Cisco Routed PON Netconf Server

User Guide

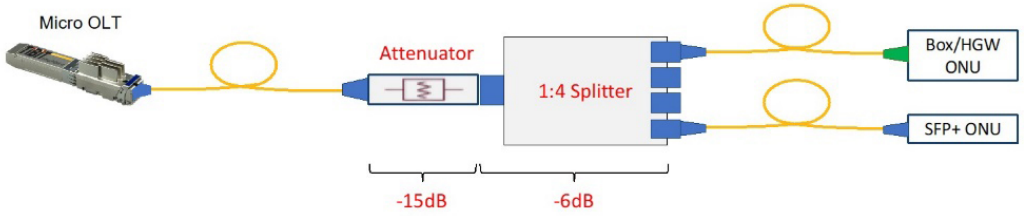
CAUTION

Refer to the 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. Refer to the example:



523682

Table of Contents

References

ID	Document Description
ConfD Basic	Cisco Tail-f ConfD Basic, <https://www.tail-f.com/confd-basic/>.
Cisco Routed PON Installation Guide	Cisco Routed PON Installation Guide
Cisco Routed PON Manager	Cisco Routed PON Manager User Guide
Cisco Routed PON REST API	Cisco Routed PON REST API Developer Guide
Cisco Routed PON Netconf Server User Guide	Cisco Routed PON Netconf Server User Guide
MongoDB	MongoDB 4.4 Manual, <https://docs.mongodb.com/v4.4/>.

Document Purpose

This document is intended for users deploying the ConfD Netconf Server and provides an overview of the architecture, installation instructions, and configuration and server administration information.

Although the open-source MongoDB is shown as part of the Cisco Routed PON architecture, MongoDB is not provided as part of the Cisco Routed PON Netconf Server package. MongoDB is a dependency of the Netconf Server. Installation, maintenance, and operation of MongoDB is considered out of scope. See [Routed PON Installation Guide] and [MongoDB] for more information on installing MongoDB.

