UPDU CLI Reference Manual

Riedo Networks Ltd
Switzerland

Revision: 3.2.0-e577406e
Date: March 26, 2024

Contents

1 Getting started

This document describes the usage of the Command Line Interface (CLI). All available commands are grouped based on the functionality these are related to.

The UPDU CLI allows accesing and configuration of all parts of the UPDU. It can be accessed locally with a serial cable (*e.g. RN1080*) connected to the AUX3 port or remotely through SSH and Telnet if enabled.

1.1 Using the UPDU CLI

The UPDU CLI is structured into two modes:

- User mode: allowing to get system information, work with measurement data, toggle outlets etc.
- Configuration mode: allowing mode to change UPDU settings.

When in user mode, the prompt is composed of the hostname and ">":

```
updu-100499>
```

In the configuration mode, the prompt changes:

```
updu-100499(config)#
```

Unless documented otherwise, commands entered in the configuration mode are effective immediately but are not persisted until the `write` command is invoked.

It is important to persist the settings as a restart of the UPDU will revert the configuration to the last persisted state

1.2 Command Line Editing and Command History

The command line can be edited using the Left and Right keys, using Backspace, Delete, End and Home.

A history of already entered commands is kept. It can be accessed using the Up and Down keys.

Furthermore, the following key combinations are available:

- CTRL-a: moves the cursor the start of the line
- CTRL-b: moves the cursor left one character
- CTRL-c: clears the current line and refreshes the prompt
- CTRL-d: deletes the character under the cursor; if pressed on an empty line, exits the current mode or the CLI session when already in the main commands
- CTRL-e: moves the cursor to the end of the line
- CTRL-f: moves the cursor right one character
- CTRL-k: deletes from the cursor to the end of the line
- CTRL-l: refreshes the current line
- ALT-b: moves the cursor backward one word
- ALT-f: moves the cursor forward one word
- ALT-Backspace: deletes the word left of the cursor

1.2.1 Show CLI History

The `show history` command shows the CLI commands in the history. The Up and Down keys can be used to navigate through the commands.

```
updu-100499> show history
1. show power
2. show energy
3. show history
```

1.2.2 Clear CLI History

The command history can be cleared using the `clear history` command.

1.3 Help

1.3.1 Contextual Help

Entering a question mark (`?`) at the prompt gives information about the currently entered command or shows a list of all available commands matching an partially entered command.

1.3.2 Get Help

Enter `help` for an online help.

1.4 Command Abbreviation

The UPDU CLI allows commands to be abbreviated to the number of letters that make them unique. For example, it is sufficient to enter `sh conf` for `show config`.

Some commands are explicitly exempted from command abbreviation.

Note that depending on the commands that will be added in future versions of the UPDU firmware, it is possible that the minimal number of letters for a specific command may change.

1.5 Comments

Text after an exclamation mark (!) or a hash (#) character is ignored by the CLI. This allows to have comments in configurations.

1.6 Output Filtering

Some commands, mainly the `show` commands, allow their output to be filtered to display only selected parts of the output. Example:

```
updu-100499> show power | include module
Module1                 0.0      231.0          0          0
```

The following output filters are supported:

- `begin PATTERN` : Filter anything until a line containing `PATTERN` appears.
- `include PATTERN` : Print all lines containing `PATTERN`.
- `exclude PATTERN` : Filter all lines containing `PATTERN`.
- `section PATTERN` : Show the first matching section. A section is a sequence of lines where the first one starts with the string `PATTERN` and all subsequent lines are further indented than the first line.

The case is ignored for the pattern matching.

1.7 Clear Screen

The `clear screen` command clears the screen and puts the cursor at the top left position.

1.8 Configuration Mode

When the `configure` command is entered, the configuration mode is activated. The configuration mode provides a number of sub-modes which allow configuring a specific part of the system.

Sub-modes can be left using the `exit` command. `exit` in the main configuration mode leaves the configuration mode and returns into the normal mode.

The prompt reflects the currently active mode.

Example configuration session modifying the hostname:

- Enter the configuration mode:

```
updu-100499> configure
updu-100499(config)#
```

- Enter the `system` configuration sub-mode:

```
updu-100499(config)# system
updu-100499(config-system)#
```

- Change the hostname to `my-pdu`:

```
updu-100499(config-system)# hostname my-pdu
my-pdu(config-system)#
```

- Return to the main configuration mode:

```
my-pdu(config-system)# exit
my-pdu(config)#
```

- Return to the normal mode:

```
my-pdu(config)# exit
my-pdu>
```

Unless documented otherwise, commands entered in the configuration mode are effective immediately but are not persisted until the "write" command is invoked.

1.8.1 Execute Normal Commands

Some non-configuration commands can be executed while being in the configuration mode. Simply prepend the desired command with `do`.

Example:

```
updu-100499(config)# do show power
Name                  I [A rms]  U [V rms]      P [W]    Q [var]
PDU                         0.0      233.3          0          0
Module 1                    0.0      233.3          0          0
...
```

1.8.2 Save Configuration

As mentioned earlier, the `write` command is used to persist configuration changes. If configuration is not persisted, a reboot will revert it back to the last persisted state.

Note that this command can not be issued from within the `configuration` context.

1.8.3 Show Configuration

The `show config` command shows the currently active UPDU configuration. The output of this command can be stored as a backup. The complete text can also be pasted as is into the `configure` context to apply settings.

Certificate and private key data is hidden by default. In order to show the entire configuration, use the `show config all` command.

1.9 Logout

The `logout` command terminates the current CLI session:

```
updu-100499> logout
Bye...
```

2 System

2.1 System Names

2.1.1 Device Name Configuration

The device name is configured in the `system` configuration sub-mode. It is shown in the web interface and used as `sysName` in the SNMP MIB-II System Group.

Example:

```
updu-100499> configure
updu-100499(config)# system
updu-100499(config-system)# device-name servers-pdu
updu-100499(config-system)# system
```

2.1.2 Hostname Configuration

The hostname can be configured in the `system` configuration sub-mode. It is used in DHCP requests.

Example:

```
updu-100499> configure
updu-100499(config)# system
updu-100499(config-system)# hostname mypdu
mypdu(config-system)#
```

2.2 Identify

2.2.1 Identify Object (UID)

The UID function (short for unit identification) is useful to help an on-site engineer to find a PDU, a module or an outlet by blinking the outlet LEDs of the selected object. It is activated with the `identify` CLI command. In a typical UPDU setup with multiple devices next to each other, this helps to avoid wiring issues.

By default, the UID function is active for 5 minutes.

Examples:

- Identify a module:

```
updu-100499> identify Module2
updu-100499>
```

- Identify Outlet2.3 during 60 seconds:

```
updu-100499> identify Outlet2.3 60
updu-100499>
```

- Stop a running UID function:

```
updu-100499> identify stop
updu-100499>
```

2.3 Reboot

2.3.1 Reboot UPDU

The `reboot` command reboots the UPDU:

```
updu-100499> reboot
Rebooting...
```

By default, only the Interface and Controller Module (ICM) is rebooted, the individual metering modules and the Ethernet switch on ETH1/ETH2 continues running. This is to avoid measurement gaps and to keep the network up in case of a ring configuration.

To also reboot these peripherals, use these commands:

- Reboot the ICM and the metering modules:

```
updu-100499> reboot modules
Rebooting...
```

- Reboot the ICM and the Ethernet switch:

```
updu-100499> reboot switch
Rebooting...
```

- Reboot the ICM, the metering modules and the Ethernet switch:

```
updu-100499> reboot all
Rebooting...
```

2.3.2 Schedule reboot UPDU

A reboot can be scheduled with `reboot in <minutes>`.

Note that this can be very helpful when remotely chaning the configuration, especially networking and service related settings. One would schedule a reboot and start changing the configuration. Should for some reason the connection been lost (e.g. wrong IP settings, disable SSH), the UDPU reboots after a certain period and restors the last persisted (`write`) configuration.

2.3.3 Cancel a schedule reboot UPDU

Cancel a scheduled reboot (`reboot in X` command) with `reboot cancel`.

2.3.4 Show Reboot

The 'show reboot' command shows information whether a reboot is scheduled or not. If a reboot is scheduled, the remaining time until reboot is shown.

```
updu-100499> show reboot
No reboot scheduled
updu-100499> reboot in 15
Scheduled reboot in 15 minutes
updu-100499> show reboot
Device will reboot in 14:57
```

2.4 System Power

2.4.1 Show Power over Ethernet Information

The `show poe` command is used to show information on Power over Ethernet. The following states exist:

- `PoE powered`: The UPDU is currently powered by PoE.
- `AC powered, supplying PoE power`: The UPDU is mains powered and is providing power to another device by PoE.

- `AC powered, not supplying PoE power`: The UPDU is mains powered and is not providing power to another device by PoE.

