongoDB collection for ONU Service Configuration files (SRV-CFG). Each SRV-CFG file contains a list of configuration attributes that is used by the PON Controller to configure subscriber services on the ONU. For GPON ONUs, the PON Controller translates the contents of the SRV-CFG file into a sequence of OMCI requests to program the ONU. For EPON ONUs, the PON Controller translates the contents of the SRV-CFG files into a sequence of DPoE OAM requests to program the ONU. Typically, these SRV-CFG files are managed using the MCMS PON Manager's OMCI Editor tool or created externally and imported directly into MongoDB.

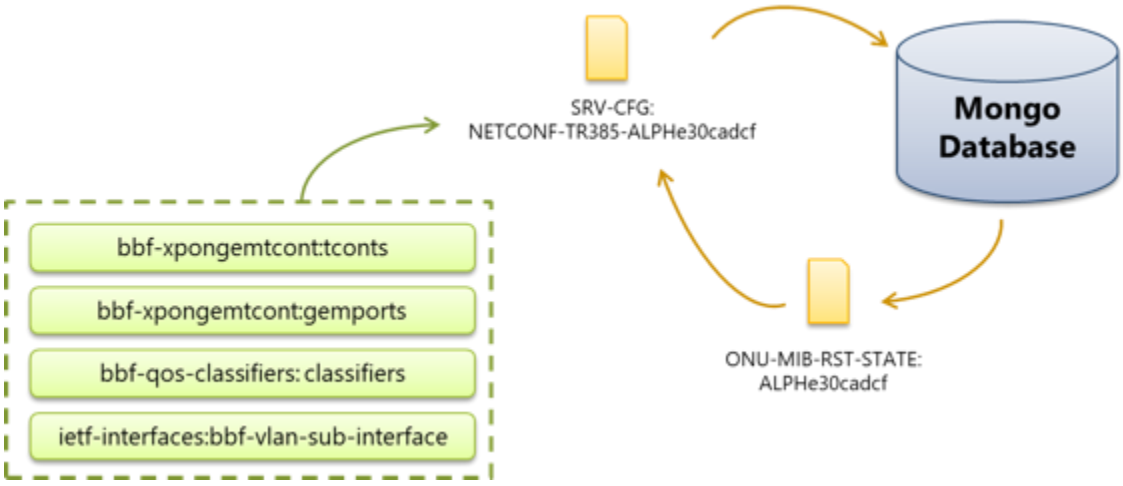


Figure 9 - ONU Service Configuration File Management

The MCMS Netconf Server automatically generates a unique SRV-CFG file for each ONU configured through BBF YANG models. The process by which NETCONF generates the SRV-CFG file is shown in Figures 9 and 10. NETCONF gathers a list of available ONU resources, such as T-CONTs, UNIs, VEIPs, and queues, and builds the SRV-CFG file from the BBF YANG configuration using the resources available on the ONU. ONU resources are reported in the ONU-MIB-RST-STATE in MongoDB.

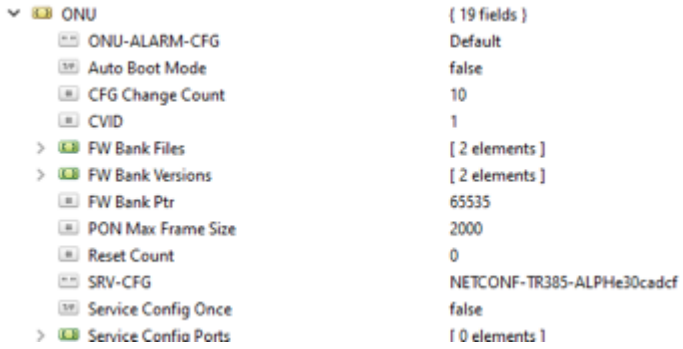


Figure 10 - ONU-CFG Reference to SRV-CFG File

In the case where the ONU is pre-provisioned (i.e., not discovered by the OLT at the time the ONU is configured), NETCONF generates the SRV-CFG file for the ONU after the ONU registers with the PON Controller.

Note that BBF YANG only supports configuration for GPON ONUs. EPON DPoE ONUs can be configured through YANG or the PON Manager.

Examples

This section describes the BBF examples provided with the MCMS Netconf Server. The examples are found under the NETCONF installation directory '/opt/tibit/netconf/examples/bbf'.

Create OLT

The Create OLT example configures the NNI and the four logical PON channel interfaces for a MicroPlug OLT device. This example also configures an olt-interface-mapping entry that associates the NNI and PON channel configuration to a specific OLT device.

Example directory: /opt/tibit/netconf/examples/bbf/create_olt

The example breaks the configuration into the following NETCONF <edit-config> requests:

1. **olt-interfaces.xml** - Create OLT Interfaces, including Uplink interface for NNI and PON Channel Group, Channel Partition, Channel Pair, and Channel Termination interfaces.
2. **tibit-olt-map-table.xml** - Create a Tibit olt-interface-map entry that associates the configuration from Step 1 to an specific Tibit MicroPlug™ OLT device by MAC address.

Running the example:

The following configures the OLT with MAC address 70:b3:d5:52:37:24 plugged into the switch on port "1/0/1".

```
$ cd /opt/tibit/netconf/examples/bbf/create_olt
$ ./create_olt.py --olt 70:b3:d5:52:37:24 --olt_port 1/0/1
```

The following deletes the configuration for the specified OLT and switch port:

```
$ cd /opt/tibit/netconf/examples/bbf/create_olt
$ ./delete_olt.py --olt 70:b3:d5:52:37:24 --olt_port 1/0/1
```

Full help text for the example script is shown below:

```
$ ./create_olt.py --help
usage: create_olt.py [--help] [-h HOST] [--olt OLT] --olt_port OLT_PORT
                     [-w PASSWD] [-p PORT] [-u USER] [-v]
```

```
optional arguments:
  --help                Show this help message and exit.
  -h HOST, --host HOST  NETCONF Server IP address or hostname. (default:
                        127.0.0.1)
  --olt OLT             OLT device MAC address (default: 0)
  --olt_port OLT_PORT   OLT Port number. This could be a logical port number
                        or a physical port number representing the switch port
                        (e.g., LLDP switch port ID) (default: None)
  -w PASSWD, --passwd PASSWD
                        Password. If no password is provided, attempt to read
                        it from .nc_edit_auth. (default: None)
  -p PORT, --port PORT  NETCONF Server port number. (default: 830)
  -u USER, --user USER  Username. (default: None)
  -v, --verbose         Verbose output. (default: False)
```

Add CTag Service

Prerequisites: Configure OLT on switch port 1/0/1. See the Create OLT example.

The Add CTag Service example configures the ONU to add an inner C-VLAN Tag with TPID 8100 and the OLT to add an outer S-VLAN Tag with TPID 88A8, and is shown in Figure 11. Tags are added by the devices in the upstream direction and removed in the downstream direction.

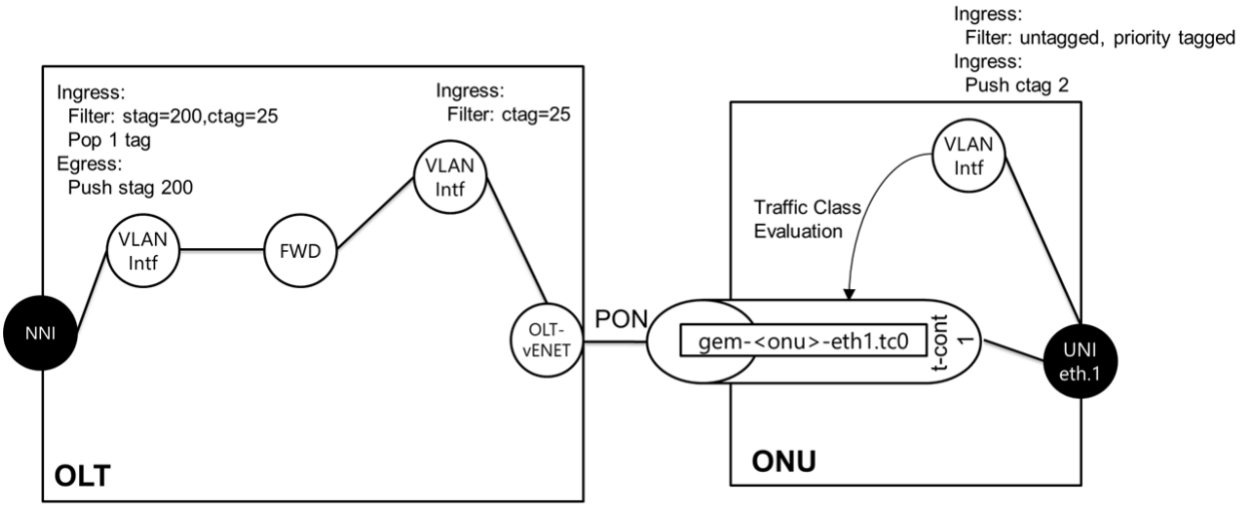


Figure 11 - Add CTag Service Example

Example directory: /opt/tibit/netconf/examples/bbf/add_ctag_service

The example breaks the configuration into the following NETCONF <edit-config> requests:

3. **onu-interfaces.xml** - Create ONU related interfaces, including the ANI, vANI, UNI, and OLT-vENET.
4. **link-table.xml** - Create entries in the Link Table to associate the vANI to an ANI and OLT-vENET to a UNI.
5. **onu-vlan-sub-interfaces.xml** - Create a VLAN Subinterface on the ONU that adds an inner CTAG to untagged and priority tagged frames ingressing the UNI.
6. **gemports.xml** - Creates a Traffic Descriptor Profile (upstream SLA), T-CONT, and GEM Port for the Add CTag service.
7. **olt-1to1-forwarding.xml** - Creates VLAN Subinterfaces and an L2 Forwarder that configures the OLT to add an outer STAG to frames received from the ONU.

Running the example:

The following configures the ONU with serial number ALPHe30cadcf to add an inner CTAG with VID 25 on UNI port 1, and configures the OLT on switch port 1/0/1 to add an outer STAG with VID 200 to frames received from the ONU.

```
$ cd /opt/tibit/netconf/examples/bbf/add_ctag_service
$ ./config_add_ctag_svc.py --olt_port 1/0/1 --onu ALPHe30cadcf --uni 1
--olt_tag 200 --onu_tag 25
```

The following deletes the Add CTag configuration for the ONU with the specified serial number and from OLT configuration for the specified switch port.

```
$ cd /opt/tibit/netconf/examples/bbf/add_ctag_service
$ ./disable_svc.py --onu ALPHe30cadcf --uni 1 --olt_tag 200 --onu_tag
25
```

Full help text for the example script is shown below::

```
$ ./add_ctag_service/config_add_ctag_svc.py --help
usage: config_add_ctag_svc.py [--help] [--best_effort BEST_EFFORT]
                              [--guaranteed GUARANTEED] [-h HOST] --olt_port
                              OLT_PORT [--olt_tag OLT_TAG] --onu ONU --onu_tag
                              ONU_TAG [-w PASSWD] [-p PORT] [--uni UNI_PORT]
                              [-u USER] [-v]

optional arguments:
  --help                Show this help message and exit.
  --best_effort BEST_EFFORT
                        Best effort bandwidth in bps (default: 10000000000)
  --guaranteed GUARANTEED
                        Guaranteed (or assured) bandwidth in bps (default: 0)
  -h HOST, --host HOST  NETCONF Server IP address or hostname. (default:
                        127.0.0.1)
  --olt_port OLT_PORT   OLT Port number. This could be a logical port number
                        or a physical port number representing the switch port
                        (e.g., LLDP switch port ID) (default: None)
  --olt_tag OLT_TAG     Tag to be added by the OLT (default: 0)
  --onu ONU             ONU Serial Number (e.g., TBITc84c00df) (default: None)
  --onu_tag ONU_TAG     Tag to be added by the ONU (default: None)
  -w PASSWD, --passwd PASSWD
                        Password. If no password is provided, attempt to read
                        it from .nc_edit_auth. (default: None)
  -p PORT, --port PORT  NETCONF Server port number. (default: 830)
  --uni UNI_PORT        UNI port number 1..5 (default: 1)
  -u USER, --user USER  Username. (default: None)
```

```
  -v, --verbose         Verbose output. (default: False)
```

Unmodified Service

Prerequisites: Configure OLT on switch port 1/0/1. See the Create OLT example.