Introduction

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

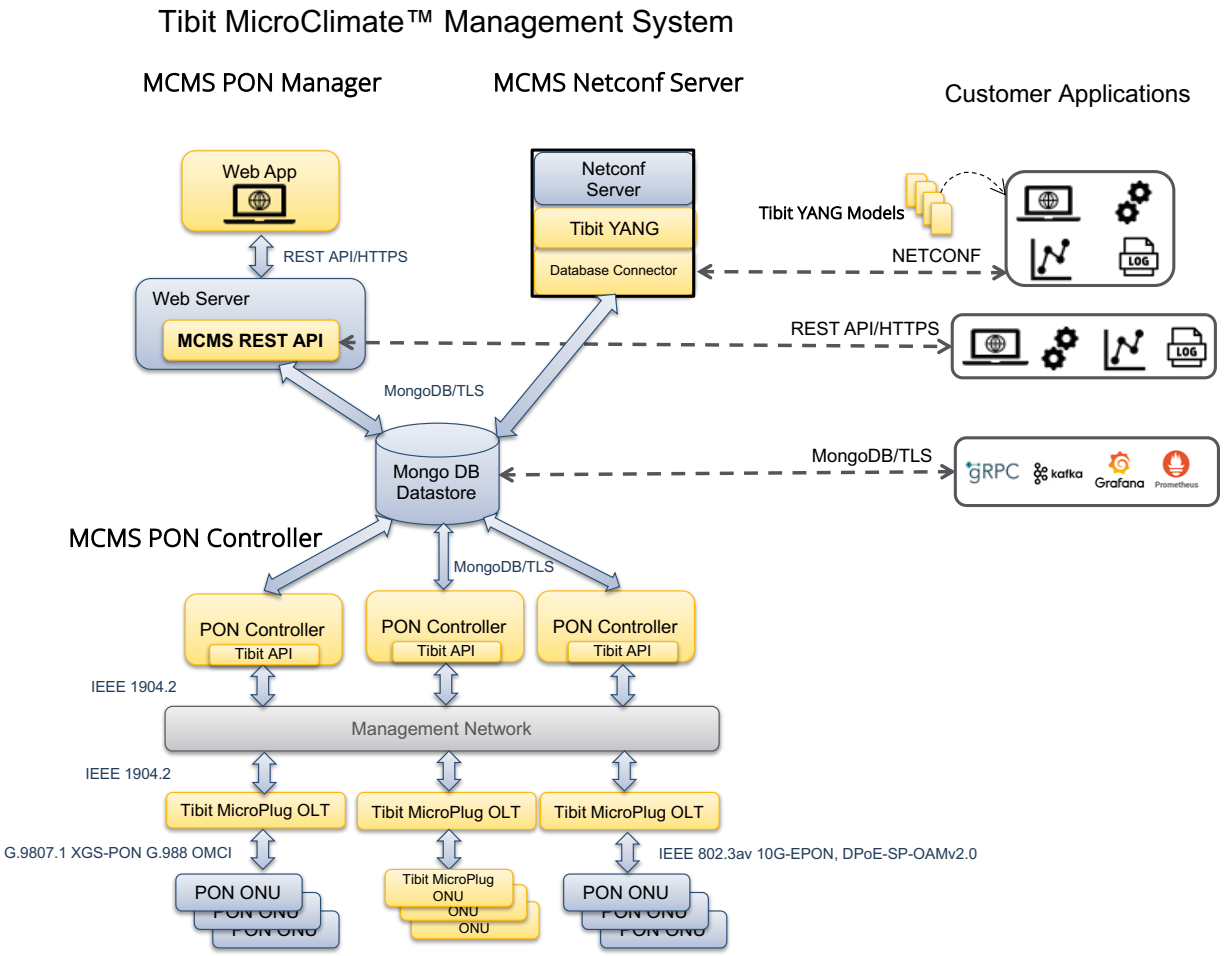


Figure 1 - Cisco Routed PON Solution Architecture

NETCONF/YANG Interface

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

The Cisco Routed PON 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

Other Software Components

This User Guide focuses on the Cisco Routed PON NETCONF/YANG interface. See [Cisco Routed PON Manager] for more information regarding the Cisco Routed PON Manager graphical user interface, REST API, and MongoDB datastore.

Architecture Overview

The ConfD Cisco Routed PON Netconf Server is built on the Cisco Tail-f ConfD Basic management framework. The Cisco Routed PON Netconf Database Connector, in conjunction with ConfD, make up the components of the Netconf Server. Together these components provide the NETCONF/YANG interface for managing a Cisco Routed PON OLT PON network. The software architecture is shown in the figure below.

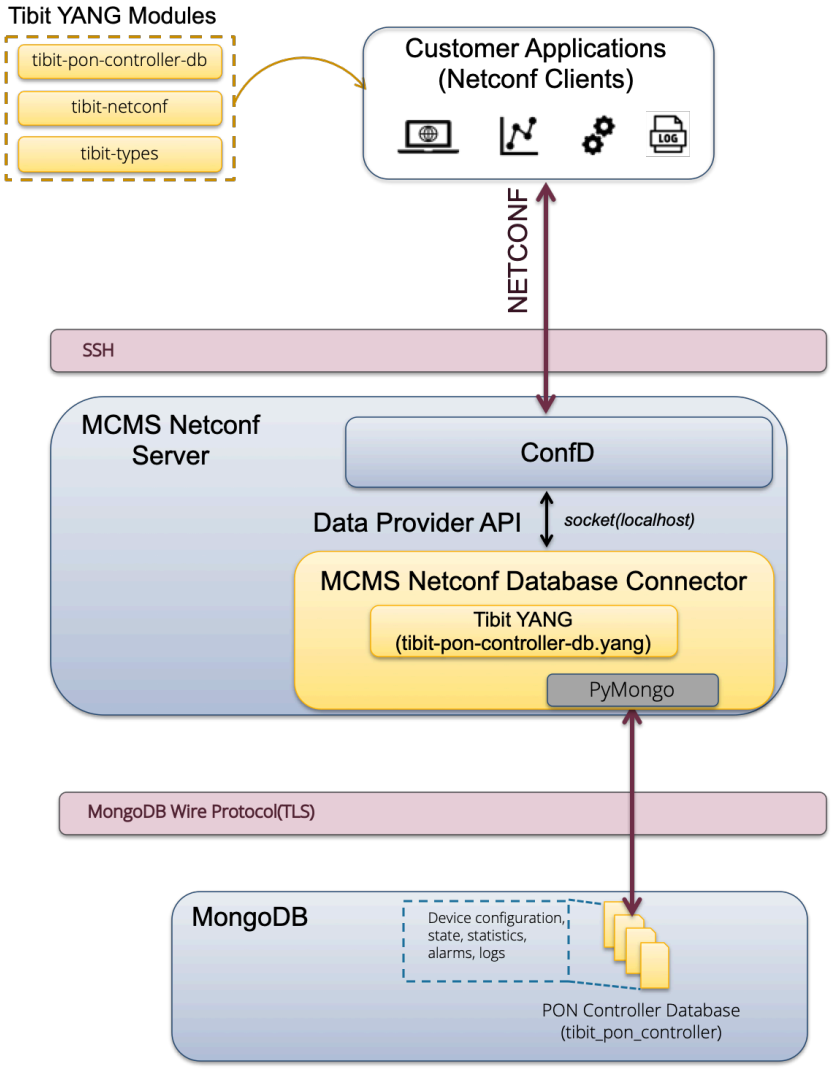


Figure 2 - Cisco Routed PON Netconf Server Architecture

ConfD Server

The ConfD Server implements the server side of the RFC 6241 NETCONF Protocol with support for Secure Shell (SSH) transport accessible on TCP port 830 by default. ConfD also provides tools and functions for managing YANG datastores, including management for the candidate and running configuration. ConfD is an application that runs as a Linux service separate from the Database Connector. The service terminates the RPC requests from

NETCONF client applications and calls the ConfD Data Provider API to service requests for specific YANG models.

Database Connector

The Cisco Routed PON Netconf Database Connector implements the application side of the ConfD Data Provider API, translating between documents in Mongo Database and the YANG data from ConfD. The Database Connector is a Linux application that runs as a service separate from the ConfD service. Northbound, the Database Connector communicates with the ConfD service through the Data Provider API over a local socket. Southbound, the Database Connector communicates with the MongoDB server through the PyMongo API over the MongoDB Wire Protocol (TLS) using a TCP socket.

The ConfD database (CDB) is not used for storing configuration data. Instead, configuration changes made through the NETCONF/YANG interface are mapped directly to MongoDB data without the need for a separate copy of the data in ConfD.