Example:

```
updu-100499> show poe
PoE powered
```

2.5 Product Information

2.5.1 Show Module Information

The `show module info` command shows information about all available modules.

```
updu-100499> show module info
module slot 0: Module1
  label 1
  part-number 100-0290-3
  serial-number 1006954
  lot-number "2132DA"
  image-1
    status RUNNING
    version 3.3.1-d17622cb
  image-2
    status BACKUP
    version 3.3.0-d9ded1fe
  nominal 16 A 230 V
  outlets
    0: C13 Outlet, 10 A
    1: C13 Outlet, 10 A
    2: C13 Outlet, 10 A
    3: C13 Outlet, 10 A
    4: C13 Outlet, 10 A
    5: C13 Outlet, 10 A
    6: C19 Outlet, 16 A
    7: C19 Outlet, 16 A
...
```

2.6 System Time

2.6.1 Show System Time

To see the date, time, uptime and the time of the last SNTP update, use the `show time` command:

```
updu-100499> show time
Date: 2021-10-18
Time: 11:11:30 UTC
Local Date/Time: 2021-10-18 13:11:30 timezone CEST
Last SNTP update: 0h48:00 ago
Uptime: 0h48:06
```

2.6.2 Configure Local Time

There are 2 different ways to configure the local time:

- By specifying a system known timezone. This method implies daylight saving time.
- By creating a custom local time with a fixed offset from UTC.

Examples:

- Configure a system timezone :

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# local zone Europe/Berlin
updu-100499(config-time)#
```

- Configure a custom time offset from UTC:

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# local offset -01:00 "myzone UTC-1"
updu-100499(config-time)#
```

- Use UTC:

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# local UTC
updu-100499(config-time)#
```

2.6.3 Simple Network Time Protocol (SNTP) Client

When enabled, the SNTP client polls the configured time server in regular intervals to synchronize the time of the UPDU.

Enable or Disable SNTP

By default the SNTP client is enabled. It can be disabled if the UPDU does not have access to a time server.

Examples:

- Disable the SNTP time synchronization:

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# sntp disabled
updu-100499(config-time)#
```

- Enable the SNTP time synchronization:

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# sntp enabled
updu-100499(config-time)#
```

SNTP Server Address and Port

Examples:

- Configure the factory-default SNTP server with the default port (123):

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# sntp server pool.ntp.org
updu-100499(config-time)#
```

- Configure a local SNTP server with a non-standard port:

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# sntp server timeserver.company.com port 1123
updu-100499(config-time)#
```

SNTP Poll Interval

Examples:

- Configure the factory-default 60 minutes:

```
updu-100499> configure
updu-100499(config)# time
updu-100499(config-time)# sntp poll-interval 60
updu-100499(config-time)#
```

2.6.4 Set the System Time Manually

The `change datetime` command can be used to set the system time manually.

Example: Set the device date and time to 2020-01-01 12:00:

```
updu-100499> change datetime 2020-01-01 12:00:00
updu-100499>
```

Notes:

- When the SNTP client is enabled, the entered date and time will be overwritten the next time the client synchronizes the time.
- The entered date and time is stored in the UPDU's real time clock (RTC) and is guaranteed to survive reboots.

2.7 Firmware

2.7.1 Show Version

The `show version` command shows information about the PDU and about the currently running firmware, about the backup firmware and about the firmware for the PIM/POM modules which is embedded in the currently running firmware image:

```
updu-100499> show version
PDU model: RN3005
PDU serial-number: 100499
PDU part-number: 100-0603-1
PDU lot-number: 2137CA
Running UPDU firmware: 2.3.0-DEV-dd4920fe
Backup UPDU firmware: 2.3.0-DEV-55f5449a
Embedded PIM/POM firmware: 3.3.1-d17622cb
RTOS: 2.6.0
Network stack: TCP 2.1.2/SSL 2.1.2/Crypto 2.1.2/SSH 2.1.2/STP 2.1.2
Toolchain: GCC 10.3.0
```

2.8 Local User Interface

2.8.1 Screen Blanker Configuration

By default, the UPDU display is switched off after 10 minutes of inactivity. This behaviour can be changed in the `display` configuration sub-mode.

- Disable the screen blanker:

```
updu-100499> configure
updu-100499(config)# display
updu-100499(config-display)# blank-time off
updu-100499(config-display)#
```

- Blank the display after 5 minutes:

```
updu-100499> configure
updu-100499(config)# display
updu-100499(config-display)# blank-time 5:00
updu-100499(config-display)#
```

2.8.2 LED Brightness

The brightness of the LEDs on the measurement modules of a UPDU can be controlled in the `display` configuration sub-mode. Three brightness levels are available, `low`, `medium` and `high`. By default, the brightest setting is configured.

Example:

- Set LED brightness to `low`:

```
updu-100499> configure
updu-100499(config)# display
updu-100499(config-display)# led-brightness low
updu-100499(config-display)#
```

3 Objects

3.1 Object Configuration

The `object` configuration sub-mode can be used to modify the configuration of UPDU objects (e.g. modules, outlets etc.).

3.1.1 Object Description

Using `description`, additional information related to the setup of the PDU can be added to each object.

Example:

```
updu-100499> configure
updu-100499(config)# object Outlet4.1
updu-100499(config-object)# description "Server 42"
updu-100499(config-object)#
```

3.1.2 Object Name

For each object, a case insensitive unique name can be defined which can then be used to address the object.

Example:

- Configure a name for `Outlet2.1`:

```
updu-100499> configure
updu-100499(config)# object Outlet2.1
updu-100499(config-object-Outlet2.1)# name MyServer
```

- Delete a name:

```
updu-100499> configure
updu-100499(config)# object Outlet2.1
updu-100499(config-object-Outlet2.1)# delete name
```

3.2 Wiring Information

3.2.1 Show Wiring

The `show wiring` shows the electrical structure of the UPDU along with electrical capabilities and installed options, such as `RCM`, `Relay` and `OVP`:

```
PDU [RCM,OVP]
├─ Inlet [CEE32/5,3x32A]
|  ├─ WireL1 [32A]
|  |  ├─ Branch1 [Carling type MCB,16A]
|  |  |  └─ Module1 [16A]
|  |  |     ├─ Outlet1.1 [Relay,C13,10A]
|  |  |     ├─ Outlet1.2 [Relay,C13,10A]
|  |  |     ├─ Outlet1.3 [Relay,C13,10A]
|  |  |     ├─ Outlet1.4 [Relay,C13,10A]
|  |  |     ├─ Outlet1.5 [Relay,C13,10A]
|  |  |     ├─ Outlet1.6 [Relay,C13,10A]
|  |  |     ├─ Outlet1.7 [Relay,C19,16A]
|  |  |     └─ Outlet1.8 [Relay,C19,16A]
...
|  └─ WireN [32A]
```

4 Users & Roles

4.1 Roles & Permissions

Each role defines a set of permissions that all users holding that role have. Users can have multiple roles.

4.1.1 Predefined Roles

After a factory reset, the UPDU has the following roles:

- `admin` : Has all available privileges and cannot be deleted or modified.
- `guest` : Is used for read-only users without any administrative permissions.
- `snmp-read` : Has SNMP read permission (for SNMPv3).
- `snmp-write` : Has SNMP write permission (for SNMPv3).

The latter three roles can be freely modified or deleted.

4.1.2 Permissions

The following permissions are known by the system:

- `change-credentials` : Allows to change the personal password.
- `cli-access` : Allows to access the CLI by Serial Console, Telnet or SSH.
- `configure` : Allows to configure the UPDU.
- `identify-objects` : Allows to use the `identify` command.
- `outlet-switching` : Allows to switch outlets on or off.
- `reboot` : Allows to reboot the UPDU.
- `snmp-read` : Allows to read data by SNMPv3.
- `snmp-write` : Allows to write data by SNMPv3.
- `update-firmware` : Allows to update the UPDU firmware.
- `view-data` : Allows to view UPDU measurement data and other information.
- `web-access` : Allows to log in to the web UI.

4.1.3 Add or Modify a Role

The `role` command in the `roles` configuration sub-mode selects the role to modify. If the role doesn't exist, a new role without any permissions is created.

When a role is selected, the prompt changes to include the role name of the selected role. In this mode, the following command can be used to modify the role:

- `permissions set [<PERMISSION>...]` : Set the desired permission for the role.
- `permissions none` : Clear all permissions.

Examples:

- Create Role with only web interface access:

```
updu-100499> configure
updu-100499(config)# roles
updu-100499(config-roles)# role web-users
Created new role web-users.
updu-100499(config-roles-web-users)# permissions set web-access
updu-100499(config-roles-web-users)#
```

- Remove all permissions for guest users:

```
updu-100499> configure
updu-100499(config)# roles
updu-100499(config-roles)# role guest
updu-100499(config-roles-guest)# permissions set none
updu-100499(config-roles-guest)#
```

4.1.4 Delete Roles

To delete a role, the `delete` command in the `roles` configuration sub-mode is used.