The Unmodified Service example configures the ONU to forward untagged and single-tagged frames transparently through the ONU without any modifications to the frame, and is shown in Figure 12. The OLT is configured to add an outer S-VLAN Tag with TPID 88A8. The S-VLAN Tag is added by the OLT in the upstream direction and removed by the OLT in the downstream direction.

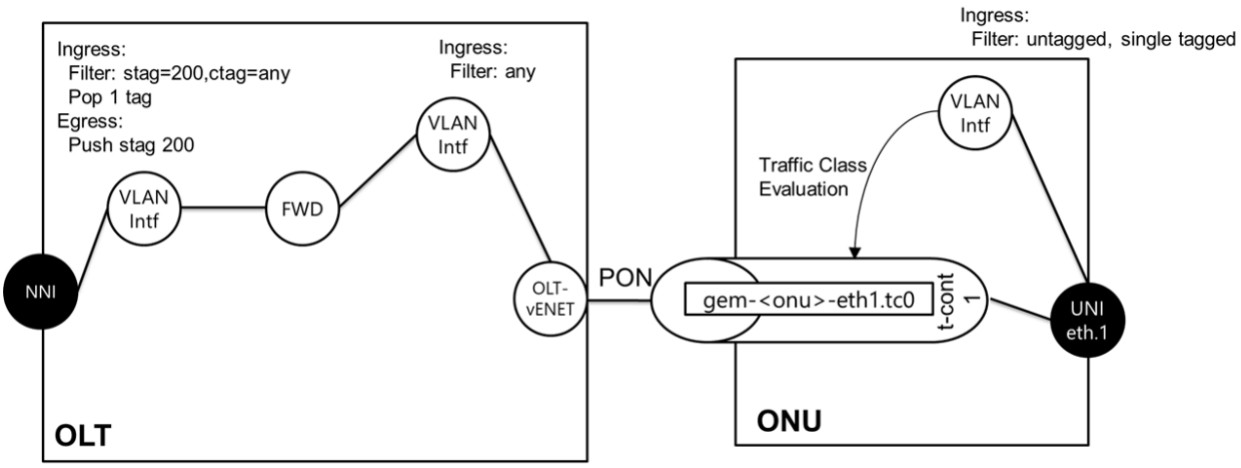


Figure 12 - Unmodified Service Example

Example directory: /opt/tibit/netconf/examples/bbf/unmodified_service

The example breaks the configuration into the following NETCONF <edit-config> requests:

1. **onu-interfaces.xml** - Create ONU related interfaces, including the ANI, vANI, UNI, and OLT-vENET.
2. **link-table.xml** - Create entries in the Link Table to associate the vANI to an ANI and OLT-vENET to a UNI.
3. **onu-vlan-sub-interfaces.xml** - Create a VLAN Subinterface on the ONU that forwards untagged and single tagged frames without modification to the tags in the frames.
4. **gemports.xml** - Create a Traffic Descriptor Profile (upstream SLA), T-CONT, and GEM Port for the Add CTag service.
5. **olt-1to1-forwarding.xml** - Create VLAN Subinterfaces and an L2 Forwarder that configures the OLT to add an outer STAG to frames received from the ONU.

Running the example:

The following configures the ONU with serial number ALPHe30cadcf for ‘unmodified’ service on UNI port 1, and configures the OLT on switch port 1/0/1 to add an outer STAG with VID 200 to frames received from the ONU.

```
$ cd /opt/tibit/netconf/examples/bbf/unmodified_service
$ ./config_unmodified_svc.py --olt_port 1/0/1 --onu ALPHe30cadcf --uni
1 --olt_tag 200
```

The following deletes the Unmodified configuration for the ONU with the specified serial number and from OLT configuration for the specified switch port.

```
$ cd /opt/tibit/netconf/examples/bbf/unmodified_service
$ ./disable_svc.py --onu ALPHe30cadcf --uni 1 --olt_tag 200
```

Full help text for the example script is shown below::

```
$ ./unmodified_service/config_unmodified_svc.py --help
usage: config_unmodified_svc.py [--help] [--best_effort BEST_EFFORT]
                                [--guaranteed GUARANTEED] [-h HOST] --olt_port
                                OLT_PORT [--olt_tag OLT_TAG] --onu ONU
                                [-w PASSWD] [-p PORT] [--uni UNI_PORT]
                                [-u USER] [-v]

optional arguments:
  --help                Show this help message and exit.
  --best_effort BEST_EFFORT
                        Best effort bandwidth in bps (default: 10000000000)
  --guaranteed GUARANTEED
                        Guaranteed (or assured) bandwidth in bps (default: 0)
  -h HOST, --host HOST  NETCONF Server IP address or hostname. (default:
                        127.0.0.1)
  --olt_port OLT_PORT   OLT Port number. This could be a logical port number
                        or a physical port number representing the switch port
                        (e.g., LLDP switch port ID) (default: None)
  --olt_tag OLT_TAG     Tag to be added by the OLT (default: 0)
  --onu ONU             ONU Serial Number (e.g., TBITc84c00df) (default: None)
  -w PASSWD, --passwd PASSWD
                        Password. If no password is provided, attempt to read
                        it from .nc_edit_auth. (default: None)
  -p PORT, --port PORT  NETCONF Server port number. (default: 830)
  --uni UNI_PORT        UNI port number 1..5 (default: 1)
  -u USER, --user USER  Username. (default: None)
  -v, --verbose         Verbose output. (default: False)
```

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089

Phone: 888.JUNIPER (888.586.4737)
Web https://support.juniper.net/support/requesting-support

Shared VLAN Service (Multipoint 1:N)

Prerequisites: Configure OLT on switch port 1/0/1. See the Create OLT example.

The Shared VLAN Service example configures the OLT with a point-to-multipoint Shared VLAN with MAC learning and flooding to support forwarding to multiple ONUs, and is shown in Figure 13. The OLT is configured to add an outer S-VLAN Tag with TPID 88A8. The S-VLAN Tag is added by the OLT in the upstream direction and removed by the OLT in the downstream direction. ONUs are configured for an 'Unmodified' service which forwards untagged and single-tagged frames transparently through the ONU without any modifications to the frame.

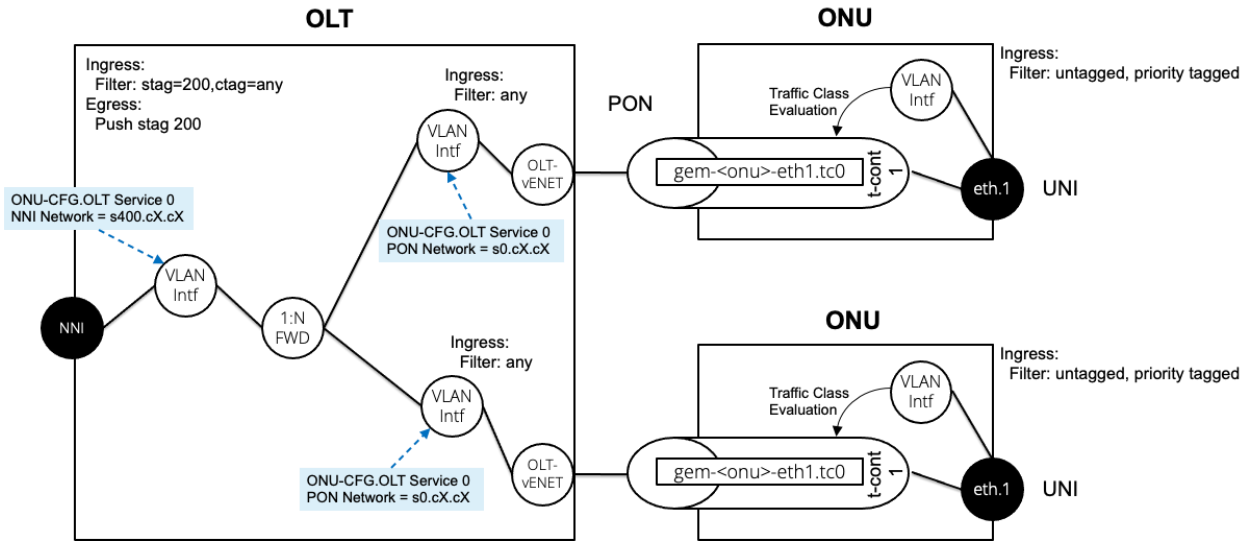


Figure 13 - Shared VLAN Service Example

Example directory: /opt/tibit/netconf/examples/bbf/shared_vlan_service

The example breaks the configuration into the following NETCONF <edit-config> requests:

1. **olt-nni-fowarding.xml** - Create the Forwarder, Forwarding Database, MAC Address Control Profile, and OLT NNI VLAN Subinterface for the Shared VLAN.
2. **onu-interfaces.xml** - Create ONU related interfaces, including the ANI, vANI, UNI, and OLT-vENET.
3. **link-table.xml** - Create entries in the Link Table to associate the vANI to an ANI and OLT-vENET to a UNI.
4. **onu-vlan-sub-interfaces.xml** - Create a VLAN Subinterface on the ONU that forwards untagged and single tagged frames without modification to the tags in the frames.

5. **gemports.xml** - Create a Traffic Descriptor Profile (upstream SLA), T-CONT, and GEM Port for the Add CTag service.
6. **olt-pon-forwarding.xml** - Create PON-side VLAN Subinterfaces that configures the OLT to add an outer STAG to frames received from the ONU.

Running the example:

The following configures a Shared VLAN with VID 200 on the OLT identified by switch port 1/0/1, where the OLT adds an outer STAG with VID 200 to frames received from the ONUs. It also configures ONUs with serial numbers ALPHe30cadcf and BFWS00123193 for 'unmodified' service on UNI port 1, and attaches them to the Shared VLAN configured on the OLT.