Cisco YANG Models

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

MongoDB

The Mongo database provides the datastore for the Cisco Routed PON Solution. See [Cisco Routed 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 Tibit management solutions.

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

Installation

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

Requirements and Dependencies

Supported Operating Systems

Note: The Cisco Routed PON ConfD Netconf Server is not supported on Ubuntu 18.04.

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

Cisco Routed PON Software and Firmware Dependencies

Tibit Package	Version
Cisco Routed PON Controller	R4.0.0
Cisco Routed PON OLT Firmware	R4.0.0
Cisco Routed PON ONU Firmware	R4.0.0

MongoDB Dependencies

MongoDB is expected to be installed and running.

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

Database Package	Ubuntu 20.04
MongoDB	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.

NOTE: You may be informed that certain Linux, SSH, and XML-related settings need to be updated during the update process.

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

TCP/IP Ports and Network Services

The Cisco Routed PON 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 Cisco Routed PON 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.

ConfD's Data Provider API communicates with the Database Connector application through a local socket, where the socket listens on localhost only. The Netconf Server requires access to the following localhost (127.0.0.1) network services:

Network Service	TCP Port	Description
ConfD API	4565	Local TCP port for communication used for the ConfD API between the Database Connector/ConfD tools and the ConfD service.

Package Contents

The Cisco Routed PON 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 Cisco Routed PON Netconf Server in a docker container.
examples/	Directory containing example scripts for configuring and managing the Cisco Routed PON OLT network using NETCONF/YANG.
examples/tibit_yang	Directory containing examples for using Cisco YANG models for configuring OLT and ONU devices.
examples/nacm	Directory containing examples for configuring the Network Configuration Access Control Model (NACM) for the Cisco Routed PON Netconf Server.
INSTALL.txt	Instructions for installing and quick start guide for running the Cisco Routed PON Netconf Server.
LICENSE.txt	Text file that lists the licensing information for third party software used by the Cisco Routed PON Netconf Server.
tibit-netconf-confd_R4.1.0_amd64.deb	Debian installer package containing the Cisco Routed PON Netconf Server application binaries, default configuration, and supporting files.
yang/	Directory containing the Cisco 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 [Cisco Routed PON Installation Guide].

Installation

This section describes the steps to install Cisco Routed PON 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 Cisco Routed PON Netconf Server.
/opt/tibit/netconf/	Cisco Routed PON Netconf Server Database Connector binaries, examples, YANG modules, and other supporting files.
/opt/tibit/confd/	Cisco Routed PON Netconf Server ConfD binaries and 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-ConfD.zip
   cd R4.0.0-Netconf-ConfD
   ```

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

   ```
   sudo apt-get install ./tibit-netconf-confd_R4.0.0_amd64.deb
   ```

3. Configure the Cisco Routed PON 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 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. Check the status of ConfD service:

```
systemctl status cisco-netconf.service
systemctl status cisco-confd.service
systemctl restart mongod.service
```

Validation Steps

From a shell, run the '/opt/tibit/netconf/examples/nc_get_version.sh' script and verify the Cisco Routed PON 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_version.sh
<data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <netconf-state xmlns="urn:com:tibitcom:ns:yang:netconf">
    <version>
      <build-date>2022-11-19T03:44:18-08:00</build-date>
      <build-sha>02a4912</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>
```

To verify the Cisco Routed PON Netconf Server service is running and to monitor its status, run the following commands from a shell:

```
sudo systemctl status tibit-confd

sudo systemctl status tibit-netconf
```

Uninstall

The uninstall process deletes the Cisco Routed PON 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 Cisco Netconf service. Note: uninstall requires root level privileges. This command will completely remove the /etc/tibit/netconf, /opt/tibit/confd, and /opt/tibit/netconf directories.

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

2. If the following warning occurs during 'apt-get purge', use 'rm -rf <dir>' to completely remove the /opt/tibit/netconf and /opt/tibit/confd directories.

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

   rm -rf /opt/tibit/netconf

   rm -rf /opt/tibit/confd
   ```

NETCONF Features

This section describes the RFC 6241 and RFC 8526 NETCONF protocol features and capabilities supported by the Cisco Routed PON ConfD Netconf Server.

Protocol Operations

The following protocol operations are supported:

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

Datastores

The following datastores are supported:

- running

The following datastores are ***not*** supported:

- candidate
- startup

Capabilities

The following NETCONF capabilities are supported:

- :writable-running
- :validate
- :xpath
- :url

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

- :candidate
- :confirmed-commit
- :rollback-on-error
- :startup

Notifications

The following notification features are supported:

- <create-subscription>
- <notification>

Cisco YANG Models

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

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

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.
RA system-name	System name.
message	Log text.
severity	Log severity.
timestamp	Timestamp when the message was logged.
/tibit-netconf:netconf-clear	
netconf-clear	RPC: 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. 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	Cisco Routed PON Authenticator service state
:controller-engine-state/	CNTL-ENGINE-STATE	Cisco Routed PON 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

tibit-pon-controller-db YANG Container	MongoDB Collection	Description
: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

Server Administration

This section provides information on how to configure and manage the Cisco Routed PON ConfD Netconf Server.

Configuration

The Cisco Routed PON Netconf Server configuration consists of three parts. The 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 Connector Configuration for more information on configuring the database connector.

The ConfD Netconf server and protocol options are configured in the /etc/tibit/netconf/confd /confd.conf configuration file. Edit this file to modify ConfD server SSH listen IP address, TCP port, protocol options, logging, and other ConfD configuration parameters. See ConfD Server Configuration for more information on configuring the ConfD server.

The user access controls to data through the Netconf Server is managed through configuration of standard YANG models defined by IETF listed in the table below. Configurations from these YANG models are loaded from the persistent 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).

Server Configuration Tools

This section describes several tools packaged with the Netconf Server that can be used to configure and manage the server.

ConfD Configuration Tool (confd_load)

The ConfD 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. A Netconf Server connection and user authentication is not required, nor is NACM applied when using confd_load. Note: root level access is required to use this tool.

Path