```
updu-100499> configure
updu-100499(config)# roles
updu-100499(config-roles)# delete guest
updu-100499(config-roles)#
```

4.2 Local Authentication

4.2.1 Add or Modify a User

The `user` command in the users configuration sub-mode selects the user to modify. If the user doesn't exist yet, a new user without any roles and without a password is created.

When a user is selected, the prompt changes to include the username of the selected user. In this mode, the following commands can be used to modify the user:

- `password <PASSWORD>` : Set a new password, given in clear-text. The clear-text password is transformed using an irreversible function and the resulting hash is stored in the configuration.
- `password-hash <HASH>` : Set the user's password to the specified hash.
- `roles set <ROLE>...` : Set the roles of the user.
- `roles none` : Clear all roles.
- `ssh-key add <KEY-DATA>` : Adds the specified SSH public key to this user. Using SSH public keys, users can log in without providing a password. Up to 8 keys can be configured for each user.
- `ssh-key delete <ID>` : Deletes the user's SSH public key identified by `<ID>` .
- `ssh-key <ID> <KEY-DATA>` : Set a user's SSH public key, `<ID>` being the number of the key (between 1 and 8).

SSH public keys have to be in OpenSSH format (key format identifier followed by the key all on a single line). The following key types are supported: `ssh-ed25519` , `ecdsa-sha2-nistp256` , `ecdsa-sha2-nistp384` , `ecdsa-sha2-nistp521` and `ssh-rsa` . `ssh-rsa` keys have to be between 2048 and 4096 bits (note that the CLI currently does not verify the key length so it is possible to configure shorter keys but they will not work for authentication).

Examples:

- Select the `admin` user and set a new password:

```
updu-100499> configure
updu-100499(config)# users
updu-100499(config-users)# user admin
updu-100499(config-users-admin)# password VerySecurePassword
```

- Create a new guest user with password `visitor`:

```
updu-100499> configure
updu-100499(config)# users
updu-100499(config-users)# user guest
Created new user guest.
updu-100499(config-users-guest)# roles set guest
updu-100499(config-users-guest)# password visitor
```

- Allow the `admin` user to log in using a SSH public key:

```
updu-100499> configure
updu-100499(config)# users
updu-100499(config-users)# user admin
updu-100499(config-users-admin)# ssh-key add "ssh-rsa ..."
```

User passwords are stored as PBKDF2-SHA256 hashes. The `show config` command shows the hash.

4.2.2 Delete a User

To delete a user, use the `delete` command:

```
updu-100499> configure
updu-100499(config)# users
updu-100499(config-users)# delete old-user
updu-100499(config-users)#
```

The logged-in user cannot delete itself.

4.2.3 Configure a User's SNMPv3 Settings

Each configured user can be enabled for SNMP version 3 access.

For SNMPv2 access configuration, refer to the SNMP Configuration section.

To enable SNMPv3 access for the `myuser`:

```
updu-100499> configure
updu-100499(config)# users
updu-100499(config-users)# user myuser
updu-100499(config-users-myuser)# snmpv3 enabled
```

To disable SNMPv3 access for the `myuser`:

```
updu-100499> configure
updu-100499(config)# users
updu-100499(config-users)# user myuser
updu-100499(config-users-myuser)# snmpv3 disabled
```

A user configured for SNMPv3 access needs at least one of these permissions, depending on their access mode:

- `snmp-read` users are allowed to read values.
- `snmp-write` users are allowed to write values, e.g. switch outlets.

The permission is assigned through the role which the user holds. Users without one of these permissions are configured in the SNMP agent, but are unable to request or set anything.

A user's configured password is not relevant for SNMPv3. SNMPv3 is secured with an authentication function (MD5 or SHA1) and an associated password, as well as a privacy function (DES or AES) and password:

Examples:

- Enable SHA1 authentication but no encryption for the currently selected user:

```
updu-100499(config-users-myuser)# snmpv3 auth sha1 password "myAuthPassword"
updu-100499(config-users-myuser)# snmpv3 privacy none
```

- Enable MD5 authentication and AES encryption for the currently selected user:

```
updu-100499(config-users-myuser)# snmpv3 auth sha1 password "myAuthPassword"
updu-100499(config-users-myuser)# snmpv3 privacy aes password "myPrivacyPassword"
```

Notes:

- The authentication and privacy passwords need to be at least 8 characters long.
- It is not possible to encrypt without using authentication.
- Both the SNMPv3 authentication and privacy passwords are stored in clear text and are shown with the `show config` command.

4.2.4 Change the Current User's Password

The `change password` CLI command allows the currently logged in user to change the personal password. The user is prompted to first enter the current password, before the new password has to be entered twice for verification:

```
updu-100499> change password
Enter the current password: ...
Enter new password: ...
Repeat new password: ...
```

Notes:

- The `change password` command only works for local users. For remotely authenticated users the password has to be modified through other means.
- The `change password` command is only available for users having the `change-credentials` permission (see *Permissions*).

4.3 RADIUS Authentication

RADIUS (short for Remote Authentication Dial In User Service) allows a UPDU to authenticate users on a RADIUS server without having to create them locally on the UPDU. When RADIUS is enabled and configured, the username and password of users trying to log in is sent to the RADIUS server which verifies if the user has access and which then responds with the roles of that user.

4.3.1 Server Setup

In order to transmit user roles to the UPDU, a vendor specific attribute called "RNX-UPDU-Roles" consisting of a string with a comma-separated list of roles must be configured for each user.

For FreeRADIUS, this can be achieved with this setting in the `dictionary` configuration file:

```
VENDOR RNX 55108
ATTRIBUTE RNX-UPDU-Roles 1 string RNX
```

Users can now be configured as follows:

```
bob     Cleartext-Password := "bobspassword"
        RNX-UPDU-Roles = "admin"
```

4.3.2 Enable or disable RADIUS

- Disable RADIUS:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# radius disabled
updu-100499(config-auth)#
```

- Enable RADIUS:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# radius enabled
updu-100499(config-auth)#
```

4.3.3 Server Address

The hostname or IP address of the RADIUS server is configured using the `radius server` command.

Examples:

- Set the RADIUS server to `authserver.local`:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# radius server authserver.local
updu-100499(config-auth)#
```

The `radius server` command also allows to configure the port to which RADIUS requests are sent. If not specified, port 1812 is used.

Examples:

- Use port 21812 for RADIUS authentication requests:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# radius server authserver.local auth-port 21812
updu-100499(config-auth)#
```

4.3.4 Shared Secret

The RADIUS shared secret is configured using the `radius shared-secret` command.

Examples:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# radius shared-secret "I am secret!"
updu-100499(config-auth)#
```

Notes:

- The RADIUS shared secret is stored in clear text and is shown with the `show config` command.

4.3.5 Various Options

The RADIUS timeout option controls how long the UPDU waits for a response from the RADIUS server until it considers retransmitting it. The timeout is given in milliseconds and defaults to 2000 ms:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# radius timeout 2000
updu-100499(config-auth)#
```

The RADIUS retries option defines how many times a timed out request is retransmitted until it is declared failed. By default it does 3 retries:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# radius retries 3
updu-100499(config-auth)#
```

4.4 LDAP Authentication

LDAP (short for Lightweight Directory Access Protocol) allows a UPDU to authenticate users on a authentication server providing LDAP interface without having to create them locally on the UPDU. When LDAP is enabled and configured, the username and password of users trying to log in is sent to the LDAP authentication server which verifies if the user has access and which then responds with the groups the user is member of.

Currently supported authentication server is Microsoft Active Directory.

Configuration of a Microsoft Active Directory User

In order to authenticate a user with certain UPDU roles, it is necessary for the user to be member of at least one group with the name RNX-UPDU-[role].

Example : User bob is member of groups RNX-UPDU-admin and RNX-UPDU-snmp-read.

4.4.1 Enable and Disable

- Disable LDAP:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap disabled
updu-100499(config-auth)#
```

- Enable LDAP:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap enabled
updu-100499(config-auth)#
```

4.4.2 Server Address

The hostname or IP address of the LDAP authentication server is configured using the `ldap server` command.

Examples:

- Set the LDAP server to `authserver.local`:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap server authserver.local
updu-100499(config-auth)#
```

4.4.3 Transport Mode

The TCP transport mode is configured using the `ldap transport` command.

Examples:

- Set the LDAP transport to plain:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap transport plain
updu-100499(config-auth)#
```

- Set the LDAP transport to tls:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap transport tls
updu-100499(config-auth)#
```

The `ldap transport` command also allows to configure the port to which LDAP requests are sent. If not specified, port 389 for plain and port 636 for tls transport are used.

Examples:

- Use port 1636 for LDAP tls requests:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap transport tls port 1636
updu-100499(config-auth)#
```

In some installations, e.g. when users are members of many groups, LDAP responses can grow very large and cause authentication problems. In this case, the size of the receive buffer may need to be tuned using the optional `rx-buffer-size` parameter. By default, buffer size is 2860 bytes for plain connections, and 16384 bytes for TLS connections.