NOTE: The create_olt_shared_vlan.py script to create the Shared VLAN is run once for the OLT. The config_onu_shared_vlan_svc.py script is run for each ONU configured on this shared VLAN.

```
$ cd /opt/tibit/netconf/examples/bbf/shared_vlan_service
$ ./create_olt_shared_vlan.py --olt_port 1/0/1 --olt_tag 200
$ config_onu_shared_vlan_svc.py --olt_port 1/0/1 --onu ALPHe30cadcf --
uni 5 --olt_tag 200
$ config_onu_shared_vlan_svc.py --olt_port 1/0/1 --onu BFWS00123193 --
uni 5 --olt_tag 200
```

The following deletes the Unmodified configuration for each ONU and the Shared VLAN from the OLT configuration with the specified switch port.

```
$ cd /opt/tibit/netconf/examples/bbf/unmodified_service
$ ./disable_svc.py --olt_port 1/0/1 --onu ALPHe30cadcf --uni 5 --
olt_tag 200$
$ ./disable_svc.py --olt_port 1/0/1 --onu BFWS00123193 --uni 5 --
olt_tag 200
$ ./delete_olt_shared_vlan.py --olt_port 1/0/1 --olt_tag 200
```

Full help text for the create_olt_shared_vlan.py example script is shown below:

```
$ ./shared_vlan_service/create_olt_shared_vlan.py --help
usage: create_olt_shared_vlan.py [--help] [-h HOST] --olt_port OLT_PORT
                                 [--olt_tag OLT_TAG] [-w PASSWD] [-p PORT]
                                 [-u USER] [-v]

Create a Shared VLAN on an OLT. Example: ./create_olt_shared_vlan.py
```

```
--olt_port 1/0/1 --olt_tag 200

optional arguments:
  --help                Show this help message and exit.
  -h HOST, --host HOST  NETCONF Server IP address or hostname. (default:
                        127.0.0.1)
  --olt_port OLT_PORT   OLT Port number. This could be a logical port number
                        or a physical port number representing the switch port
                        (e.g., LLDP switch port ID) (default: None)
  --olt_tag OLT_TAG     Tag to be added by the OLT (default: 0)
  -w PASSWD, --passwd PASSWD
                        Password. If no password is provided, attempt to read
                        it from .nc_edit_auth. (default: None)
  -p PORT, --port PORT  NETCONF Server port number. (default: 830)
  -u USER, --user USER  Username. (default: None)
  -v, --verbose         Verbose output. (default: False)
```

Full help text for the config_onu_shared_vlan_svc.py example script is shown below:

```
$ ./shared_vlan_service/config_onu_shared_vlan_svc.py --help
usage: config_onu_shared_vlan_svc.py [--help] [-h HOST] --olt_port OLT_PORT
                                     [--olt_tag OLT_TAG] --onu ONU [-w PASSWD]
                                     [-p PORT] [-t TRAFFIC_DESCRIPTOR_PROFILE]
                                     [--uni UNI_PORT] [-u USER] [-v]

Configure a Shared VLAN service for an ONU. Example:
./config_onu_shared_vlan_svc.py --olt_port 1/0/1 --onu TBITc84c0083 --uni 1
--olt_tag 200

optional arguments:
  --help                Show this help message and exit.
  -h HOST, --host HOST  NETCONF Server IP address or hostname. (default:
                        127.0.0.1)
  --olt_port OLT_PORT   OLT Port number. This could be a logical port number
                        or a physical port number representing the switch port
                        (e.g., LLDP switch port ID) (default: None)
  --olt_tag OLT_TAG     Tag to be added by the OLT (default: 0)
  --onu ONU             ONU Serial Number (e.g., TBITc84c00df) (default: None)
  -w PASSWD, --passwd PASSWD
                        Password. If no password is provided, attempt to read
                        it from .nc_edit_auth. (default: None)
  -p PORT, --port PORT  NETCONF Server port number. (default: 830)
  -t TRAFFIC_DESCRIPTOR_PROFILE, --traffic_descriptor_profile
TRAFFIC_DESCRIPTOR_PROFILE
                        Configure the traffic descriptor profile name (e.g.,
                        SLA) for this service. (default: Max)
  --uni UNI_PORT        UNI port number 1..5 (default: 1)
  -u USER, --user USER  Username. (default: None)
```

```
  -v, --verbose         Verbose output. (default: False)
```

Multi-XGem Service

Prerequisites: Configure OLT on switch port 1/0/1. See the Create OLT example.

The Multi-XGem Service example, shown in Figure 14, configures the ONU for an Add CTag service, where multiple services separated by GEM Ports share the . The ONU adds an inner C-VLAN Tag with TPID 8100 and the OLT adds outer S-VLAN Tags with TPID 88A8 for each service. Tags are added by the devices in the upstream direction and removed in the downstream direction. In this example, GEM Ports are configured for four classes of traffic or services. All GEM Ports map to a single T-CONT. Frames are classified to GEM Ports based on the VLAN Tag priority bits (PCP value) received in the frame.

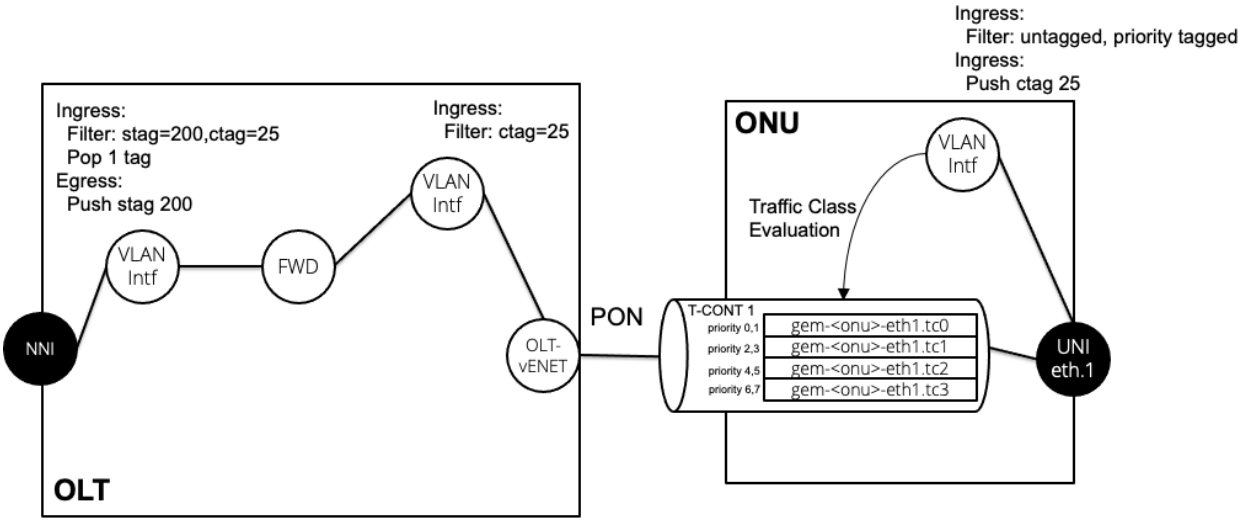


Figure 14 - Multi-XGem Service Example

Example directory: /opt/tibit/netconf/examples/bbf/multixgem_service

The example breaks the configuration into the following NETCONF <edit-config> requests:

1. **onu-interfaces.xml** - Create ONU related interfaces, including the ANI, vANI, UNI, and OLT-vENET.
2. **link-table.xml** - Create entries in the Link Table to associate the vANI to an ANI and OLT-vENET to a UNI.
3. **onu-classifiers.xml** - Create classifiers and a QoS profile on the ONU for classifying PCP values to GEM Ports.
4. **onu-vlan-sub-interfaces.xml** - Create a VLAN Subinterface on the ONU that adds an inner CTAG to untagged and priority tagged frames ingressing the UNI.
5. **gemports.xml** - Creates a Traffic Descriptor Profile (upstream SLA), T-CONT, and four GEM Ports for the Add CTag service.

6. **olt-classifiers.xml** - Create classifiers and a QoS profile on the OLT for classifying PCP values to GEM Ports.
7. **olt-1to1-forwarding.xml** - Creates VLAN Subinterfaces and an L2 Forwarder that configures the OLT to add an outer STAG to frames received from the ONU.

Running the example:

The following configures the ONU with serial number ALPHe30cadcf to add an inner CTAG with VID 25 on UNI port 1, and configures the OLT on switch port 1/0/1 to add outer STAGs with VID 200 to frames received from the ONU.

```
$ cd /opt/tibit/netconf/examples/bbf/multixgem_service
$ ./config_multixgem_svc.py --olt_port 1/0/1 --onu ALPHe30cadcf --uni
5 --olt_tag 200 --onu_tag 25
```

The following deletes the Add CTag configuration for the ONU with the specified serial number and from OLT configuration for the specified switch port.

```
$ cd /opt/tibit/netconf/examples/bbf/multixgem_service
$ ./disable_svc.py --olt_port 1/0/1 --onu ALPHe30cadcf --uni 5 --
olt_tag 200 --onu_tag 25
```

Full help text for the example script is shown below:

```
$ ./multixgem_service/config_multixgem_svc.py --help
usage: config_multixgem_svc.py [--help] [-h HOST] --olt_port OLT_PORT
                               --olt_tag OLT_TAG --onu ONU --onu_tag ONU_TAG
                               [-w PASSWD] [-p PORT]
                               [-t TRAFFIC_DESCRIPTOR_PROFILE]
                               [--uni UNI_PORT] [-u USER] [-v]

optional arguments:
  --help                Show this help message and exit.
  -h HOST, --host HOST  NETCONF Server IP address or hostname. (default:
                        127.0.0.1)
  --olt_port OLT_PORT   OLT Port number. This could be a logical port number
                        or a physical port number representing the switch port
                        (e.g., LLDP switch port ID) (default: None)
  --olt_tag OLT_TAG     Tag to be added by the OLT (default: None)
  --onu ONU             ONU Serial Number (e.g., TBITc84c00df) (default: None)
  --onu_tag ONU_TAG     Tag to be added by the ONU (default: None)
  -w PASSWD, --passwd PASSWD
                        Password. If no password is provided, attempt to read
                        it from .nc_edit_auth. (default: None)
```

```
  -p PORT, --port PORT  NETCONF Server port number. (default: 830)
  -t TRAFFIC_DESCRIPTOR_PROFILE, --traffic_descriptor_profile
TRAFFIC_DESCRIPTOR_PROFILE
                        Configure the traffic descriptor profile name (e.g.,
                        SLA) for this service. (default: Max)
  --uni UNI_PORT        UNI port number 1..5 (default: 1)
  -u USER, --user USER  Username. (default: None)
  -v, --verbose         Verbose output. (default: False)
```

Add CTag Service with ONU-vENET interface

Prerequisites: Configure OLT on switch port 1/0/1. See the Create OLT example.

The Add CTag Service with ONU-vENET example is similar to the Add CTag example above except that the VLAN configuration is automatically applied to all UNIs on the ONU. This example configures the ONU to add an inner C-VLAN Tag with TPID 8100 and the OLT to add an outer S-VLAN Tag with TPID 88A8, and is shown in Figure 15. Tags are added by the devices in the upstream direction and removed in the downstream direction. In this example, the VLAN tagging and forwarding is configured on the ONU-vENET interface only. The Netconf Server automatically applies this configuration for all UNIs discovered on the ONU.