```
/opt/tibit/confd/bin/confd_load
```

Usage

```
$ confd_load -h
A utility that saves and loads the running configuration, usage:
    confd_load [options] [filename]
    confd_load -l [options] [filename...]
 Valid options are (for further details, see the manpage):
    -d            debug flag
    -l            load config into ConfD [default is to save]
    -F  <format>  format can be one of:
                    x  XML [default]
                    p  Pretty XML
                    o  JSON
                    j  J-style CLI
                    c  C-style CLI
                    i  I-style CLI
                    t  C-style using turbo parser. Only applicable
                       for load config into ConfD
    -W            include default values
    -S            include default values as comments
    -m            when loading config, merge [default is delete and replace]
    -r            when loading config, replace [default is delete and
replace]
    -H            hide all hidden nodes [default depends on transaction]
    -U            unhide all hidden nodes [default depends on transaction]
    -a            when loading 'c' or 'i' config, commit after each line
    -e            when loading, do not abort on errors
    -p  <path>    when saving the path to save
                  when loading this path will first be deleted
    -N            when saving, do Not include parents to the path
    -P  <xpath>   when saving apply this xpath filter
    -D            do maapi_delete_all(MAAPI_DEL_ALL) before loading
    -o            when saving include operational data
                  when loading ignore operational data
```

```
    -A            when saving, only include nodes for which the user
                  has read_write access
    -u  <user>    use this user
    -g  <group>   use this group (may be repeated)
    -c  <context> use this context [default is 'system']
    -i            attach to init session instead of starting new session
    -s            start transaction towards startup [default is running]
    -O            when saving, include only operational data, not config
                  when loading, start operational transaction (load oper
                  data via a transaction instead of via CDB)
    -b            when saving, include only data stored in CDB
    -t            measure how long the requested command takes
    -R            generate CDB operational subscription notifications
    -w            enable "delayed when" mode of transaction
```

Examples

Export All Configuration from the Running Datastore

```
$ sudo /opt/tibit/confd/bin/confd_load -F p
<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/confd/bin/confd_load -l -r nacm-config.xml
```

Netconf Client Command Line Interface Tool (netconf-console)

The netconf-console tool is a command line NETCONF client tool. This tool uses the standard NETCONF protocol for managing the server. Unlike the confd_load tool, NETCONF user authentication is required and NACM rules are enforced when managing the server using netconf-console tool. If the user's password is omitted, netconf-console attempts to authenticate with the user's SSH key.

Path

`/opt/tibit/confd/bin/netconf-console`

Usage

```
$ netconf-console -h
Usage: netconf-console.py [-h | --help] [options] [cmdoptions | <filename> | -]

    Where <filename> is a file containing a NETCONF XML command session.
    If <filename> is not given, one of the Command Options must be given.
    Filename '-' means standard input.


Options:
  -h, --help            show this help message and exit
  -v VERSION, --version=VERSION
                        force NETCONF version 1.0 or 1.1
  -d, --debug
  -u USERNAME, --user=USERNAME
                        username
  -p PASSWORD, --password=PASSWORD
                        password
  --proto=PROTO         Which transport protocol to use, one of ssh or tcp
  --host=HOST           NETCONF server hostname
  --port=PORT           NETCONF server port
  --listen              Start NETCONF Call Home listener
  --listen-address=LISTEN_ADDRESS
                        Listen to this address
  --listen-port=LISTEN_PORT
                        Listen to this port
  --iter=ITER           Sends the same request ITER times.  Useful only in
                        test scripts
  -i, --interactive

  Style Options:
    raw:    print bytes received on the wire, framing is removed
```

```
  plain:  skip the <hello> reply, print one reply as raw, skip the rest
  pretty: skip the <hello> reply, pretty-print one reply, skip the rest
  all:    skip the <hello> reply, pretty-print the rest of the replies
  noaaa:  as pretty, but remove Tail-f AAA and IETF NACM from the output

  -s STYLE, --outputStyle=STYLE
                      Display output in: raw, plain, pretty, all, or noaaa
                      format

SSH Options:
  --privKeyType=PRIVKEYTYPE
                      type of private key, rsa or dss
  --privKeyFile=PRIVKEYFILE
                      file which contains the private key

TCP Options:
  -g GROUPS, --groups=GROUPS
                      Set group membership for user - comma separated string
  -G SUPGROUPS, --sup-groups=SUPGROUPS
                      Set supplementary UNIX group ids for user - comma
                      separated string of integers