Examples:

- Increase the receive buffer of a plain connection to 8 KB:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap transport plain rx-buffer-size 8192
updu-100499(config-auth)#
```

4.4.4 Bind Authentication

In order for the UPDU to request the LDAP authentication server, a user having read access is on the LDAP directory is required.

Examples:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap bind-auth CN=admin,CN=Users,DC=authserver,DC=local "I am secret!"
updu-100499(config-auth)#
```

or with the Active Directory logon name:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap bind-auth admin@authserver.local "I am secret!"
updu-100499(config-auth)#
```

Notes:

- The LDAP bind password is stored in clear text and is shown with the `show config` command.

4.4.5 Base Search DN

A base DN may be given point in the directory where to start the search for the user.

Examples:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap base-dn CN=Users,DC=authserver,DC=local
updu-100499(config-auth)#
```

4.4.6 Login Name Attribute

The Login Name Attribute may be adapted according to your LDAP schema. By default it is set to `sAMAccountName`.

Examples:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap login-name-attribute sAMAccountName
updu-100499(config-auth)#
```

In Active Directory this may be set to `userPrincipalName` for users to authenticate with their AD logon name.

Examples:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap login-name-attribute userPrincipalName
updu-100499(config-auth)#
```

4.4.7 Request Timeout

The timeout in milliseconds may be configured for all LDAP requests.

Examples:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# ldap timeout 2000
updu-100499(config-auth)#
```

4.5 TACACS+ Authentication

TACACS+ (short for Terminal Access Controller Access-Control System Plus) is a protocol allowing a UPDU to authenticate users through an external server. When configured and enabled, the UPDU sends credentials of users trying to log in to the configured server which checks if that user should be granted access and what roles it is member of.

4.5.1 Server Setup

Users are configured as follows on the TACACS+ server:

```
user = <USERNAME> {
  login = <PASSWORD-SPEC>
  service = rnx-updu {
     roles="<ROLES>"
  }
}
```

Where `<USERNAME>` is the username allowed to log in, `<PASSWORD-SPEC>` is a specification of the password (see the documentation of your TACACS+ server for more information) and `<ROLES>` is a comma-separated list of roles which are configured on the UPDU in question. The user logging in will have the combined set of permissions of all the specified roles. Note that whitespace is not stripped from the list of roles.

4.5.2 Enable or disable TACACS+

- Disable TACACS+:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# tacacs disabled
updu-100499(config-auth)#
```

- Enable TACACS+:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# tacacs enabled
updu-100499(config-auth)#
```

4.5.3 Server Address

The hostname or IP address of the TACACS+ server is configured using the `tacacs server` command.

Examples:

- Set the TACACS+ server to `authserver.local`:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# tacacs server authserver.local
updu-100499(config-auth)#
```

The `tacacs server` command also allows to configure the port to which TACACS+ requests are sent. If not specified, port 49 is used.

Examples:

- Use port 4949 for TACACS+ requests:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# tacacs server authserver.local port 4949
updu-100499(config-auth)#
```

4.5.4 Shared Secret

The TACACS+ shared secret is configured using the `tacacs shared-secret` command.

Examples:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# tacacs shared-secret "super-secret"
updu-100499(config-auth)#
```

Notes:

- The TACACS+ shared secret is stored in clear text and is shown with the `show config` command.

4.5.5 Various Options

The TACACS+ timeout option controls how long the UPDU waits for a response from the TACACS+ server until it considers the request failed. The default timeout is 5 seconds:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# tacacs timeout 5000
updu-100499(config-auth)#
```

4.6 Authentication Method Order

The UPDU allows for users to be configured locally and remotely. Local users are configured on the UPDU itself, while remote users are configured on an external service such as *RADIUS*, *TACACS+* or *LDAP*, and the UPDU is configured to trust the external service to authenticate users.

The UPDU tries to authenticate users locally first. When a user exists locally, remote authentication methods are skipped.

When the authenticating user does not exist locally, the remote authentication methods RADIUS, TACACS+ and LDAP are tried, depending on what is configured.

If more than one external authentication method is configured, the order of the remote authentication methods can be changed with the `order` command. By default, RADIUS is tried before LDAP, which is tried before TACACS+.

Example:

- Authenticate LDAP before RADIUS and TACACS+:

```
updu-100499> configure
updu-100499(config)# auth
updu-100499(config-auth)# order local ldap radius tacacs
updu-100499(config-auth)#
```

Note that the `order` command does not disable or enable authentication methods.

5 Measurement

5.1 Power

5.1.1 Show Instantaneous Power

The `show power` command shows the latest current, voltage and power measurements available:

```
updu-100499> show power
Object     Name                    I [A rms]  U [V rms]    P [W]  Q [var]     PF
PDU        cabinet3-4                    0.0        0.0        0        0  0.000
WireL1                                   0.0        0.0        0        0  0.000
WireL2                                   0.0        0.0        0        0  0.000
WireL3                                   0.0        0.0        0        0  0.000
WireN                                    0.0        n/a      n/a      n/a    n/a
...

Legend: I: Current, U: Voltage, P: Active power, Q: Reactive power, PF: Power Factor
```

These values are updated approximately once per second.

With `show power description`, the object descriptions are added to the output.

5.1.2 Show Instantaneous Frequency

The `show frequency` command shows the latest frequency measurements available:

```
 updu-101782> show frequency
Object     Name                      F [Hz]
WireL1                                49.99
WireL2     Phase2                     49.99
WireL3                                49.99

Legend: F: Frequency
```

These values are updated approximately once per second.

With `show frequency description`, the object descriptions are added to the output.

5.2 Energy

5.2.1 Show Energy Counters

The `show energy` shows the energy counters:

```
updu-100499> show energy
Object     Name                          A+ [kWh]     R1 [kvarh]     R4 [kvarh]
PDU        cabinet3-4                       4.039          0.000          2.019
WireL1                                      4.038          0.000          2.019
WireL2                                      0.000          0.000          0.000
WireL3                                      0.000          0.000          0.000
Branch1                                     0.109          0.025          0.001
Branch2                                     0.109          0.026          0.000
Branch3                                     0.109          0.025          0.000
...

Legend:
  A+: Positive active energy
  R1: Positive reactive energy (quadrant 1)
  R4: Negative reactive energy (quadrant 4)
```

Note that the energy counters can not be reset, which is by design.

With `show energy description`, the object descriptions are added to the output.

5.3 Environment

5.3.1 Show Sensor Measurements

The `show sensor` command shows the measurements of the sensors plugged into the AUX ports:

```
updu-100123> show sensor
Object     Name                          Port      T [C]   RH [RH%]    DP [Pa]
Sensor1                                  AUX1        n/a        n/a        n/a
Sensor2                                  AUX2        n/a        n/a      0.033
Sensor3                                  AUX3        n/a        n/a        n/a

Legend: T: Temperature, RH: Relative Humidity, DP: Differential Pressure
```

With `show sensor description`, the object descriptions are added to the output.

5.4 Residual Current

5.4.1 Show Residual Current

On UPDUs equipped with an RCM module, the residual current can be shown using the `show rcm` command:

```
updu-100499> show rcm
Object     Name                    RMS [mA rms]      DC [mA]
RCM                                         0.7          0.3

Legend: RMS: RMS residual current, DC: DC residual current
```

With `show rcm description`, the object descriptions are added to the output.

5.4.2 RCM Information

The `show rcm info` command gives more information about the RCM modules:

```
updu-100499> show rcm info
Object               Info
RCM                  S/N: 2002514564, APP: 107, API: 256, SW: 604
```

5.4.3 RCM Statistics

The `show rcm stats` command shows statistics related to the communication with the RCM module:

```
updu-100499> show rcm stats
Object                     tx  timeouts     unexp       crc     excep
RCM                      2426         0         0         0         0
```

Data fields:

- `tx` : Number of commands sent to the RCM module
- `timeouts` : Number of commands timed out
- `unexp` : Number of unexpected responses
- `crc` : Number of CRC errors
- `excep` : Number of reported exceptions

5.4.4 RCM Module Test

RCM modules have a built-in selftest function. The test takes about 2 seconds, during which the RCM values are not updated.

```
updu-100499> test rcm
PDU                 Test OK
```

6 Control

6.1 Outlet Switching

Outlets equipped with relays can be switched on and off individually. Outlets are identified using the module label and the outlet number printed on the PDU.

6.1.1 Switch Outlet Off

Outlets can be switched off using the `outlet off` command:

```
updu-100499> outlet off Outlet1.3
```

6.1.2 Switch Outlet On

Outlets can be switched on using the `outlet on` command:

```
updu-100499> outlet on Outlet1.1
```

6.1.3 Power Cycle Outlet

To power cycle an outlet, use the `outlet cycle` command. The outlet in question is switched off and on again after five seconds. This can come handy if for example the network connection to the UPDU is powered off and on after a certain period.