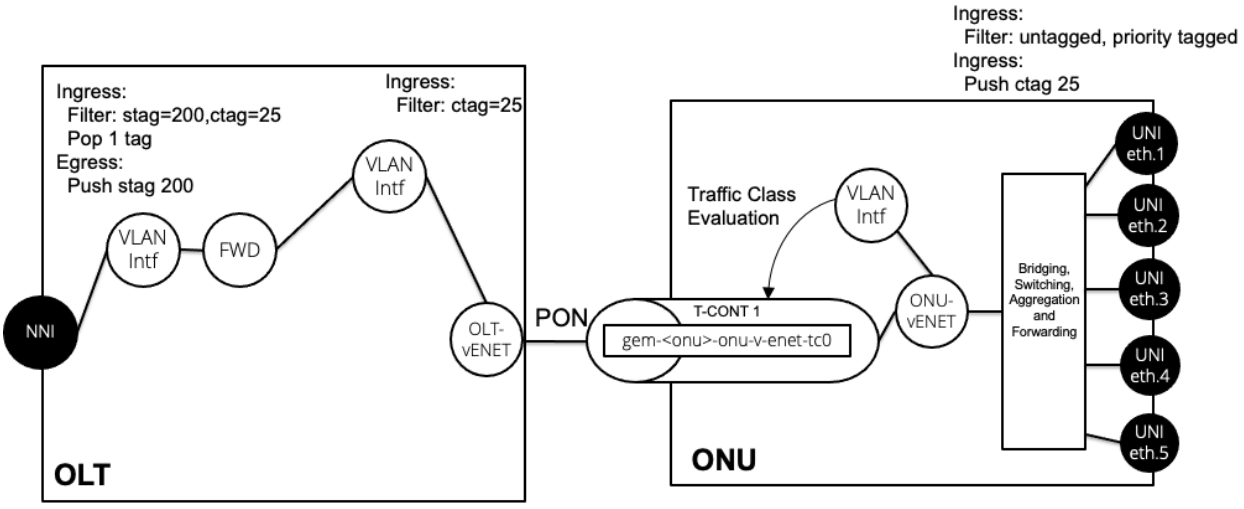


Figure 15 - Add CTag Service with ONU-vENET Example

Example directory: /opt/tibit/netconf/examples/bbf/add_ctag_onu_venet_service

The example breaks the configuration into the following NETCONF <edit-config> requests:

1. **onu-interfaces.xml** - Create ONU related interfaces, including the ANI, vANI, OLT-vENET, and ONU-vENET.
2. **link-table.xml** - Create entries in the Link Table to associate the vANI to an ANI and OLT-vENET to an ONU-vENET.
3. **onu-vlan-sub-interfaces.xml** - Create a VLAN Subinterface on the ONU that adds an inner CTAG to untagged and priority tagged frames ingressing the ONU-vENET.
4. **gemports.xml** - Creates a Traffic Descriptor Profile (upstream SLA), T-CONT, and GEM Port for the Add CTag service.

5. **olt-1to1-forwarding.xml** - Creates VLAN Subinterfaces and an L2 Forwarder that configures the OLT to add an outer STAG to frames received from the ONU.

Running the example:

The following configures the ONU with serial number ALPHe30cadcf to add an inner CTAG with VID 25 on UNI port 1, and configures the OLT on switch port 1/0/1 to add outer STAGs with VID 200 to frames received from the ONU.

```
$ cd /opt/tibit/netconf/examples/bbf/add_ctag_onu_venet_service
$ ./config_add_ctag_onu_venet_svc.py --olt_port 1/0/1 --onu
ALPHe30cadcf --olt_tag 200 --onu_tag 25
```

The following deletes the Add CTag configuration for the ONU with the specified serial number and from OLT configuration for the specified switch port.

```
$ cd /opt/tibit/netconf/examples/bbf/add_ctag_onu_venet_service
$ ./disable_svc.py --olt_port 1/0/1 --onu ALPHe30cadcf --olt_tag 200 -
-onu_tag 25
```

Full help text for the example script is shown below:

```
$ ./add_ctag_onu_venet_service/config_add_ctag_onu_venet_svc.py --help
usage: config_add_ctag_onu_venet_svc.py [--help] [-h HOST] --olt_port OLT_PORT --
olt_tag OLT_TAG --onu ONU --onu_tag ONU_TAG [-w PASSWD] [-p PORT]
                                        [-t TRAFFIC_DESCRIPTOR_PROFILE] [-u USER] [-v]

optional arguments:
  --help                Show this help message and exit.
  -h HOST, --host HOST  NETCONF Server IP address or hostname. (default: 127.0.0.1)
  --olt_port OLT_PORT   OLT Port number. This could be a logical port number or a
physical port number representing the switch port (e.g., LLDP switch port ID)
(default:
                        None)
  --olt_tag OLT_TAG     Tag to be added by the OLT (default: None)
  --onu ONU             ONU Serial Number (e.g., TBITc84c00df) (default: None)
  --onu_tag ONU_TAG     Tag to be added by the ONU (default: None)
  -w PASSWD, --passwd PASSWD
                        Password. If no password is provided, attempt to read it
from .nc_edit_auth. (default: None)
  -p PORT, --port PORT  NETCONF Server port number. (default: 830)
  -t TRAFFIC_DESCRIPTOR_PROFILE, --traffic_descriptor_profile
TRAFFIC_DESCRIPTOR_PROFILE
                        Configure the traffic descriptor profile name (e.g., SLA) for
this service. (default: Max)
```

```
-u USER, --user USER  Username. (default: None)
-v, --verbose         Verbose output. (default: False)
```

Server Administration

This section provides information on how to configure and manage the MCMS Netconf Server.

Configuration

The Netconf Server configuration consists of two parts. The MCMS Netconf Server Database Connector configuration is specified in the file /etc/tibit/netconf/NetconfInit.json. Edit this file to modify MongoDB connection information, logging, and other Database Connector configuration parameters. See Database Connection Configuration for more information on configuring the database connector.

The Netconf SSH server, protocol, and users are managed through configuration of standard YANG models defined by IETF listed in the table below. Configurations from these YANG models are loaded from the startup datastore when the server is started.

YANG Model	Description
ietf-netconf-acm	Configuration for the Network Access Control Model (NACM), including the definition of groups and ACL rules that define user permissions for accessing data and actions on the Netconf Server. See Network Access Control (NACM).
ietf-netconf-server	Configuration for the SSH server and user access to the Netconf Server, including the listen IP address, TCP port number, SSH authentication methods, and the list of users permitted access to the server. See Network Configuration for more information on configuring SSH server settings. See User Management for more information on managing Netconf users.

Server Configuration Tools

The Netconf Server SSH settings and users are configured through standard YANG models. As such, the server can be administered using any standard Netconf client. This section describes several tools packaged with the Netconf Server that can be used to configure the server.

Sysrepo Configuration Tool (sysrepocfg)

The sysrepo configuration tool is used to modify the YANG datastore directly without utilizing the Netconf SSH protocol. This tool is used for local configuration of the server for initial server setup and for sensitive security settings that have NACM rules in place to deny access from the Netconf Interface. Listen IP address, TCP port number, users, and NACM are examples of settings that may need to be configured locally using the sysrepocfg tool. A Netconf Server user is not required, nor is NACM applied when using sysrepocfg. Root level access is required to use this tool.

Path

`/opt/tibit/netconf/bin/sysrepocfg`

Usage

```
$ /opt/tibit/netconf/bin/sysrepocfg --help
sysrepocfg - sysrepo configuration manipulation tool, compiled with libsysrepo v1.4.122 (SO v5.6.38)

Usage:
  sysrepocfg <operation-option> [other-options]

Available operation-options:
  -h, --help                    Prints usage help.
  -V, --version                 Prints only information about sysrepo version.
  -I, --import[=<file-path>]    Import the configuration from a file or STDIN.
  -X, --export[=<file-path>]    Export configuration to a file or STDOUT.
  -E, --edit[=<file-path>/<editor>]
                                Edit configuration data by merging (applying) a configuration (edit) file or
                                by editing the current datastore content using a text editor.
  -R, --rpc[=<file-path>/<editor>]
                                Send a RPC/action in a file or using a text editor. Output is printed to STDOUT.
  -N, --notification[=<file-path>/<editor>]
                                Send a notification in a file or using a text editor.
  -C, --copy-from <file-path>/<source-datastore>
                                Perform a copy-config from a file or a datastore.
  -W, --new-data <file-path>    Set the configuration from a file as the initial one for a new module only scheduled
                                to be installed. Is useful for modules with mandatory top-level nodes.

       When both a <file-path> and <editor>/<target-datastore> can be specified, it is always first checked
       that the file exists. If not, then it is interpreted as the other parameter.
       If no <file-path> and no <editor> is set, use text editor in $VISUAL or $EDITOR environment variables.

Available other-options:
  -d, --datastore <datastore>  Datastore to be operated on, "running" by default ("running", "startup",
                               "candidate", or "operational") (import, export, edit, copy-from op).
  -m, --module <module-name>   Module to be operated on, otherwise it is operated on full datastore
                               (import, export, edit, copy-from, mandatory for new-data op).
  -x, --xpath <xpath>          XPath to select (export op).
  -f, --format <format>        Data format to be used, by default based on file extension or "xml" if not applicable
                               ("xml", "json", or "lyb") (import, export, edit, rpc, notification, copy-from, new-data op).
  -l, --lock                   Lock the specified datastore for the whole operation (edit op).
  -n, --not-strict             Silently ignore any unknown data (import, edit, rpc, notification, copy-from op).
  -p, --depth <number>         Limit the depth of returned subtrees, 0 so unlimited by default (export op).
  -t, --timeout <seconds>      Set the timeout for the operation, otherwise the default one is used.
  -w, --wait                   Wait for all the callbacks to be called on a data change including DONE or ABORT.
```

```
  -e, --defaults <wd-mode>     Print the default values, which are hidden by default ("report-all",
                               "report-all-tagged", "trim", "explicit", "implicit-tagged") (export, edit, rpc op).
  -v, --verbosity <level>      Change verbosity to a level (none, error, warning, info, debug) or number (0, 1, 2, 3, 4).
```

Examples

Export All Configuration from the Startup Datastore

```
$ sudo /opt/tibit/netconf/bin/sysrepocfg --export --format=xml --datastore=startup
<nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
  <enable-nacm>true</enable-nacm>
  <read-default>deny</read-default>
  <write-default>deny</write-default>
  <exec-default>deny</exec-default>
  <groups>
    <group>
      <name>admin</name>
      <user-name>netconf-admin</user-name>
       ...
```

Export All Configuration from the Running Datastore

```
$ sudo /opt/tibit/netconf/bin/sysrepocfg --export --format=xml --datastore=running
<nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
  <enable-nacm>true</enable-nacm>
  <read-default>deny</read-default>
  <write-default>deny</write-default>
  <exec-default>deny</exec-default>
  <groups>
    <group>
      <name>admin</name>
      <user-name>netconf-admin</user-name>
       ...
```

Replace ietf-server-acm Configuration

```
$ cat nacm-config.xml
<nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm"
xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0">
  <!-- Enable NACM -->
  <enable-nacm>true</enable-nacm>
  <!-- Default deny all -->
  <read-default>deny</read-default>
  <write-default>deny</write-default>
  <exec-default>deny</exec-default>
  <groups>
    <group>
      <!-- Administrators -->
```

```
      <name>admin</name>
      <user-name>netconf-admin</user-name>
    </group>
    <group>
      <!-- Read-only users -->
      <name>read-only</name>
      <user-name>netconf-readonly</user-name>
    </group>
    <group>
      <!-- Users responsible for Tibit device and service configuration →
       ...

# Apply configuration to the running datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --module=ietf-netconf-
acm --format=xml --datastore=running

# Apply configuration to the startup datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --module=ietf-netconf-
acm --format=xml --datastore=startup
```

Replace ietf-netconf-server Configuration

```
$ cat server-config.xml
<netconf-server xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-server">
  <listen>
    <endpoint>
      <name>default-ssh</name>
      <ssh>
        <tcp-server-parameters>
          <local-address>0.0.0.0</local-address>
          <keepalives>
            <idle-time>1</idle-time>
            <max-probes>10</max-probes>
            <probe-interval>5</probe-interval>
          </keepalives>
        </tcp-server-parameters>
        <ssh-server-parameters>
          <server-identity>
            <host-key>
              <name>default-key</name>
              <public-key>
                <keystore-reference>genkey</keystore-reference>
              </public-key>
             ...

# Apply configuration to the running datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=server-config.xml --module=ietf-
netconf-server --format=xml --datastore=running
```

```
# Apply configuration to the startup datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=server-config.xml --module=ietf-
netconf-server --format=xml --datastore=startup
```

Netconf User Management Tool (netconf-users)

The Netconf User Management tool is a utility for user and NACM group management. This tool is used to configure Netconf Server users and to assign users to NACM groups. Note that only NACM group membership can be managed by this tool, but the groups themselves and associated ACL rules must be configured directly through the ietf-netconf-acm YANG model. Root level access is required to use the netconf-users tool.