NETCONF Command Options:
  --hello             Connect to the server and print its capabilities
  --get               Takes an optional -x argument for XPath filtering or
                      --subtree-filter for subtree filtering.
  --get-config        Takes an optional --db argument, default is 'running',
                      and an optional -x argument for XPath filtering or
                      --subtree-filter for subtree filtering.
  --get-data          Takes an optional --db argument, default is 'running',
                      an optional --subtree-filter argument, an optional -x
                      argument for XPath filtering, an optional --only-
                      config-false argument, an optional --only-config-true
                      argument (these arguments can not be used together),
                      an optional --max-depth argument, an optional
                      --origin-filter argument oran optional --negated-
                      origin-filter argument.
  --max-depth=MAX_DEPTH
                      Can be used to limit the depth of the --get-data
                      command.
  --config-filter=CONFIG_FILTER
                      Can be set to 'true' or 'false'. If not set at all,
                      both operational and config data will be returned from
                      --get-data. If set to true, only config data will be
                      returned. If set to false, only operational data will
                      be returned.
  --subtree-filter=SUBTREE_FILTER
                      Takes a filename (or '-' for standard input) as
                      argument. Can be used with --get* operations,
                      --create-subscription and --establish-subscription.
  --with-origin       If present, the 'origin' annotation for the nodes will
                      be returned.
  --origin-filter=ORIGIN_FILTER
                      It can be used with --get-data. If one or more origin
                      filter is set, only nodes derived from given origin
```

```
                    will be returned.
--negated-origin-filter=NEGATED_ORIGIN_FILTER
                    It can be used with --get-data. If one or more origin
                    filter is set, only nodes that are not derived from
                    given origin will be returned.
--db=DB             Database for commands that operate on a database.
--with-defaults=WDEFAULTS
                    One of 'explicit', 'trim', 'report-all' or 'report-
                    all-tagged'.  Use with --get, --get-config, --get-data
                    or --copy-config.
--with-inactive     Send with-inactive parameter.  Use with --get, --get-
                    config, --get-data, --copy-config, --edit-config, or
                    --edit-data.
-x, --xpath         XPath filter to be used with --get, --get-config,
                    --get-data, --create-subscription and --establish-
                    subscription
--kill-session=KILL_SESSION
                    Takes a session-id as argument.
--discard-changes
--commit
--validate          Takes an optional --db argument, default is 'running'.
--copy-running-to-startup
--copy-config=COPY  Takes a filename (or '-' for standard input) as
                    argument. The contents of the file is data for a
                    single NETCONF copy-config operation (put into the
                    <config> XML element). Takes an optional --db
                    argument, default is 'running'.
--edit-config=EDIT  Takes a filename (or '-' for standard input) as
                    argument. The contents of the file is data for a
                    single NETCONF edit-config operation (put into the
                    <config> XML element). Takes an optional --db
                    argument, default is 'running'.
--edit-data=EDIT_DATA
                    Takes a filename (or '-' for standard input) as
                    argument. The contents of the file is data for a
                    single NETCONF edit-data operation (put into the
                    <config> XML element). Takes an optional --db
                    argument, default is 'running'.
--get-schema=GET_SCHEMA
                    Takes an identifier (typically YANG module name) as
                    parameter
--create-subscription=CREATE_SUBSCRIPTION
                    Takes a stream name as parameter, and an optional -x
                    for XPath filtering or --subtree-filter for subtree
                    filtering.
--establish-subscription=ESTABLISH_SUBSCRIPTION
                    Takes a stream name as parameter, and an optional -x
                    for XPath filtering or --subtree-filter for subtree
                    filtering.
--kill-subscription=KILL_SUBSCRIPTION
                    Takes a subscription id as parameter
--replay-start-time=START_TIME
                    Takes a timestamp in YYYY-MM-DDTHH:MM:SS format, for
                    e.g. 2017-01-10T12:05:21. Assumes UTC.  Used with
                    --establish-subscription
```

```
    --rpc=RPC           Takes a filename (or '-' for standard input) as
                        argument. The contents of the file is a single NETCONF
                        rpc operation (w/o the surrounding <rpc>).
    --start-time=START_TIME
                        Takes a timestamp in YYYY-MM-DDTHH:MM:SS format, for
                        e.g. 2017-01-10T12:05:21. Assumes UTC.  Used with
                        --create-subscription
    --stop-time=STOP_TIME
                        Takes a timestamp in YYYY-MM-DDTHH:MM:SS format, for
                        e.g. 2017-01-10T12:05:21. Assumes UTC. Used with
                        --create-subscription or --establish-subscription.The
                        session will be closed after stop-time hasbeen reached
    --rpc-attribute=RPC_ATTRIBUTE
                        XML attributes to send in the rpc element
```

Examples

Get Netconf Server Version

```
$ netconf-console --host localhost --user <USER> --get -x /netconf-state/version
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="1">
  <data>
    <netconf-state xmlns="urn:com:tibitcom:ns:yang:netconf">
      <version>
        <build-date>2022-11-19T03:44:18-08:00</build-date>
        <build-sha>02a4912</build-sha>
        <version>R4.0.0</version>
      </version>
    </netconf-state>
  </data>
</rpc-reply>
```

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 Cisco Routed PON 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 Cisco Routed PON 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	This field is deprecated.
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 logging configuration from the NETCONF-CFG collection.