```
updu-100499> outlet cycle Outlet1.7
```

6.2 Configuration

6.2.1 Power Cycle Delay

For switchable outlets, the power cycle delay can be configured. It defaults to 5 seconds.

Example:

- Set the power cycle delay of `Outlet2.1` to 30 seconds:

```
updu-100499> configure
updu-100499(config)# object Outlet2.1
updu-100499(config-object-Outlet2.1)# powercycle-delay 30
```

- Set the power cycle delay to the default value:

```
updu-100499> configure
updu-100499(config)# object Outlet2.1
updu-100499(config-object-Outlet2.1)# powercycle-delay default
```

7 Monitoring

7.1 Rules

With `rules`, it is possible to define conditions which are then monitored. If a condition is not satisfied, an alert message is logged which can be sent by SNMP traps or Syslog.

Example:

- Configure that the voltage of Outlet4.1 is expected to be within 10% of 230V (207.0V - 253.0V):

```
updu-100499> configure
updu-100499(config)# object Outlet4.1
updu-100499(config-object-Outlet4.1)# rules
updu-100499(config-object-Outlet4.1-rules)# voltage 207 none none 253
```

7.1.1 Current Monitoring

Current monitoring is enabled by defining current thresholds:

```
current <critical-low> <warning-low> <warning-high> <critical-high>
```

Thresholds may be omitted by entering `none`.

Example:

- On WireL1 and WireL2, currents above 8.0A are considered critical. On WireL2, currents between 4.0A and 8.0A generate a warning:

```
updu-100499> configure
updu-100499(config)# object WireL1
updu-100499(config-object-WireL1)# rules
updu-100499(config-object-WireL1-rules)# current none none none 8.0
updu-100499(config)# object WireL2
updu-100499(config-object-WireL2)# rules
updu-100499(config-object-WireL2-rules)# current none none 4.0 8.0
```

7.1.2 Voltage Monitoring

Voltage monitoring is enabled by defining voltage thresholds:

```
voltage <critical-low> <warning-low> <warning-high> <critical-high>
```

Thresholds may be omitted by entering `none`.

Example:

- For Outlet4.1, voltages below 207V and above 253V are considered critical, within the [207V, 218.5V] and [241.5V, 253V] ranges they are warning and within the [218.5V, 241.5V] range they are acceptable.

```
updu-100499> configure
updu-100499(config)# object Outlet4.1
updu-100499(config-object-Outlet4.1)# rules
updu-100499(config-object-Outlet4.1-rules)# voltage 207 218.5 241.5 253
```

7.1.3 Residual Current Monitoring

Residual current monitoring on RCM modules is enabled by defining current thresholds. Monitoring can be set on RMS or DC current:

```
resiudal-current rms <warning> <critical>
resiudal-current dc <warning> <critical>
```

Thresholds may be omitted by entering `none`.

Example:

- Residual RMS currents above 10.0mA are considered critical. Residual RMS currents between 4.5mA and 10.0mA generate a warning. Residual DC currents above 5.0mA are considered critical:

```
updu-100499> configure
updu-100499(config)# object RCM
updu-100499(config-object-WireL1)# rules
updu-100499(config-object-WireL1-rules)# residual-current rms 4.5 10.0
updu-100499(config-object-WireL1-rules)# residual-current dc none 5.0
```

7.1.4 Temperature Monitoring

Temperature monitoring is enabled by defining temperature thresholds:

```
temperature <critical-low> <warning-low> <warning-high> <critical-high>
```

Thresholds may be omitted by entering `none`.

Example:

- On Sensor1 and Sensor2, temperatures above 29°C and below 5°C are considered critical. On Sensor2, temperatures between 25°C and 29°C or between 5°C and 10°C generate a warning:

```
updu-100499> configure
updu-100499(config)# object Sensor1
updu-100499(config-object-Sensor1)# rules
updu-100499(config-object-Sensor1-rules)# temperature 5.0 none none 29.0
updu-100499(config)# object Sensor2
updu-100499(config-object-Sensor2)# rules
updu-100499(config-object-Sensor2-rules)# temperature 5.0 10.0 25.0 29.0
```

7.1.5 Relative Humidity Monitoring

Relative Humidity monitoring is enabled by defining RH% thresholds:

```
relative-humidity <critical-low> <warning-low> <warning-high> <critical-high>
```

Thresholds may be omitted by entering `none`.

Example:

- On Sensor1 and Sensor2, a relative humidity above 65% and below 35% is considered critical. On Sensor2, a relative humidity between 60% and 65% or between 35% and 40% generates a warning:

```
updu-100499> configure
updu-100499(config)# object Sensor1
updu-100499(config-object-Sensor1)# rules
updu-100499(config-object-Sensor1-rules)# relative-humidity 35.0 none none 65.0
updu-100499(config)# object Sensor2
updu-100499(config-object-Sensor2)# rules
updu-100499(config-object-Sensor2-rules)# relative-humidity 35.0 40.0 60.0 65.0
```

7.1.6 Differential Pressure Monitoring

Differential pressure monitoring is enabled by defining pressure thresholds in Pascal (Pa):

```
diff-pressure <critical-low> <warning-low> <warning-high> <critical-high>
```

Thresholds may be omitted by entering `none`.

Example: Differential pressure values from Sensor1 will... * result in a critical condition for values below 20 Pa * result in a warning condition for values below 40 Pa

```
updu-100499> configure
updu-100499(config)# object Sensor1
updu-100499(config-object-Sensor1)# rules
updu-100499(config-object-Sensor1-rules)# diff-pressure 20 40 none none
```

7.1.7 Enable Monitoring Rules

The `enable` command allows to enable monitoring rules on an object.

Example:

- Enable all rules on WireL1, enable current monitoring on WireL2:

```
updu-100499> configure
updu-100499(config)# object WireL1
updu-100499(config-object-WireL1)# rules
updu-100499(config-object-WireL1-rules)# enable all
updu-100499(config)# object WireL2
updu-100499(config-object-WireL2)# rules
updu-100499(config-object-WireL2-rules)# enable current
```

7.1.8 Disable Monitoring Rules

The `disable` command allows to disable monitoring rules on an object.

Example:

- Disable all rules on WireL1, disable current monitoring on WireL2:

```
updu-100499> configure
updu-100499(config)# object WireL1
updu-100499(config-object-WireL1)# rules
updu-100499(config-object-WireL1-rules)# disable all
updu-100499(config)# object WireL2
updu-100499(config-object-WireL2)# rules
updu-100499(config-object-WireL2-rules)# disable current
```

7.1.9 Delete Monitoring Rules

The `delete` command allows to delete monitoring rules on an object.

Example:

- Delete all rules on WireL1, delete current monitoring on WireL2:

```
updu-100499> configure
updu-100499(config)# object WireL1
updu-100499(config-object-WireL1)# rules
updu-100499(config-object-WireL1-rules)# delete all
updu-100499(config)# object WireL2
updu-100499(config-object-WireL2)# rules
updu-100499(config-object-WireL2-rules)# delete current
```

7.1.10 Show Monitoring Rules

The configured monitoring rules and their state can be displayed with the `show rules` command:

```
updu-100499> show rules
Object     Metric                 CritLow  WarnLow WarnHigh CritHigh State
PDU        Voltage                   none     none    230.0    290.0 WARNING HIGH
PDU        System Power              none     none     none     none OK
WireL1     Current                    1.0      3.0     12.0     14.0 OK
WireL2     Current                    1.0      3.0     12.0     14.0 WARNING HIGH
WireL3     Current                    1.0      3.0     12.0     14.0 CRITICAL HIGH
Sensor1    Temperature               12.0     15.0     25.5     28.0 OK
```

7.1.11 Show Monitoring Conditions

To see active conditions, use the `show conditions` command:

```
updu-100499> show conditions
Start               Object          Metric               Status
2023-03-17 14:33:23 PDU             Voltage              WARNING (>235.0V)
```

Likewise, past conditions can be seen with the `show conditions history` command:

```
updu-100499> show conditions history
Start               End                 Object          Metric               Status
2023-03-17 14:33:23 2023-03-17 14:40:10 PDU             Voltage              WARNING (>235.0V)
```

7.2 Log Messages

The logging system constitutes an important component of the UPDU firmware. For noteworthy events, messages are logged into an internal log buffer which can be obtained via the CLI or the web interface.

The log buffer is not persistent across reboots of the UPDU. The space in the log buffer is limited. Once the buffer is full, old messages are dropped from the buffer as new messages are logged.

When configured, messages are also sent to a remote server using the syslog protocol (see *Set Syslog Server*) the moment they are logged. Should the network not be ready yet (e.g. after reboot), all messages in the buffer are sent once the network interface is up.

7.2.1 Log Levels and Log Facilities

Every log message is composed of a timestamp, a log level indicating the severity of the event, a log facility representing the component of the firmware where the event happened, as well as a textual message:

```
<TIMESTAMP> <LOG-LEVEL> <FACILITY>: <MESSAGE>
```

All messages are logged with one of the following four log levels, sorted by their severity:

- `err`: Errors

- `wrn` : Warnings
- `inf` : Informational messages
- `dbg` : Debug messages (hidden by default)

When logging to a remote syslog server, the internal log level is translated into the respective level of the syslog protocol (`LOG_ERR` , `LOG_WARNING` , `LOG_INFO` and `LOG_DEBUG`).

Most messages logged are of informational nature.

Error and warning level messages can point to a temporary problem, a usage or configuration problem, but also a malfunction of the device or an internal firmware error.

Debug messages are disabled by default. For some facilities, these can be activated with the `trace` CLI command (see *Tracing*).

The two most important log facilities from a user perspective are the `monitoring` and the `audit` facility:

- The `monitoring` facility logs messages when the state of a user configured rule changes (e.g. current drawn above configured threshold).
- The `audit` facility logs message when user initiated or certain external events occur (e.g. an outlet is switched or when the PDU is rebooted).

7.2.2 Monitoring Messages

When a user-configured monitoring rule changes its state, a message is logged by the `monitoring` facility telling what happened to which object and which metric.

All monitoring messages are informational messages and are therefore logged with the `inf` log level, regardless of the monitoring event.

Monitoring messages have the following format:

```
<OBJECT> (<METRIC>): <STATUS> (<INFO>)`
```

Message components are:

- `<OBJECT>` is the name of the object in question, like e.g. `PDU` , `Branch1` or `Outlet2.1` .
- `<METRIC>` is the metric, which is one of the following:
  - Power measurements metrics: `Current` , `Voltage`
  - RCM metrics: `Residual Current RMS` , `Residual Current DC`
  - Sensor metrics: `Temperature` , `Relative Humidity`
  - Over-voltage protection: `OVP fitness`
  - System power source (A/C or PoE-powered): `System Power`
- `<STATUS>` is the status of the rule and depends on the actual context:
  - `OK` : The measurement value is back within the expected (good) range.
  - `CRITICAL LOW` : The measurement value is below the *critical-low* threshold.
  - `WARNING LOW` : The measurement value is below the *warning-low* threshold.
  - `WARNING HIGH` : The measurement value is above the *warning-high* threshold.
  - `CRITICAL HIGH` : The measurement value is above the *critical-high* threshold.
  - `FAULT` : A fault has occurred which requires manual intervention to restore good working condition. This is e.g. logged for the `OVP fitness` metric, to indicate that the the over-voltage protection has tripped.
  - `FAILOVER` : Some component is in a failover state. This is used when the system is powered by Power over Ethernet instead of A/C (`System Power` metric).
  - `UNKNOWN` : The measurement value is not known. This means that the PDU is unable to obtain a measurement value required to check configured monitoring thresholds. The reason for this can be intermittent communication issue, a missing external sensor or a hardware defect.

- `<INFO>` is optionally logged if additional information about the event is available. It typically contains the measurement value that crossed a threshold, as well as the threshold itself: e.g. `(231.5V>230.0V)` if the measured voltage was 231.5 volts which is above the configured threshold of 230.0 volts.

Examples of monitoring messages:

- The warning high threshold of 230.0 volts on Outlet1.4 was crossed:

```
Outlet1.4 (Voltage): WARNING HIGH (231.5V>230.0V)
```

- The warning low threshold of 4.0 amperes on the entire PDU was crossed:

```
PDU (Current): WARNING LOW (3.5A<4.0A)
```

- The over-voltage protection module connected to the Inlet has failed:

```
Inlet (OVP fitness): CRITICAL FAULT
```

- The temperature cannot be read from the temperature sensor in the AUX2 port (e.g. because there is no sensor plugged in):

```
Sensor2 (Temperature): UNKNOWN
```

- The ICM has lost A/C power and is powered over Power over Ethernet (PoE):

```
PDU (System Power): WARNING FAILOVER (on PoE power)
```

7.2.3 Audit Messages

For certain important events, mostly user-initiated, a message is logged with the `audit` log facility. Audit messages are informational and logged with the `inf` log level, regardless of what happened.

Audit messages can have different formats and always start with initiator context:

- System-initiated action. Currently limited to `System booted`, which is logged after the PDU has booted and is ready for operation:

```
System <OPERATION>
```

- Operation initiated via the SNMP interface:

```
SNMP <OPERATION>
```

- Operation initiated by `<USERNAME>` via the CLI:

```
CLI user <USERNAME> <OPERATION>
```

- Operation initiated by `<USERNAME>` via the Web interface:

```
WEB user <USERNAME> <OPERATION>
```

The following `<OPERATION>'s` are logged via audit messages:

- Settings are reset to factory defaults: `factory-resets`
- The PDU is rebooted: `reboots PDU`
- A firmware upgrade is initiated: `initiates firmware upgrade`
- An outlet is switched on or off: `switches <OUTLET> [on|off]`
- An outlet is power cycled: `power-cycles <OUTLET>`
- Debug logging is enabled or disabled for a facility: `[enables|disables] tracing <FACILITY>`

Examples of audit messages:

- Outlet1.6 was switched off with an SNMP request:

```
SNMP switches Outlet1.6 off
```

- User `john` used the web interface to switch Outlet1.2 on:

```
WEB user john switches Outlet1.2 on
```

- User `alice` has entered the `reboot` command in a CLI.

```
CLI user alice reboots PDU
```

7.3 Local Logging

7.3.1 Show System Log

The `show log` command shows content of the system log buffer. Messages logged by the system are written into that buffer. If the buffer is full, old messages are rotated out of the buffer.

Log messages are formatted as follows:

```
[timestamp] level facility: message
```

By default, the timestamp is the uptime of the device when the message was logged.

- Using the `show log utc` command, the timestamp is converted to UTC.
- Using the `show log localtime` command, the timestamp is converted to the configured local time

(requires the clock to be synchronized, otherwise the system assumes the device has booted at 1/1/1970).

The other components are:

- `level`: One of `err` (error), `wrn` (warning), `inf` (information) or `dbg` (debugging).
- `facility`: Tells which part of the system logged the message.

7.3.2 Clear System Log

The log is cleared using the `clear log` command.

7.3.3 Real-Time Log

To activate a real-time log of the syslog messages into the current CLI session, use the `monitor log` command:

```
updu-100499> monitor log
Real-time log monitoring enabled
...
```

All logged messages are now printed in the current CLI session in real-time. To disable the real-time log, use `monitor log off`:

```
updu-100499> monitor log
Real-time log monitoring disabled
...
```

7.3.4 Show Bootloader Log

The `show log bootloader` command shows the log messages written by the bootloader. The format of the messages is similar to the system log messages, with the only difference that the timestamp is given as `[----------]`.

7.4 Syslog Logging

7.4.1 Configuration

To send syslog messages to `192.168.1.1`:

```
updu-100499> configure
updu-100499(config)# logging
updu-100499(config-logging)# syslog enabled
updu-100499(config-logging)# syslog server 192.168.1.1
updu-100499(config-logging)#
```

By default, messages are sent with the syslog facility `local0`. This can be changed as follows:

```
updu-100499> configure
updu-100499(config)# logging
updu-100499(config-logging)# syslog enabled
updu-100499(config-logging)# syslog server 192.168.1.1 facility local3
updu-100499(config-logging)#
```

By default, messages are sent to port 514 (UDP). This can be changed as follows:

```
updu-100499> configure
updu-100499(config)# logging
updu-100499(config-logging)# syslog enabled
updu-100499(config-logging)# syslog server 192.168.1.1 port 1514
updu-100499(config-logging)#
```

Syslog messages are sent with UTC timestamps by default. In order to use local timestamps, use the `syslog time` configuration command.

Examples:

- Use UTC timestamps:

```
updu-100499(config-logging)# syslog time utc
```

- Use local timestamps:

```
updu-100499(config-logging)# syslog time local
```

7.4.2 Disable

To disable syslog:

```
updu-100499> configure
updu-100499(config)# logging
updu-100499(config-logging)# syslog disabled
updu-100499(config-logging)#
```

7.5 Email Logging

To setup Email notifications sent by the UPDU, refer to the *SMTP Configuration* section.

8 Networking

8.1 Network Interface Configuration

The `interface` configuration sub-mode can be used to modify interface settings.

8.1.1 Link Configuration

The UPDU features three network interface:

ETH1 and *ETH2* are switched (or bridged), thus act as a single interface above layer two. The physical ports (link) can be individually enabled or disabled, but any higher level configuration (IP, ..) is always applied to both ports.

ETH3 is a dedicated interface and can be independently configured.

The following network interface link settings are available:

- `enabled` : The interface is active. This setting is available for ETH1 and ETH3.
- `disabled` : The interface is shut down.
- `bridged-to-eth1` : The interface is bridged to ETH1. This setting is only available for ETH2.

Examples:

- Disable an interface:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# link disabled
updu-100499(config-interface-ETH1)#
```