Path

```
/opt/tibit/netconf/bin/netconf-users
```

Usage

```
usage: netconf-users.py [-h] [-a] [-d] [-g GROUP] [-u USER]

optional arguments:
  -h, --help            show this help message and exit
  -a, --add             Add a user to the Netconf Server or NACM Group.
                        (default: None)
  -d, --delete          Delete a user from the Netconf Server or NACM Group.
                        (default: None)
  -g GROUP, --group GROUP
                        NACM group name. (default: None)
  -u USER, --user USER  Username (default: None)
```

Examples

Display All Netconf Server Users

```
$ sudo /opt/tibit/netconf/bin/netconf-users

Username             Full Name                  System UID     NACM Groups
--------             ---------                  ----------     -----------
netconf-readonly     Netconf Read-only User     1003           read-only
netconf-admin        Netconf Administrator User 1002           admin
netconf-service                                 1006           service-provisioning
```

Creating a Netconf User

```
$ sudo /opt/tibit/netconf/bin/netconf-users --add --user netconf-readonly
Adding netconf-readonly to the Netconf Server.
```

Adding a Netconf User to a NACM Group

```
$ sudo /opt/tibit/netconf/bin/netconf-users --add --user netconf-readonly --group
read-only
Adding netconf-readonly to NACM group read-only.
```

Netconf Client Command Line Interface Tool (netopeer2-cli)

The netopeer2-cli tool is a command line NETCONF client. This tool uses the standard NETCONF protocol for managing the server. Unlike the sysrepocfg tool, a valid Netconf user must be configured and NACM rules are enforced when managing the server using netopeer2-cli tool.

Path

`/opt/tibit/netconf/bin/netopeer2-cli`

Usage

```
$ /opt/tibit/netconf/bin/netopeer2-cli
> help
Available commands:
  auth            Manage SSH authentication options
  knownhosts      Manage the user knownhosts file
  cert            Manage trusted or your own certificates
  crl             Manage Certificate Revocation List directory
  outputformat    Set the output format of all the data
  searchpath      Set the search path for models
  verb            Change verbosity
  version         Print Netopeer2 CLI version
  disconnect      Disconnect from a NETCONF server
  status          Display information about the current NETCONF session
  connect         Connect to a NETCONF server
  listen          Wait for a Call Home connection from a NETCONF server
  quit            Quit the program
  help            Display commands description
  editor          Set the text editor for working with XML data
  cancel-commit   ietf-netconf <cancel-commit> operation
  commit          ietf-netconf <commit> operation
  copy-config     ietf-netconf <copy-config> operation
  delete-config   ietf-netconf <delete-config> operation
  discard-changes ietf-netconf <discard-changes> operation
  edit-config     ietf-netconf <edit-config> operation
  get             ietf-netconf <get> operation
  get-config      ietf-netconf <get-config> operation
  kill-session    ietf-netconf <kill-session> operation
  lock            ietf-netconf <lock> operation
  unlock          ietf-netconf <unlock> operation
  validate        ietf-netconf <validate> operation
  subscribe       notifications <create-subscription> operation
```

```
  get-schema      ietf-netconf-monitoring <get-schema> operation
  get-data        ietf-netconf-nmda <get-data> operation
  edit-data       ietf-netconf-nmda <edit-data> operation
  user-rpc        Send your own content in an RPC envelope
  timed           Time all the commands (that communicate with a server) from issuing
a RPC to getting a reply
  ?               Display commands description
  exit            Quit the program
>
```

Examples

Connect to the Netconf Server

```
$ /opt/tibit/netconf/bin/netopeer2-cli
> connect --host localhost --login netconf-admin
netconf-admin@localhost password:
>
```

Get Netconf Server Status

```
> get  --filter-xpath=/tibit-netconf:netconf-state
DATA
<netconf-state xmlns="urn:com:tibitcom:ns:yang:netconf">
  <database>
    <ip-address>127.0.0.1</ip-address>
    <port>27017</port>
    <name>tibit_pon_controller</name>
    <status>online</status>
  </database>
  <diagnostics>
    <netconf-sr-change>2</netconf-sr-change>
       ...
>
```

Netconf Client Python Library (ncclient)

The ncclient package is a Python library that facilitates client-side scripting and application development around the NETCONF protocol. This library is used by several examples provided with the Netconf Server package. See https://pypi.org/project/ncclient/ for more information on how to install and use the ncclient library.

Database Connector Configuration

This section describes the configuration for the Netconf Server Database Connector.

MongoDB Connection

The MongoDB connection configuration is specified under the 'MongoDB' section in NetconfInit.json. Edit the file to modify the MongoDB connection configuration parameters. Restart the MCMS Netconf Server to apply the changes.

```
$ cat /etc/tibit/netconf/NetconfInit.json
{
    "MongoDB": {
        "auth_db": "tibit_users",
        "auth_enable": false,
        "ca_cert_path": "/etc/tibit/ca.pem",
        "host": "127.0.0.1",
        "name": "tibit_pon_controller",
        "netconf_db": "tibit_netconf",
        "password": "",
        "port": "27017",
        "tls_enable": false,
        "username": ""
    }
}
```

The MongoDB configuration parameters are described in the table below.

Parameter	Default	Description
auth_db	tibit_users	MongoDB authentication database name. Only applies when 'auth_enable' is true.
auth_enable	false	Enable MongoDB authentication.
ca_cert_path	/etc/tibit/ca.pem	Path to CA or self-signed certificate. Only applies when 'tls_enable' is true.
db_uri		MongoDB connection URI with the format mongo mongodb://<username>:<password>@<host>... **Note:** all other database connection parameters are ignored, except for the database name fields.
dns_srv	false	Acquire replica set members and other connection information from a DNS SRV record.

		When set to ‘true’, the host parameter must reference a DNS server.
host	127.0.0.1	IP address or hostname of the MongoDB server. When dns_srv is ‘true’, IP address or hostname of the DNS server.
name	tibit_pon_controller	Name of the PON Controller database that contains the Controller, OLT, and ONU device configuration and state.
netconf_db	tibit_netconf	Name of the NETCONF database that is used as persistent storage for BBF YANG configuration tables. See BBF Configuration Persistence for more information on how MongoDB is used to persist BBF configuration.
password		MongoDB authentication password. Only applies when ‘auth_enable’ is true.
port	27017	TCP port number the MongoDB server is listening on.
replica_set_enable	false	Enables connecting to a MongoDB replica set for purposes of redundancy.
replica_set_hosts	[127.0.0.1:27017]	List of Replica Set members used for connecting to a MongoDB Replica Set for purposes of redundancy. An IP address or hostname and optional port number is specified for each member of the Replica Set. This field is only used when replica_set_enable is set to ‘true’.
replica_set_name	rs0	Replica set name. This field is required when replica_set_enable is set to ‘true’.
tls_enable	false	Enable TLS encryption for the MongoDB connection.
username		MongoDB authentication username. Only applies when ‘auth_enable’ is true.

Logging Configuration

The logging configuration is specified under the ‘Logging’ section in NetconfInit.json.

NOTE: The logging levels in this configuration are only effective during the initial boot. After the connection to MongoDB is established the logging levels will be updated to reflect the primary configuration in the NETCONF-CFG collection.

```
{
    "Logging": {
        "Filename"  : "/var/log/tibit/netconf.log",
        "FileCount" : 3,
        "FileSize"  : 1024000,
        "Netconf"   : {
            "Console" : "INFO",
            "File"    : "INFO",
            "Syslog"  : "INFO"
        }
    },
}
```

The secondary logging configuration parameters are described in the table below.

Parameter	Default	Description
Filename	/var/log/tibit /netconf.log	Path and name of the log file.
FileCount	3	Number of log files to rotate (e.g., netconf.log.1, netconf.log.2).
FileSize	1024000	Maximum size in bytes of the log file before rotating.
Netconf.Console	INFO	Logging level for messages logged to the console: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the MCMS Netconf Server Database Connector during boot only.
Netconf.File	INFO	Logging level for messages logged to the netconf.log file: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the MCMS Netconf Server Database Connector during boot only.

Netconf.Syslog	INFO	Logging level for messages logged to syslog: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the MCMS Netconf Server Database Connector during boot only.

Database Connector MongoDB Collections

This section describes the MongoDB configuration (CFG), state (STATE), and log (SYSLOG) collections used for managing the Netconf Server Database Connector service.

NETCONF-CFG

The NETCONF-CFG collection is used to store the logging level configuration for the Database Connector. The Database Connector logging levels can be managed through this collection.

The configuration parameters are described in the table below.

Parameter	Default	Description
Logging.Console	INFO	Logging level for messages logged to the console: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the MCMS Netconf Server Database Connector only.
Logging.File	INFO	Logging level for messages logged to the netconf.log file: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the MCMS Netconf Server Database Connector only.
Logging.Syslog	INFO	Logging level for messages logged to syslog: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the MCMS Netconf Server Database Connector only.
Logging.MongoDB	INFO	Logging level for messages logged to the netconf.log file: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the MCMS Netconf Server Database Connector only.
Logging.Max SYSLOG Size	50000000	MongoDB SYSLOG-NETCONF maximum size for storing Database Connector logs. When the limit is reached, the oldest logs will be removed to make room for new ones.

Note that while the Database Connector is booting, the logging levels in NetconfInit.json will be used. After the connection to MongoDB has been established, the logging levels will be updated to reflect the NETCONF-CFG collection.

NETCONF-STATE

The NETCONF-STATE collection is used to store information about the current state of the Database Connector.

The information contained in the NETCONF-STATE collection is described in the table below:

Attribute	Description
Database	
IP Address	IP address used to connect to the database.
Port	TCP port number used to connect to the database.
Name	Name of the database.
Status	The state of the NETCONF Connectors's connection to the database.
Replica Set	Replica set information containing an enabled/disabled flag and the replica set name.
Servers	A List of all known MongoDB servers. Contains IP address, port, latency, status, and error status information for each server.
Diagnostics	
SR Change	Count of Netconf change request callbacks.
SR Get	Count of Netconf get operational data request callbacks.
SR RPC	Count of Netconf rpc request callbacks.
DB Init	Count of PON Controller database (re)initializations.
PonCntlDB Update	Count of PON Controller database updates.
PonCntlDB Drop	Count of PON Controller database drop requests.
NetconfDB Update	Count of Netconf database updates.
NetconfDB Drop	Count of Netconf database drop requests.