```
{
    "Logging": {
        "Filename"  : "/var/log/tibit/netconf.log",
        "FileCount" : 3,
        "FileSize"  : 1024000,
        "Netconf"   : {
```

```
            "Console" : "INFO",
            "File"    : "INFO",
            "Syslog"  : "INFO"
        }
    },
}
```

The 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 Cisco Routed PON Netconf Server Database Connector 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 Cisco Routed PON Netconf Server Database Connector only.
Netconf.Syslog	INFO	Logging level for messages logged to syslog: CRITICAL, ERROR, WARNING, INFO, DEBUG, DISABLE. This logging level applies to the Cisco Routed PON Netconf Server Database Connector only.

Database Connector MongoDB Collections

This section describes the MongoDB configuration (CFG), state (STATE), and log (SYSLOG) collections used for managing the Cisco Routed PON 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 Cisco Routed PON 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 Cisco Routed PON 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 Cisco Routed PON 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 Cisco Routed PON 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 Cisco Routed PON 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(ms), 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(ms), min latency(ms), max latency(ms), 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	Cisco Routed PON NETCONF Database Connector version.
Build Date	Cisco Routed PON NETCONF Database Connector build date.
Build SHA	Cisco Routed PON NETCONF Database Connector build simple hashing algorithm.
Build OS	Cisco Routed PON NETCONF Database Connector OS.
DB Version	Cisco Routed PON NETCONF Database Connector database version.
Txn ID	Cisco Routed PON 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.

ConfD Server Configuration

Network Configuration

Network protocol, transport, and other NETCONF server parameters are configured through the ConfD configuration file: /etc/tibit/netconf/confd/confd.conf. The ConfD configuration file uses an XML format and can be modified with any text editor. By default, root access is required to modify confd.conf. This section uses paths (.e.g., /confdConfig/netconf/transport/ssh/ip) to identify the location of specific parameters in the ConfD XML configuration file.

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 modifying the listen IP address and TCP port from default values.

To modify the listen IP address and TCP port:

1) Edit the /confdConfig/netconf/transport/ssh 'ip' and 'port' parameters, replacing *ip address* and *port number* shown below with the desired values.

```
<netconf>
  <enabled>true</enabled>

  <transport>
    <ssh>
      <enabled>true</enabled>
      <ip>ip address</ip>
      <port>port number</port>
    </ssh>
  </transport>
```

2) Restart the Cisco Routed PON Netconf Server

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

SSH Algorithms

The ConfD Netconf Server supports configuration of the cryptography algorithms enabled for the SSH server. The table below lists the SSH algorithms supported by the Netconf Server.

Type	ConfD Attribute	Description
Encryption	/confdConfig/ssh /algorithms /encryption	Supported encryption algorithms: aes128-gcm@openssh.com, chacha20-poly1305@openssh.com, aes128-ctr,

		aes192-ctr, aes256-ctr, aes128-cbc, aes256-cbc, 3des-cbc **Default:** aes128-gcm@openssh.com, chacha20-poly1305@openssh.com, aes128-ctr, aes192-ctr, aes256-ctr
Key Exchange	/confdConfig/ssh /algorithms/kex	Supported key exchange algorithms: ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp512, curve25519-sha256, diffie-hellman-group14-sha256, diffie-hellman-group-exchange-sha256, diffie-hellman-group-exchangesha1, diffie-hellman-group14-sha1. **Default:** curve25519-sha256, ecdh-sha2-nistp256, diffie-hellman-group14-sha256, diffie-hellman-group-exchange-sha256
MAC	/confdConfig/ssh /algorithms/mac	Supported MAC algorithms: hmac-md5, hmac-sha1, hmac-sha2-256, hmac-sha2-512, hmac-sha1-96, hmac-md5-96. **Default:** hmac-sha2-512, hmac-sha2-256, hmac-sha1
Server Host Key	/confdConfig/ssh /algorithms /serverHostKey	Supported server host key algorithms: ssh-dss, ssh-rsa, sshed25519, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521. **Default:** ssh-ed25519, ssh-rsa

To modify the SSH algorithms enabled on the ConfD server:

1) Modify the /confdConfig/ssh/algorithms parameters shown below with the desired values.
   Note: the parameters may not be present and may need to be added to the confd.conf file.

```
<ssh>
  <algorithms>
    <encryption>
      aes128-gcm@openssh.com,chacha20-poly1305@openssh.com,
      aes128-ctr,aes192-ctr,aes256-ctr
    </encryption>
    <kex>
      curve25519-sha256,ecdh-sha2-nistp256,diffie-hellman-group14-sha256,
      diffie-hellman-group-exchange-sha256
    </kex>
    <mac>hmac-sha2-512, hmac-sha2-256, hmac-sha1</mac>
```