Logging	
Console	Severity level for log messages sent to the console.
File	Severity level for log messages sent to the log file.
Syslog	Severity level for log messages sent to the Syslog.
MongoDB	Severity level for logs messages sent to the database.
Max SYSLOG Size	The maximum size of the netconf log collection in MongoDB. When the maximum size is reached, the oldest entries are removed to make space for new entries.
Stats	
DB Commands	MongoDB command statistics. Contains statistics for find, update, listcollections, insert, distinct, and aggregate commands. Statistics include average latency, min latency, max latency, success count, and failure count.
DB Connections	MongoDB connection statistics. Includes the maximum, minimum, and average number of MongoDB connections reported.
System	
Hostname	Fully qualified domain name of the system hosting the NETCONF server.
System Name	Name of the system hosting the NETCONF server.
NETCONF	
Version	NETCONF Database Connector version.
Build Date	NETCONF Database Connector build date.
Build SHA	NETCONF Database Connector build simple hashing algorithm.
Build OS	NETCONF Database Connector OS.
DB Version	NETCONF Database Connector database version.
Txn ID	NETCONF Database Connector transaction ID.

SYSLOG-NETCONF

The SYSLOG-NETCONF collection is used to store system logs emitted by the Database Connector. Logs are ordered from oldest to newest. Once the configured size limit is reached, the oldest logs are removed to make room for new logs. The logging level and maximum collection size are configurable via the NETCONF-CFG collection.

The information contained in each log is described in the table below:

Attribute	Description
time	The timestamp when the log was created.
device ID	The device that was responsible for emitting the log.
severity	The logging level associated with the system log.
message	A message string containing the information of the log.
valid	A flag indicating the validity of the log. Logs are typically marked as invalid when a user clears the database log.

Network Configuration

Network protocol, transport, authentication, authorization, and logging parameters can be configured by one of the following methods:

- Standard NETCONF IETF YANG models
- MCMS Netconf Server 'sysrepocfg' utility

Server IP Address and TCP Port Number

The server listens on IP address 0.0.0.0 (all IP addresses) and TCP port 830 by default. This section provides instructions for using the sysrepocfg utility to modify the listen IP address and TCP port from default values.

See the following examples for more information on modifying the server's listen IP address and port number:

- /opt/tibit/netconf/examples/nc_edit_ssh_port.sh

To modify the listen IP address and TCP port:

1) Create a temporary config.xml file with the following XML content, replacing *ip address* and *port number* with the desired values.

```
$ cat ./config.xml
<netconf-server xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-server">
   <listen>
     <endpoint>
       <name>all-interfaces</name>
       <ssh>
         <address>ip address</address>
         <port>port number</port>
         <host-keys>
           <host-key>
             <name>imported SSH key</name>
             <public-key>ssh_host_rsa_key</public-key>
           </host-key>
         </host-keys>
       </ssh>
     </endpoint>
   </listen>
 </netconf-server>
```

2) Use the sysrepocfg utility with the *import* option to apply the configuration changes.

```
$ sudo /opt/tibit/netconf/bin/sysrepocfg \
      --import=./config.xml \
      --datastore=startup \
      --format=xml \
      ietf-netconf-server
The new configuration was successfully applied.
```

3) Restart the MCMS Netconf Server

```
$ sudo systemctl restart tibit-netconf
```

The script /opt/tibit/netconf/examples/nc_edit_ssh_port.sh provides an example of updating both running and startup datastores, which allows the changes to take effect without restarting the server.

SSH Algorithms

The Netconf Server supports configuration of the cryptography algorithms enabled for the SSH server. The file /etc/tibit/netconf/libssh_server_config contains the SSH algorithm configuration for the Netconf Server. The table below lists the SSH algorithms supported by the Netconf Server.

Type	Attribute	Description
Encryption	ciphers	Supported encryption algorithms: aes256-cbc, aes192-cbc, aes128-cbc, 3des-cbc aes256-gcm@openssh.com, aes128-gcm@openssh.com, aes256-ctr, aes192-ctr, aes128-ctr, chacha20-poly1305@openssh.com **Default:** aes256-gcm@openssh.com, aes128-gcm@openssh.com, aes256-ctr, aes192-ctr, aes128-ctr, chacha20-poly1305@openssh.com
Key Exchange	kexalgorithms	Supported key exchange algorithms: ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp521, curve25519-sha256, curve25519-sha256@libssh.org, diffie-hellman-group1-sha1, diffie-hellman-group18-sha512, diffie-hellman-group16-sha512, diffie-hellman-group14-sha256 **Default:** curve25519-sha256, curve25519-sha256@libssh.org, diffie-hellman-group18-sha512, diffie-hellman-group16-sha512, diffie-hellman-group14-sha256
MAC	macs	Supported MAC algorithms: hmac-md5, hmac-sha1, hmac-sha2-256, hmac-sha2-512 **Default:** hmac-md5, hmac-sha1, hmac-sha2-256, hmac-sha2-512
Server Host Key	hostkeyalgorithms	Supported server host key algorithms: ssh-ed25519, ssh-rsa, rsa-sha2-512, rsa-sha2-256 **Default:** rsa-sha2-512, rsa-sha2-256

To modify the SSH algorithms enabled on the Netconf server:

1) Edit the file /etc/tibit/netconf/libssh_server_config and modify the parameters shown below with the desired values.

```
$ sudo vi /etc/tibit/netconf/libssh_server_config

$ sudo cat /etc/tibit/netconf/libssh_server_config
#disabled kexalgorithms=ecdh-sha2-nistp256,ecdh-sha2-nistp384,ecdh-
sha2-nistp521,diffie-hellman-group1-sha1
kexalgorithms=curve25519-sha256,curve25519-sha256@libssh.org,diffie-
hellman-group18-sha512,diffie-hellman-group16-sha512,diffie-hellman-
group14-sha256
#disabled hostkeyalgorithms=ssh-ed25519,ssh-rsa
hostkeyalgorithms=rsa-sha2-512,rsa-sha2-256
#disabled ciphers=aes256-cbc,aes192-cbc,aes128-cbc,3des-cbc
ciphers=aes256-gcm@openssh.com,aes128-gcm@openssh.com,aes256-
ctr,aes192-ctr,aes128-ctr,chacha20-poly1305@openssh.com
#macs=
```

2) Restart the MCMS Netconf Server

```
$ sudo systemctl restart tibit-netconf
```

SSH Authentication Methods

The Netconf Server supports three SSH authentication methods: public key, password, and interactive. The table below lists the SSH authentication methods supported by the Netconf Server.

Auth Method	YANG Attribute	Description
Public Key	<publickey/>	Use public key client authentication. See Public Key Authentication for information on setting up a public key.
Password	<password/>	Use password authentication.
Keyboard-Interactive	<other> interactive </other>	Use SSH keyboard interactive authentication, which in the future, could be used for more advanced authentication methods such as Linux Pluggable Authentication Modules (PAM). **Note:** The 'interactive' method is the same as 'password' authentication in the current version of the Netconf Server.

Enable Host Key Authentication Only

Use the sysrepocfg tool to configure the Netconf Server to disable password and interactive authentication methods and enable public key authentication only. To disable password and interactive authentication, remove <password/> and <other>interactive</other> from the server endpoint configuration as shown below.

```
<supported-authentication-methods>
  <publickey/>
  <password/>
  <other>interactive</other>
</supported-authentication-methods>
```

Example

```
$ cat server-config.xml
 <netconf-server xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-server">
    <listen>
      <endpoint>
        <name>default-ssh</name>
        <ssh>
          <tcp-server-parameters>
            <local-address>0.0.0.0</local-address>
```

```
            <keepalives>
              <idle-time>1</idle-time>
              <max-probes>10</max-probes>
              <probe-interval>5</probe-interval>
            </keepalives>
          </tcp-server-parameters>
          <ssh-server-parameters>
            <server-identity>
              <host-key>
                <name>default-key</name>
                <public-key>
                  <keystore-reference>genkey</keystore-reference>
                </public-key>
              </host-key>
            </server-identity>
            <client-authentication>
              <supported-authentication-methods>
                <publickey/>
              </supported-authentication-methods>
              <users\>
            </client-authentication>
          </ssh-server-parameters>
        </ssh>
      </endpoint>
    </listen>
  </netconf-server>

# Apply configuration to the running datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=server-config.xml --module=ietf-
netconf-server --format=xml --datastore=running

# Apply configuration to the startup datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=server-config.xml --module=ietf-
netconf-server --format=xml --datastore=startup
```

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089

Phone: 888.JUNIPER (888.586.4737)
Web https://support.juniper.net/support/requesting-support

User Management

The Netconf Server uses the local Linux server's users and passwords, and supports the same SSH authentication features provided by the native Linux SSH server, including password and public key authentication. Netconf users, passwords, and public keys are configured on the server using standard Linux tools such as useradd, usermod, userdel, passwd, and ssh-keygen.

In addition to creating the user in Linux, Netconf Users must be added to the ietf-netconf-server configuration, which permits a specific Linux user access to the Netconf Interface. If the user is not added to in ietf-netconf-server, the user will not be able to log in to the Netconf Server.

Netconf Users are configured in conjunction with the Network Configuration Access Control Model (NACM) to provide secure access to data through the Netconf interface. See NACM for more information regarding the Netconf security model.

ietf-netconf-server

The Netconf Server implements the following draft version of the ietf-netconf-server YANG model.

ietf-netconf-server@2019-07-02:
NETCONF Client and Server Models
https://tools.ietf.org/html/draft-ietf-netconf-netconf-client-server-14
YANG Groupings for SSH Clients and SSH Servers
https://tools.ietf.org/html/draft-ietf-netconf-ssh-client-server-14

The current version of the Netconf Server has limited support for ietf-netconf-server user configuration. Only configuration of the user's name is supported. Configuring the user's password and public key is not supported through ietf-netconf-server. Instead, the user's password and public key must be managed through Linux. The MCMS Netconf Server applies the ietf-netconf-server user configuration attributes as described in the table below.

Attribute	Description
/ietf-netconf-server:netconf-server/listen/endpoint/ssh/ssh-server-parameters/ client-authentication/users	
name	Name of the Netconf User permitted to access the Netconf Interface. Note that a Linux user with the same name must also be configured on the system.

password	**Not supported.** Netconf User passwords are managed through Linux.
authorized-key	**Not supported.** Netconf User SSH public keys are managed through Linux.

Create a Netconf Server User

The following steps describe how to create a new Netconf User:

1) Use the 'useradd' command to create the Netconf User in Linux. This step may not be required if the user already exists in Linux.

```
sudo useradd -m netconf-user --comment "Netconf User"
```

2) Use the netconf-users tool to add a user to the Netconf Server. Alternatively, the sysrepocfg tool or NETCONF client application can be used to add a user to ietf-netconf-server.

```
sudo /opt/tibit/netconf/bin/netconf-users --add --user netconf-user
```

Delete a Netconf Server User

The following steps describe how to delete an existing Netconf User:

1) Use the netconf-users tool to remove a user from the Netconf Server. Alternatively, the sysrepocfg tool or NETCONF client application can be used to remove a user from ietf-netconf-server.

```
sudo /opt/tibit/netconf/bin/netconf-users --del --user netconf-user
```

2) Use the 'userdel' command to remove the Netconf User from Linux. This step is optional.

```
sudo userdel -r netconf-user
```

Public Key Authentication

This section provides instructions for configuring SSH public key authentication for use with the MCMS Netconf Server.

See the following examples for more information on configuring public key authentication:

- /opt/tibit/netconf/examples/n2cli_set_ssh_keys.sh

Note: the n2cli_set_ssh_keys.sh example configures the local netconf client to use public key authentication, which allows the examples to run without entering a password.

Add a Client Key

Note: These instructions are for installing a public key from the client. However, these instructions can also be used for installing a public key from the server itself. When installing the public key from the server, specify 'localhost' as <netconf server address>.

To add a client key from a Linux client:

1) (Optional) Create an SSH public private key pair using ssh-keygen from OpenSSH. Skip this step if the client key files already exist.

```
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_rsa.
Your public key has been saved in id_rsa.pub.
The key fingerprint is:
SHA256:byH4+ckNcMQ/KDPGsn1Jw0a9223/fdqhiIcBxRHUua0 tibit@ubuntu
The key's randomart image is:
+---[RSA 2048]----+
|        o++ .    |
|         + +     |
|        . + +    |
|       + + + o   |
|      o S O =    |
|       * & E + . |
|      . + O . ..o|
|         *.=. .o=|
|         .=.....B|
+----[SHA256]-----+
```

2) Install the public key for use with the NETCONF server using the ssh-copy-id utility.

From the client: `ssh-copy-id -i ~/.ssh/id_rsa.pub <server address>`

From the server: `ssh-copy-id -i ~/.ssh/id_rsa.pub localhost`

3) Test public key authentication using the ssh client and verify the connection without a password.

From the client: `ssh <netconf server address>`
From the server: `ssh localhost`

Remove a Client Key

To remove a client key:

1) SSH to the NETCONF server that has the key to be removed.

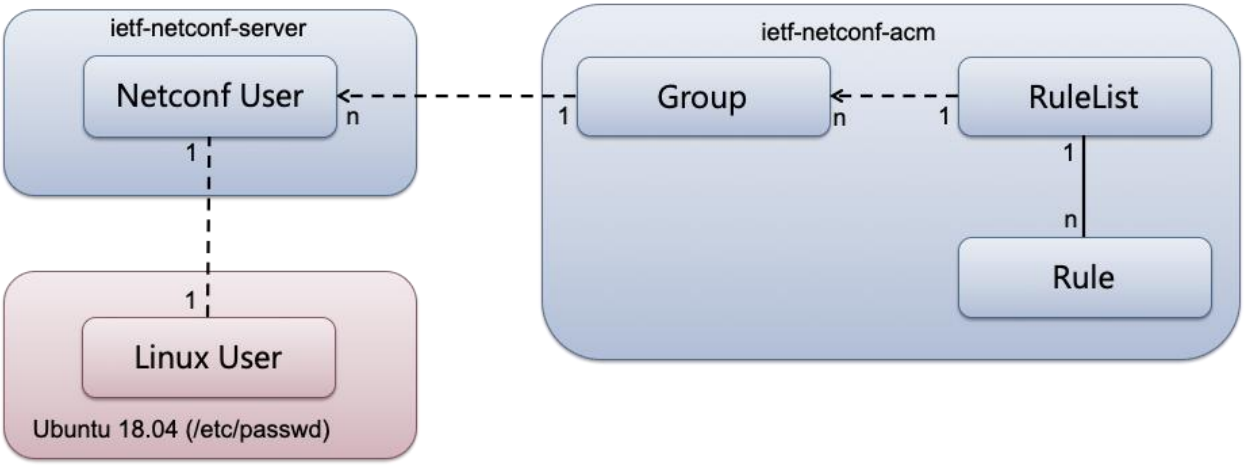
```
ssh <netconf server address>
```

2) Edit the authorized_keys file and remove the public key.

```
$ nano ~/.ssh/authorized_keys
```

Network Access Control (NACM)

The Netconf Server supports RFC8341 Network Configuration Access Control Model which defines role-based access controls for the Netconf interface. NACM works in conjunction with ietf-netconf-server user management through defining groups with ACL rules that provide CRUDX-style permissions to access data and actions: Read, Create, Update, Delete, and Execute. The NACM configuration model defined by IETF is shown in the figure below.



Users

The NACM configuration references users on the system, but configuration of the users themselves are managed separately from NACM. Users referenced in the NACM model must be configured in ietf-netconf-server on the endpoint configured for the Netconf Server. Also, the user must exist as a Linux system user on the system hosting the Netconf Server. See User Management for information on managing Netconf Server users.

Groups

Users are assigned to one or more Groups, which are a set of permissions that define access to the data and actions. Users can be assigned to zero, one, or more groups depending on the desired permissions. A group can reference one or more rule sets.

Rules

Rules are organized into RuleLists, which contain a set of ACL-style rules that define individual permissions to access a specific module, data node, notification, or protocol operation. A basic ‘permit’ or ‘deny’ permission is configured for each rule. Access operations permissions are summarized by the table below. RuleLists can reference one or more groups.

Permission	Operation	Description
create	<edit-config>, <edit-data>	Add or create a new data node instance to a datastore.
read	<get>, <get-config>, <get-data>	Read a data node instance from a datastore or receive a notification event.
update	<edit-config>, <edit-data>	Modify an existing data node instance in a datastore.
delete	<edit-config>, <edit-data>	Delete a data node instance from a datastore.
exec	Any RPC or Action	Execute an RPC or Action operation.

Enable NACM

Network access control is configured using the standard ietf-netconf-acm YANG model defined by RFC8341. By default, NACM is disabled when the Netconf Server is installed. Use the sysrepocfg tool to enable NACM with the following XML.

```
$ cat nacm-config.xml
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <enable-nacm>true</enable-nacm>
  </nacm>

# Apply configuration to the running datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --
module=ietf-netconf-acm --format=xml --datastore=running

# Apply configuration to the startup datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --
module=ietf-netconf-acm --format=xml --datastore=startup
```

An example for configuring NACM is included with the Netconf Server package. The following script and example XML is provided as part of the example:

- nacm-config.sh - Shell script that uses the sysrepocfg tool to configure NACM.
- nacm-example.xml - Example NACM, Group, and Rule configuration.

Configure Default Access

By default, all users have read-only access to data nodes from the Netconf Server when NACM is enabled. To override the default access configure the ietf-netconf-acm read-default, write-default, and exec-default attributes. Use the sysrepocfg tool with the following XML to modify the NACM configuration to deny all access by default.

```
$ cat nacm-config.xml
  <nacm xmlns="urn:ietf:params:xml:ns:yang:ietf-netconf-acm">
    <enable-nacm>true</enable-nacm>
    <read-default>deny</read-default>
    <write-default>deny</write-default>
    <exec-default>deny</exec-default>
  </nacm>

# Apply configuration to the running datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --
module=ietf-netconf-acm --format=xml --datastore=running

# Apply configuration to the startup datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --
module=ietf-netconf-acm --format=xml --datastore=startup
```

Create a NACM Group

By default, no NACM groups are configured for the Netconf Server. Use the sysrepocfg tool to configure NACM groups and associated rules.

```
$ cat nacm-config.xml
  <groups>
    <group>
      <!-- Administrators -->
      <name>admin</name>
      <user-name>netconf-admin</user-name>
    </group>
  </groups>
  <!-- Administrator User Group ACLs -->
  <rule-list>
    <name>admin</name>
    <group>admin</group>
    <rule>
      <!-- Allow full read/write access to all data nodes and RPCs -->
      <name>rw</name>
      <module-name>*</module-name>
      <access-operations>*</access-operations>
      <action>permit</action>
```

```
    </rule>
  </rule-list>

# Apply configuration to the running datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --module=ietf-netconf-
acm --format=xml --datastore=running

# Apply configuration to the startup datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=nacm-config.xml --module=ietf-netconf-
acm --format=xml --datastore=startup
```

Add a User to an Existing NACM Group

Use the netconf-users tool to add a user to an existing NACM group. Alternatively, the sysrepocfg tool or NETCONF client application can be used to add a user to a NACM group.

```
sudo /opt/tibit/netconf/bin/netconf-users --add --user netconf-user --group
read-only
```

Remove a User from a NACM Group

Use the netconf-users tool to remove a user from a NACM group. Alternatively, the sysrepocfg tool or NETCONF client application can be used to remove a user from the NACM group.

```
sudo /opt/tibit/netconf/bin/netconf-users --del --user netconf-user --group
read-only
```

Starting, Stopping, and Restarting the Software

The MCMS Netconf Server is composed of multiple processes that provide the complete NETCONF solution. Systemd scripts are used to manage the collection of processes as a single application. The MCMS Netconf Server processes are described in the table below.

Systemd Service	Process Name	Application Binary	Description
tibit-netopeer2-server.service	netopeer2-server	/opt/tibit/netconf/bin/netopeer2-server	Implements the NETCONF protocol according to RFC 6241.
tibit-netconf.service	tibit-netconf	/opt/tibit/netconf/bin/tibit-netconf	Tibit NETCONF Database Connector.

The tibit-netopeer2-server service is configured as a systemd dependency of the tibit-netconf service. This systemd dependency configuration allows all MCMS Netconf Server processes to be managed as a single application using systemd. The tibit-netopeer2-server service is started,

stopped, and restarted automatically when the tibit-netconf service is started, stopped, and restarted.

Server Status

Use the ‘systemctl status’ command to display the current status of the MCMS Netconf Server processes. All services should report ‘active (running)’ when the server is up and running under normal conditions. Unlike the systemd start, stop, and restart actions, the ‘systemd status’ command reports status for each service individually.

```
$ systemctl status tibit-netconf.service
 * tibit-netconf.service - Tibit Communications, Inc. NetCONF Server
   Loaded: loaded (/lib/systemd/system/tibit-netconf.service; enabled; vendor preset:
enabled)
   Active: active (running) since Thu 2020-05-28 09:16:06 EDT; 24s ago
  Process: 15905 ExecStartPre=/opt/tibit/netconf/bin/shm_clean.sh (code=exited,
status=0/SUCCESS)
 Main PID: 15909 (tibit-netconf)
    Tasks: 7 (limit: 4915)
   CGroup: /system.slice/tibit-netconf.service
           /opt/tibit/netconf/bin/tibit-netconf
           /opt/tibit/netconf/bin/tibit-netconf


$ systemctl status tibit-netopeer2-server.service
 * tibit-netopeer2-server.service - Tibit Communications, Inc. Netopeer2 Server
   Loaded: loaded (/lib/systemd/system/tibit-netopeer2-server.service; enabled; vendor
preset: enabled)
   Active: active (running) since Thu 2020-05-28 09:16:06 EDT; 2min 56s ago
  Process: 15910 ExecStart=/opt/tibit/netconf/bin/netopeer2-server -v 1 (code=exited,
status=0/SUCCESS)
 Main PID: 15912 (netopeer2-serve)
    Tasks: 7 (limit: 4915)
   CGroup: /system.slice/tibit-netopeer2-server.service
           /opt/tibit/netconf/bin/netopeer2-server -v 1
```

Starting the Server

Note: starting and restarting the service requires root level privileges.