```
    <serverHostKey>ssh-ed25519,ssh-rsa</serverHostKey>
  </algorithms>
 </ssh>
```

2) Restart the Cisco Routed PON Netconf Server

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

User Management

The ConfD Netconf Server uses the local Linux server's users and passwords through the Linux Pluggable Authentication Modules (PAM) service for 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.

Netconf Users are configured in conjunction with the Network Configuration Access Control Model (NACM) to provide secure access to data through the Netconf interface. Users must be assigned to a group in order to access data through the Netconf interface. Users are not assigned to groups by default. When a user is not assigned to a group, the user is allowed to establish the SSH connection, but is denied access to all data and blocked from executing all operations. See NACM for more information regarding the Netconf security model.

Public Key Authentication

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

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.

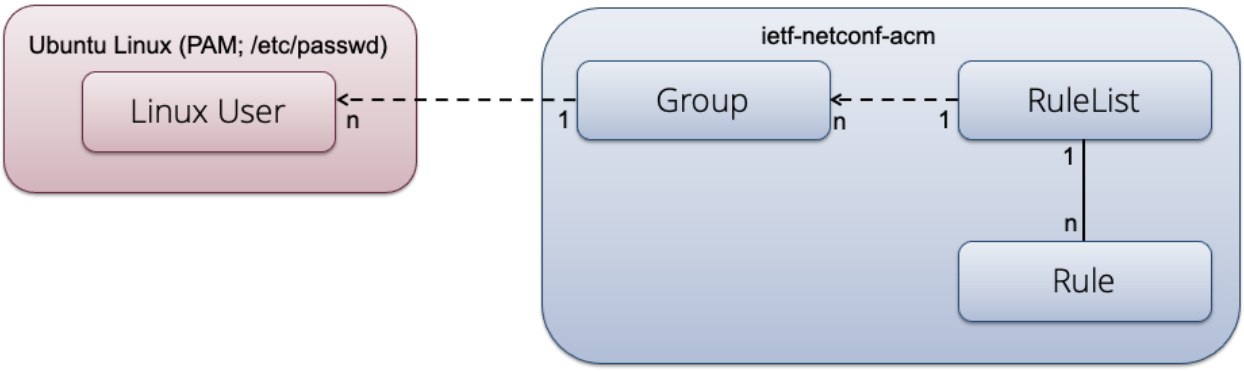
```
ssh <netconf server address>
```

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 Linux 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 exist as a Linux 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 confd_load 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/confd/bin/confd_load -l -r nacm-config.xml
```

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 confd_load 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 confd_load 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/confd/bin/confd_load -l -r nacm-config.xml
```

Create a NACM Group

By default, no NACM groups are configured for the Netconf Server. Use the confd_load 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/confd/bin/confd_load -l -r nacm-config.xml
```

Add a User to an Existing NACM Group

The Netconf Server examples/nacm directory contains helper scripts for adding and removing users from NACM groups. Use the nacm-add-user.sh script to add a user to an existing NACM group. Alternatively, the confd_load tool or NETCONF client application can be used to add a user to a NACM group.

```
cd /opt/tibit/netconf/examples/nacm
sudo ./nacm-add-user.sh netconf-user read-only


$ ./nacm-add-user.sh --help

./nacm-add-user.sh  <username> <group>

Configure NACM by adding username to the specified group.
```

Remove a User from a NACM Group

Use the nacm-rm-user.sh script to remove a user from a NACM group. Alternatively, the confd_load tool or NETCONF client application can be used to remove a user from the NACM group.

```
cd /opt/tibit/netconf/examples/nacm
sudo ./nacm-rm-user.sh netconf-user read-only


$ ./nacm-rm-user.sh --help

./nacm-rm-user.sh  <username> <group>

Configure NACM by removing username from the specified group.
```

Starting, Stopping, and Restarting the Software

The Cisco Routed PON 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 Cisco Routed PON Netconf Server processes are described in the table below.

Systemd Service	Process Name	Application Binary	Description
tibit-confd.service	confd.smp	/opt/tibit/confd/lib/erts /bin/confd.smp	Implements the NETCONF protocol according to RFC 6241.
tibit-netconf.service	tibit-netconf	/opt/tibit/netconf/bin/ tibit-netconf	Cisco Routed PON NETCONF Database Connector.

The tibit-confd service is configured as a systemd dependency of the tibit-netconf service. This systemd dependency configuration allows all Cisco Routed PON Netconf Server processes to be managed as a single application using systemd. The tibit-confd 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 Cisco Routed PON 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 2022-11-17 16:02:01 EST; 1 weeks 5 days ago
   Main PID: 2031883 (tibit-netconf)
      Tasks: 22 (limit: 18903)
     Memory: 27.9M
     CGroup: /system.slice/tibit-netconf.service
             ├─2031883 /opt/tibit/netconf/bin/tibit-netconf
             └─2031887 /opt/tibit/netconf/bin/tibit-netconf

           /opt/tibit/netconf/bin/tibit-netconf
           /opt/tibit/netconf/bin/tibit-netconf


$ systemctl status tibit-confd.service
 * tibit-confd.service - Tibit Communications, Inc. Conf-D Service
     Loaded: loaded (/lib/systemd/system/tibit-confd.service; enabled; vendor preset:
enabled)
```

```
     Active: active (running) since Thu 2022-11-17 16:02:01 EST; 1 weeks 5 days ago
   Main PID: 2031863 (confd.smp)
      Tasks: 19 (limit: 18903)
     Memory: 9.9M
     CGroup: /system.slice/tibit-confd.service
             ├─2031863 /opt/tibit/confd/lib/confd/erts/bin/confd.smp -S 1 -K false -
MHe true -- -root /opt/tibit/confd/lib/confd -progname confd ->
             └─2031867 erl_child_setup 1024
```

Starting the Server

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

Use the ‘systemctl start’ command to start the Cisco Routed PON Netconf Server.

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

Use the ‘systemctl restart’ command to restart the Cisco Routed PON 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 Cisco Routed PON Netconf Server.

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

Troubleshooting

The Cisco Routed PON 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/tibit/confd.log	General diagnostic logging for the ConfD Server.
/var/log/tibit/confd_audit.log	User authentication and access logging for the ConfD Server.
/var/log/tibit/confd_netconf.log	NETCONF protocol specific diagnostic logging for the ConfD Server.
/var/log/tibit/netconf.log	Diagnostic logging for the Cisco Routed PON Netconf Server Database Connector.

ConfD Logging

The ConfD Server logs to the /var/log/confd*.log files.

```
<INFO> 17-Nov-2022::16:02:00.980 - Starting to listen for Internal IPC on 127.0.0.1:4565
<INFO> 17-Nov-2022::16:02:01.148 - CDB load: processing file: /var/tibit/confd/cdb/aaa_init.xml
<INFO> 17-Nov-2022::16:02:01.228 - Starting to listen for NETCONF SSH on 0.0.0.0:830
<INFO> 17-Nov-2022::16:02:01.363 - ConfD started vsn: 7.8.2
```

Database Connector Logging

The Cisco Routed PON Netconf Server Database Connector logs to the /var/log/tibit/netconf.log file as well as the SYSLOG-NETCONF collection in MongoDB. See Cisco Routed PON Netconf Server for information on retrieving system logs and modifying the logging configuration.

```
2022-11-30 15:57:17.761 INFO     Tibit Netconf version    R4.0.0
2022-11-30 15:57:17.761 INFO                  build date 2022-11-19T03:44:18-08:00
2022-11-30 15:57:17.761 INFO                  build SHA  02a4912
2022-11-30 15:57:17.761 INFO                  build OS   Ubuntu20.04
2022-11-30 15:57:17.761 INFO                  DB Version R4.0.0
2022-11-30 15:57:17.761 INFO     Initializing.
2022-11-30 15:57:17.761 INFO     Conf-D debug log level changed to NONE (0)
2022-11-30 15:57:17.761 INFO     Connecting to database mongodb://10.2.10.222:27017
2022-11-30 15:57:17.761 INFO       host      10.2.10.222
2022-11-30 15:57:17.761 INFO       port      27017
2022-11-30 15:57:17.761 INFO       database  tibit_pon_controller_r31
2022-11-30 15:57:17.762 INFO       auth      False
2022-11-30 15:57:17.762 INFO       auth db   tibit_users
2022-11-30 15:57:17.762 INFO       username
2022-11-30 15:57:17.762 INFO       password
2022-11-30 15:57:17.762 INFO       tls       False
2022-11-30 15:57:17.762 INFO       ca cert   /etc/tibit/ca.pem
2022-11-30 15:57:17.762 INFO       srv       False
2022-11-30 15:57:17.762 INFO       replica set
2022-11-30 15:57:17.762 INFO         enable False
2022-11-30 15:57:17.762 INFO         name   rs0
2022-11-30 15:57:17.762 INFO         hosts  ['127.0.0.1:27017', '10.2.10.222:27017', 'localhost',
'127.0.0.1']
2022-11-30 15:57:17.762 INFO       compress True
```

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 confd_load 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/confd/bin/confd_load -l -r ./logging-config.xml
```

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 Cisco Routed PON 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.

```
2022-11-30 16:00:31.576 INFO     NC Server: tibit pam authentication succeeded via netconf from
10.2.10.66:57348 with ssh
2022-11-30 16:00:31.577 INFO     NC Server: tibit logged in via netconf from 10.2.10.66:57348
with ssh using pam authentication
2022-11-30 16:00:31.585 INFO     NC Server: tibit assigned to groups: read-only,netconf-readonly
2022-11-30 16:00:31.586 INFO     NC Server: tibit created new session via netconf from
10.2.10.66:57348 with ssh
2022-11-30 16:02:10.984 INFO     NC Server: tibit terminated session (reason: normal)
2022-11-30 16:00:31.555 INFO     NC Server: [withheld] local authentication failed via netconf
from 10.2.10.66:57348 with ssh: no such local user
```

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.

```
2022-11-30 16:18:09.619 INFO     NC tibit@10.2.10.66 modify /tibitcntlr:controller/
controller[name="52:54:00:1e:fa:d9"]/controller/address = Petaluma, CA
2022-11-30 16:18:09.619 INFO     NC tibit@10.2.10.66 <edit-confg> 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.

```
2022-11-30 16:00:54.174 INFO     NC tibit@10.2.10.66 <get> /tibitnc:netconf-state succeeded with
response 'ok'
```

Limitations

NETCONF Server

Cisco Routed PON ConfD Netconf Server has the following limitations:

- The following NETCONF datastores are not supported: startup, candidate
- The following NETCONF capabilities are not supported: :confirmed-commit, :rollback-on-error, :startup.

Cisco YANG Models

- Standard /ietf-netconf-notifications:netconf-config-change events are not generated when MongoDB changes are made by applications other than the Netconf Server (e.g., PON Controller device discovery, PON Manager UI, etc.)
- NETCONF <get> operations for retrieving statistics and logs for all devices are limited to returning a single most recent entry per device.
- NETCONF <get> operations for retrieving statistics and logs for a single device are limited to returning 128 most recent stats entries and 1024 most recent log entries respectively.

DISCLAIMER

Cisco reserves the right to make changes without any further notice to any products or data herein to improve reliability, function, or design.

Cisco does not assume any liability arising out of the application or use of this information, nor the application or use of any products or circuits described herein, neither does it convey any license under its patent rights, nor the rights of others.

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA

Asia Pacific Headquarters
Cisco Systems (USA) Pte. Ltd.
Singapore

Europe Headquarters
Cisco Systems International BV Amsterdam,
The Netherlands

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