Use the ‘systemctl start’ command to start the MCMS Netconf Server.

```
sudo systemctl start tibit-netconf.service
```

Use the ‘systemctl restart’ command to restart the MCMS Netconf Server. This is similar to running ‘systemctl stop’ followed by ‘systemctl start’.

```
sudo systemctl restart tibit-netconf.service
```

Stopping the Server

Note: stopping the service requires root level privileges.

Use the ‘systemctl stop’ command to shutdown the MCMS Netconf Server.

```
sudo systemctl stop tibit-netconf.service
```

Troubleshooting

Log Files

The MCMS Netconf Server generates Syslog messages that can be used for diagnosing and troubleshooting. The log files are described in the table below.

Log File	Description
/var/log/syslog	Diagnostic logging for the Netopeer2 Server.
/var/log/tibit/netconf.log	Diagnostic logging for the MCMS Netconf Server Database Connector.

Database Connection Statistics

The MCMS Netconf server captures MongoDB connection statistical information. Provided by the *“tibit-netconf”* yang model, the database connection statistics report success/fail counters, latency, MongoDB server topology and status, and various other related information.

Yang Path	Description
/netconf-state/database/replica-set	Replica set enabled status and associated replica set name.
/netconf-state/database/servers	Reports all known MongoDB server instances and server info.
/netconf-state/statistics/db-commands	Currently utilized MongoDB commands and their

	associated latency and success rate.
/netconf-state/statistics/db-connections	Reports MongoDB connection pool statistics.

Netopeer2 Logging

The Netopeer2 Server logs to /var/log/syslog as the ‘netopeer2-server’ service.

```
May 28 10:59:28 ubuntu systemd[1]: Starting Tibit Communications, Inc. Netopeer2 Server...
May 28 10:59:28 ubuntu systemd[1]: Started Tibit Communications, Inc. Netopeer2 Server...
```

Database Connector Logging

The MCMS Netconf Server Database Connector logs to the /var/log/tibit/netconf.log file as well as the SYSLOG-NETCONF collection in MongoDB.

```
2020-07-07 12:00:26.147 INFO     Initializing.
2020-07-07 12:00:26.147 INFO     Subscribing to Sysrepo.
2020-07-07 12:00:26.161 INFO     Initializing /tibit-pon-controller-db:controller
2020-07-07 12:00:26.162 INFO     Initializing /tibit-pon-controller-db:controller-alarm-profile
2020-07-07 12:00:26.162 INFO     Initializing /tibit-pon-controller-db:olt
2020-07-07 12:00:26.162 INFO     Initializing /tibit-pon-controller-db:olt-alarm-profile
2020-07-07 12:00:26.163 INFO     Initializing /tibit-pon-controller-db:onu
2020-07-07 12:00:26.163 INFO     Initializing /tibit-pon-controller-db:onu-alarm-profile
2020-07-07 12:00:26.163 INFO     Initializing /tibit-pon-controller-db:sla-profile
2020-07-07 12:00:26.164 INFO     Initializing /tibit-pon-controller-db:switch
2020-07-07 12:00:26.164 INFO     Subscribing to /tibit-pon-controller-db:controller//*
2020-07-07 12:00:26.164 INFO     Subscribing to /tibit-pon-controller-db:controller-alarm-
profile//*
2020-07-07 12:00:26.165 INFO     Subscribing to /tibit-pon-controller-db:olt//*
2020-07-07 12:00:26.165 INFO     Subscribing to /tibit-pon-controller-db:olt-alarm-profile//*
2020-07-07 12:00:26.165 INFO     Subscribing to /tibit-pon-controller-db:onu//*
2020-07-07 12:00:26.165 INFO     Subscribing to /tibit-pon-controller-db:onu-alarm-profile//*
2020-07-07 12:00:26.166 INFO     Subscribing to /tibit-pon-controller-db:sla-profile//*
2020-07-07 12:00:26.166 INFO     Subscribing to /tibit-pon-controller-db:switch//*
2020-07-07 12:00:26.166 INFO     Subscribing to /tibit-pon-controller-db:controller-state
2020-07-07 12:00:26.166 INFO     Subscribing to /tibit-pon-controller-db:controller-auth-state
2020-07-07 12:00:26.167 INFO     Subscribing to /tibit-pon-controller-db:controller-stats
2020-07-07 12:00:26.167 INFO     Subscribing to /tibit-pon-controller-db:controller-log
2020-07-07 12:00:26.167 INFO     Subscribing to /tibit-pon-controller-db:cpe-state
2020-07-07 12:00:26.167 INFO     Subscribing to /tibit-pon-controller-db:olt-state
2020-07-07 12:00:26.168 INFO     Subscribing to /tibit-pon-controller-db:olt-stats
2020-07-07 12:00:26.168 INFO     Subscribing to /tibit-pon-controller-db:olt-log
2020-07-07 12:00:26.168 INFO     Subscribing to /tibit-pon-controller-db:onu-state
2020-07-07 12:00:26.168 INFO     Subscribing to /tibit-pon-controller-db:onu-stats
2020-07-07 12:00:26.169 INFO     Subscribing to /tibit-pon-controller-db:onu-log
2020-07-07 12:00:26.169 INFO     Subscribing to /tibit-netconf:netconf-state
```

```
2020-07-07 12:00:26.169 INFO     Subscribing to rpc /tibit-pon-controller-db:controller-set-
status
2020-07-07 12:00:26.169 INFO     Subscribing to rpc /tibit-pon-controller-db:olt-allow-onu-
registration
2020-07-07 12:00:26.170 INFO     Subscribing to rpc /tibit-pon-controller-db:olt-disable-onu
2020-07-07 12:00:26.170 INFO     Subscribing to rpc /tibit-pon-controller-db:olt-reset
2020-07-07 12:00:26.170 INFO     Subscribing to rpc /tibit-pon-controller-db:onu-reset
2020-07-07 12:00:26.170 INFO     Connecting to database mongodb://127.0.0.1:27017
2020-07-07 12:00:26.171 INFO       ip address 127.0.0.1
2020-07-07 12:00:26.171 INFO       tcp port   27017
2020-07-07 12:00:26.171 INFO       database   tibit_pon_controller
2020-07-07 12:00:26.171 INFO       auth       False
2020-07-07 12:00:26.171 INFO       auth db    tibit_users
2020-07-07 12:00:26.171 INFO       username
2020-07-07 12:00:26.172 INFO       password
2020-07-07 12:00:26.172 INFO       tls        False
2020-07-07 12:00:26.172 INFO       ca cert    /etc/tibit/ca.pem
2020-07-07 12:00:26.180 INFO     Import running configuration from database...
...
2020-07-07 12:00:27.074 INFO     Import running configuration from database... done.
2020-07-07 12:00:27.074 INFO     Tibit NetCONF Database Connector is ready.
```

System logs stored in SYSLOG-NETCONF can be viewed through the netconf-log container in the tibit-netconf YANG model. The log table will contain up to 1,000 of the most recent system logs. To retrieve system logs, perform a get request using the following XML:

```
<get>
  <filter type="subtree">
    <tibitnc:netconf-log xmlns:tibitnc="urn:com:tibitcom:ns:yang:netconf"/>
  </filter>
</get>
```

The logging configuration can be configured through the tibit-netconf YANG model.
To modify the logging config, perform an RPC action request with the following XML, replacing *level* and *size* with the desired values:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="">
  <edit-config>
    <target>
      <running/>
    </target>
    <config>
      <netconf xmlns="urn:com:tibitcom:ns:yang:netconf">
        <logging>
          <console>level</console>
          <file>level</file>
          <syslog>level</syslog>
```

```
          <database>level</database>
          <maximum-log-size>size</maximum-log-size>
        </logging>
      </netconf>
    </config>
  </edit-config>
</rpc>
```

The logging configuration may also be modified via the sysrepocfg tool with the following XML, replacing *level* and *size* with the desired values:

```
$ cat logging-config.xml
  <netconf xmlns="urn:com:tibitcom:ns:yang:netconf">
    <logging>
      <console>level</console>
      <file>level</file>
      <syslog>level</syslog>
      <database>level</database>
      <maximum-log-size>size</maximum-log-size>
    </logging>
  </netconf>

# Apply configuration to the running datastore.
sudo /opt/tibit/netconf/bin/sysrepocfg --import=logging-config.xml --module=netconf --
format=xml --datastore=running
```

Alternatively, the logging configuration can be configured directly via the NETCONF-CFG collection in MongoDB. For more information on logging configuration parameters, see NETCONF-CFG.

User Activity Logging

The MCMS Netconf Server logs activity are logged in the Netconf Server logs. User activity logs include the username and IP address of the client accessing the Netconf Server.

User Authentication and Authorization

Logs are generated for user authentication and authorization events, including login, logout, authentication failures, and NACM authorization failures.

```
2022-04-07 10:35:35.531 INFO     NC Server: NETCONF session start for user tibit@10.2.10.106
2022-04-07 10:35:43.851 INFO     NC Server: NETCONF session end for user tibit@10.2.10.106,
reason 'closed'
2022-04-07 10:36:05.913 INFO     NC Server: SSH user authentication failed for user
tibit@10.2.10.106.
2022-04-07 10:42:22.461 INFO     NC Server: NETCONF operation <edit-config> access denied for
user tibit@10.2.10.106, NACM authorization failed
```

Configuration Changes

Logs are generated for configuration changes made through the Netconf interface. Each log message includes the username and IP address of the client, the configuration parameter being modified, new and previous values, and the Netconf operation with response code.

```
2022-04-07 10:59:49.213 INFO     NC tibit@10.2.10.106 modify /tibit-pon-controller-db:onu/
onu[name='BFWS00123193']/onu/address = Petaluma, CA, prev = Other Address
2022-04-07 10:45:09.928 INFO     NC tibit@10.2.10.106 <edit-config> succeeded with response 'ok'
```

State, Statistics, and Device Logs

Logs are generated when retrieving device state information through the Netconf interface. Each log message includes the username and IP address of the client, the key or identity of the record being retrieved, and the Netconf operation with response code. The detailed contents of the record are not logged as part of this message.

```
2022-04-07 10:47:54.464 INFO     NC tibit@10.2.10.106 <get> /tibit-pon-controller-db:onu-state/
onu[name='ALPHe30cadcf'] succeeded with response 'ok'
```

Limitations

NETCONF Server

MCMS Netconf Server has the following limitations:

- The following NETCONF datastores are not supported *for YANG modules*: startup.
- The following NETCONF capabilities are not supported: :confirmed-commit, :startup.
- NETCONF Notifications are not supported.

BBF YANG Models

Support for BBF YANG support has the following limitations:

- See section Supported Modules for a complete list of supported BBF YANG models.
- ONU Service configurations that classify or edit DEI and DSCP are not supported.
- The following feature are cannot be managed using BBF YANG*:
    - 802.1x Authentication
    - Downstream SLAs
    - Firmware Upgrade
    - Logs
- Modifying the vANI's expected-serial-number is not supported after the interface has been created. The work-around is to delete the vANI with the old serial number and create a new vANI interface with the new serial number.
- BBF YANG does not support configuration for EPON DPoE ONUs.

**Note: features not manageable using BBF YANG can be configured using Tibit YANG.*