- Enable an interface:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# link enabled
updu-100499(config-interface-ETH1)#
```

- Bridge ETH2 to ETH1:

```
updu-100499> configure
updu-100499(config)# interface ETH2
updu-100499(config-interface-ETH2)# link bridged-to-eth1
updu-100499(config-interface-ETH2)#
```

8.1.2 IPv4 Configuration

Network interfaces can be configured to use a static IP address or to use DHCP to get a dynamic IP address.

Examples:

- Disable IPv4:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv4 disabled
updu-100499(config-interface-ETH1)#
```

- Static IP address, netmask and gateway:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv4 address 192.168.1.100/24 gateway 192.168.1.1
updu-100499(config-interface-ETH1)#
```

- Use DHCP, fall-back to AutoIP after 30 seconds (default setting):

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv4 dhcp autoip-fallback 30
updu-100499(config-interface-ETH1)#
```

- Use DHCP, disable AutoIP fall-back:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv4 dhcp autoip-fallback off
updu-100499(config-interface-ETH1)#
```

8.1.3 IPv4 DNS Configuration

For each network interface, up to two IPv4 DNS servers can be configured:

- No IPv4 DNS servers:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv4 dns-server none
updu-100499(config-interface-ETH1)#
```

- Automatically configure IPv4 DNS servers:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv4 dns-server auto
updu-100499(config-interface-ETH1)#
```

- Set two IPv4 DNS servers:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv4 dns-server 8.8.8.8 8.8.4.4
updu-100499(config-interface-ETH1)#
```

8.1.4 DNS Lookups

DNS Lookups are done using the DNS servers configured on the default outbound network interface (see Default Outbound Interface).

If the default outbound network interface has IPv6 active, the UPDU first does IPv6 lookups. If that lookup fails, it falls back to IPv4 lookups.

8.1.5 IPv6 Configuration

The following IPv6 configuration options are available:

- Disable IPv6:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 disabled
updu-100499(config-interface-ETH1)#
```

- Stateless address autoconfiguration (SLAAC):

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 slaac
updu-100499(config-interface-ETH1)#
```

- DHCPv6: Use stateful DHCPv6 to get an IPv6 address and discover the network prefix and default gateway automatically:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 dhcpv6
updu-100499(config-interface-ETH1)#
```

- Static IPv6 address and gateway:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 address fc00:0:0:1::1234/64 gateway fc00:0:0:1::
updu-100499(config-interface-ETH1)#
```

- Static IPv6 with automatic gateway discovery:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 address fc00:0:0:1::1234/64
updu-100499(config-interface-ETH1)#
```

If the prefix length is statically configured as `/0`, the prefix will also be discovered automatically.

8.1.6 IPv6 DNS Configuration

For each network interface, up to two IPv6 DNS servers can be configured:

- No IPv6 DNS servers:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 dns-server none
updu-100499(config-interface-ETH1)#
```

- Automatically configure IPv6 DNS servers:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 dns-server auto
updu-100499(config-interface-ETH1)#
```

- Set up to two IPv6 DNS servers:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# ipv6 dns-server 2001:4860:4860::8888 2001:4860:4860::8844
updu-100499(config-interface-ETH1)#
```

See DNS Lookups for information on how DNS lookups are done.

8.1.7 Apply ACLs to Network Interfaces

An ACL controlling inbound network traffic can be applied to an interface using the `acl in` command:

- Apply ACL `acl-3` to ETH1:

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# acl in acl-3
updu-100499(config-interface-ETH1)#
```

- Remove ACL from ETH1

```
updu-100499> configure
updu-100499(config)# interface ETH1
updu-100499(config-interface-ETH1)# acl in none
updu-100499(config-interface-ETH1)#
```

See *Access Control Lists (ACL)* on how to configure ACLs.

8.1.8 Override DHCP Hostname

Both the IPv4 and IPv6 DHCP clients use the system hostname with their requests. That hostname can be overridden per interface.

Example: Configure the system hostname `myupdu` to be used for DHCP requests from ETH1/ETH2 and set the hostname `myupdu-eth3` for DHCP requests sent from ETH3:

```
updu-100499> configure
updu-100499(config)# system
updu-100499(config-system)# hostname myupdu
myupdu(config-system)# exit
myupdu(config)# interface eth1
myupdu(config-interface-ETH1)# dhcp system-hostname
myupdu(config-interface-ETH1)# interface eth3
myupdu(config-interface-ETH3)# dhcp hostname myupdu-eth3
myupdu(config-interface-ETH3)#
```

8.1.9 Show Network Interface Information

The `show interface` command shows the status of the UPDU's network interfaces:

```
updu-100499> show interface
ETH1
  link up
  mac-address d4:66:a8:10:09:1e
ETH2
  link down
  mac-address d4:66:a8:10:09:1e
ETH3
  link down
  mac-address d4:66:a8:10:09:1d
```

8.2 General Network Configuration

The `network` configuration sub-mode can be used to modify general network settings.

8.2.1 Show IP Information

The `show ip` command shows the current TCP/IP setup:

```
updu-100499> show ip
ETH1/2
  ipv4 address 192.168.111.28/24
  ipv4 gateway 192.168.111.1
  ipv4 dns-server 192.168.111.1
  ipv6 link-local fe80::d666:a8ff:fe10:1123
  ipv6 global fc00:0:0:1::3000
  ipv6 prefix fc00:0:0:1::/64
  ipv6 gateway fe80::250:b6ff:fe14:2de7
  ipv6 dns-server fc00:0:0:3::
ETH3
  ipv4 address n/a
  ipv6 link-local n/a
  ipv6 global n/a
  ipv6 prefix n/a
  ipv6 gateway n/a
```

8.2.2 Default Outbound Interface

The interface to be used for outbound traffic can be configured as follows:

- Use ETH1 and ETH2 (if ETH2 is bridged to ETH1):

```
updu-100499> configure
updu-100499(config)# network
updu-100499(config-network)# default-outbound ETH1/ETH2
updu-100499(config-network)#
```

- Use ETH3:

```
updu-100499> configure
updu-100499(config)# network
updu-100499(config-network)# default-outbound ETH3
updu-100499(config-network)#
```

8.2.3 Clear DNS Cache

The DNS cache of the default outbound interface can be cleared using the `clear dns-cache` CLI command.

8.2.4 Broadcast and Multicast Storm Protection

The storm protection feature limits the bandwidth of broadcast and multicast traffic on the switch which backs the ETH1 and ETH2 interfaces. When enabled, broadcast, and optionally, multicast traffic exceeding 2% of the available link bandwidth is dropped as it enters ETH1 and ETH2.

This feature helps to mitigate problems caused by switching loops, which use up CPU resources of UPDUs and can cause UDPUs to become unreachable.

Broadcast and multicast storm protection is enabled by default. It can be enabled or disabled in the `network` section of the configuration:

- Switch off storm protection:

```
updu-100499> configure
updu-100499(config)# network
updu-100499(config-network)# switch storm-protection disabled
updu-100499(config-network)#
```

- Enable storm protection for broadcast traffic only:

```
updu-100499> configure
updu-100499(config)# network
updu-100499(config-network)# switch storm-protection broadcast
updu-100499(config-network)#
```

- Enable storm protection for broadcast and multicast traffic:

```
updu-100499> configure
updu-100499(config)# network
updu-100499(config-network)# switch storm-protection broadcast multicast
updu-100499(config-network)#
```

8.3 Access Control Lists

Access Control Lists (ACL) are a simple mechanism to manage access to network services of a UPDU.

An ACL is a sequence of rules each defining a condition for accepting or dropping network traffic. When an ACL is configured on a network interface, every incoming data packet on the interface is compared the rules in the ACL in the configured order, starting with the first entry. The first matching rule defines what happens with the packet: If it is a `drop` rule, the packet is dropped which will deny access to a service. On the other side, when an `accept` rule matches, access is granted.

The default policy of an ACL is to accept traffic. Thus if no rule matches, access is granted. The same is happens when an empty ACL is applied to an interface.

A rule is composed by the following characteristics:

- Action: `accept` or `drop`
- Protocol: IP, TCP, UDP, ICMP
- Source address range
- Destination address range
- For TCP and UDP: Source and destination port range
- For ICMP: A range of ICMP packet types

8.3.1 ACL Configuration

The UPDU supports up to four independent ACLs which can be applied to network interfaces. ACLs are named `acl-1` through `acl-4`.

ACLs are configured with the `acl` command:

- Configure the ACL `acl-1`:

```
updu-100499> configure
updu-100499(config)# acl acl-1
updu-100499(config-acl-1)#
```

- Set a description for this ACL:

```
updu-100499(config-acl-1)# description "Restrict management"
```

IPv4 and IPv6 rules are configured in two separate lists. To enter the rule configuration, use either the `ipv4-rules` or the `ipv6-rules` commmand in `append` or `replace` mode. In `append` mode, the currently configured rules kept and new rules are appended at the end of the list. In `replace` mode, on the other hand, existing rules are deleted:

- Enter the IPv4 rules configuration while deleting all existing rules:

```
updu-100499(config-acl-1)# ipv4-rules replace
updu-100499(config-acl-1-ipv4)#
```

- Enter the IPv6 rules configuration in order to append new rules:

```
updu-100499(config-acl-1)# ipv6-rules append
updu-100499(config-acl-1-ipv6)#
```

Rules Configuration

Each ACL rule contains a rule type, protocol, source, destination and other protocol-specific details.

The rule type is one of:

- `accept` : Defines traffic which is accepted
- `drop` : Defines traffic which is dropped

The following protocols are supported:

- `ip` : Matches any IP traffic (including ICMP, UDP and TCP)
- `tcp` : Matches TCP traffic
- `udp` : Matches UDP traffic
- `icmp` : Matches ICMP and ICMPV6 traffic, depending on whether the rule is in the IPv4 or IPv6 list

The source and destination are given as IP address ranges in address/prefix-length notation. The keyword `any` is a short form for `0.0.0.0/0` (IPv4) and `::/0` (IPv6) matching all IP addresses:

- `any` : Matches all IP addresses
- `192.168.1.0/24` : Match IPv4 addresses in 192.168.1.0-192.168.1.255
- `fe80::/64` : Match all Link-Local IPv6 addresses

For the `tcp` and `udp` protocols, the source and destination may optionally be followed by a port or a port range:

- `port 443` : Match port 443
- `port 0 to 1023` : Match ports 0 to 1023 (also called privileged ports)

For the `icmp` protocol, the ICMP packet type or a range of types can be optionally specified:

- `type 8` : Matches the ICMP "Echo Request" packet
- `type 128` : Matches the ICMPV6 "Echo Request" packet
- `type 135 to 136` : Matches the ICMPV6 Neighbor Solicitation and Advertisement packets

Examples for complete rules:

- Reject all IP traffic:

```
updu-100499(config-acl-1-ipv4)# drop ip any any
```

- Grant access to the built-in webserver for a selected network (HTTPS):

```
updu-100499(config-acl-1-ipv4)# accept tcp 192.168.100.0/24 any port 443
```

- Grant access to Modbus/TCP for a selected network:

```
updu-100499(config-acl-1-ipv4)# accept tcp 192.168.100.0/24 any port 502
```

- Allow ICMPV6 Neighbor Discovery to work:

```
updu-100499(config-acl-1-ipv6)# accept icmp any any type 135 to 136
```

Complete ACLs

Example: Restrict access to SSH (TCP port 22) and SNMP (UDP port 161) from management network 10.1.0.0/24:

```
updu-100499(config)# acl acl-1
updu-100499(config-acl-1)# ipv4-rules replace
updu-100499(config-acl-1-ipv4)# accept tcp 10.1.0.0/24 any port 22
updu-100499(config-acl-1-ipv4)# drop tcp any any port 22
updu-100499(config-acl-1-ipv4)# accept udp 10.1.0.0/24 any port 161
updu-100499(config-acl-1-ipv4)# drop udp any any port 161
updu-100499(config-acl-1-ipv4)# exit
updu-100499(config-acl-1)# ipv6-rules replace
updu-100499(config-acl-1-ipv6)# drop tcp any any port 22
updu-100499(config-acl-1-ipv6)# drop udp any any port 161
updu-100499(config-acl-1-ipv6)# exit
```

Example: Grant access to Modbus/TCP to one single client with IP 192.168.123.234:

```
updu-100499(config)# acl acl-1
updu-100499(config-acl-1)# ipv4-rules replace
updu-100499(config-acl-1-ipv4)# accept tcp 192.168.123.234/32 any port 502
updu-100499(config-acl-1-ipv4)# drop tcp any any port 502
updu-100499(config-acl-1-ipv4)# exit
updu-100499(config-acl-1)# ipv6-rules replace
updu-100499(config-acl-1-ipv6)# drop tcp any any port 502
updu-100499(config-acl-1-ipv6)# exit
```

Example: Restrict TCP/IP traffic to a management network at 10.3.2.0/24 and 2001:0DB8:1234::/48 and block everything else. For IPv6, Neighbor Discovery and SLAAC is accepted:

```
updu-100499(config)# acl acl-1
updu-100499(config-acl-1)# ipv4-rules replace
updu-100499(config-acl-1-ipv4)# accept ip 10.3.2.0/24 any
updu-100499(config-acl-1-ipv4)# drop ip any any
updu-100499(config-acl-1-ipv4)# exit
updu-100499(config-acl-1)# ipv6-rules replace
updu-100499(config-acl-1-ipv6)# accept ip 2001:0db8:1234::/48 any
updu-100499(config-acl-1-ipv6)# accept icmp any any type 133 to 136
updu-100499(config-acl-1-ipv6)# drop ip any any
updu-100499(config-acl-1-ipv6)# exit
```

Notes on ACLs

- Destination IP address: Since ACLs are currently only used to filter incoming network traffic, specifying the destination IP address as `any` will in most cases be perfectly fine.
- DHCP traffic: To allow the DHCP client to receive DHCP server responses, accept UDP traffic to port 68:

```
accept udp any any port 68
```

- IPv6 neighbor discovery: Note that IPv6 requires the neighbor discovery protocol to work. This protocol is based in ICMPV6 messages, so if you plan to restrict ICMPV6 traffic, you have to explicitly accept the neighbor discovery protocol messages:

```
accept icmp any any type 135 to 136
drop icmp any any
```

8.3.2 Clear ACL Description and Rules

An ACL is cleared using the `clear` command in the ACL configuration sub-mode. This will clear all rules as well as the description. If the ACL is currently active on an interface, all traffic will be accepted:

```
updu-100499(config)# acl acl-2
updu-100499(config-acl-2)# clear
```

8.4 Spanning Tree Protocol (STP) Configuration

The `spanning-tree` configuration sub-mode can be used to modify spanning tree protocol settings. The spanning tree protocol is used on ETH1 and ETH2.

8.4.1 Enable or Disable STP

- Enable STP:

```
updu-100499> configure
updu-100499(config)# spanning-tree
updu-100499(config-stp)# enabled
updu-100499(config-stp)#
```

- Disable STP:

```
updu-100499> configure
updu-100499(config)# spanning-tree
updu-100499(config-stp)# disabled
updu-100499(config-stp)#
```

8.4.2 Select STP Bridge Priority

The STP bridge priority can be configured using the `bridge-priority` command.

Examples:

- Set bridge priority to 0:

```
updu-100499> configure
updu-100499(config)# spanning-tree
updu-100499(config-stp)# bridge-priority 0
updu-100499(config-stp)#
```

- Set bridge priority to 32768 (default value):

```
updu-100499> configure
updu-100499(config)# spanning-tree
updu-100499(config-stp)# bridge-priority 32768
updu-100499(config-stp)#
```

8.4.3 Set STP Timers

The STP hello time, max-age and forward delay timers can be set using the `stp-timers` command:

Example:

- Set timers to the default values

```
updu-100499> configure
updu-100499(config)# spanning-tree
updu-100499(config-stp)# stp-timers hello-time 2 max-age 20 forward-delay 15
updu-100499(config-stp)#
```

8.4.4 Select STP Version

The UPDU supports both the traditional Spanning Tree Protocol (802.1d) well as the Rapid Spanning Tree Protocol (802.1w).

- Select the traditional Spanning Tree Protocol:

```
updu-100499> configure
updu-100499(config)# spanning-tree
updu-100499(config-stp)# version STP
updu-100499(config-stp)#
```

- Select the Rapid Spanning Tree Protocol:

```
updu-100499> configure
updu-100499(config)# spanning-tree
updu-100499(config-stp)# version RSTP
updu-100499(config-stp)#
```

Note that if you use the traditional Spanning Tree Protocol together with DHCP, the default `autoip-fallback` of 30 seconds may need to be increased. See IPv4 Configuration.

8.4.5 Show Spanning Tree Protocol Status

The `show spanning-tree` command shows the current STP status:

```
updu-100499> show spanning-tree
Version: RSTP

Bridge ID: 32768-d4:66:a8:10:09:1e
Bridge hello time: 2
Bridge max age: 20
Bridge forward delay: 15

Topology changes count: 0
Root bridge ID: 32768-d4:66:a8:10:09:1e

Interface  Role       Status
ETH1       designated forwarding
ETH2       disabled   broken
```

9 Services

9.1 Simple Network Management Protocol (SNMP) Configuration

The `snmp` configuration context is used to modify all SNMP related settings.

9.1.1 Enable or Disable SNMPv2

SNMP version 2 is enabled or disabled as follows:

- Enable SNMPv2:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv2 enabled
updu-100499(config-snmp)#
```

- Disable SNMPv2:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv2 disabled
updu-100499(config-snmp)#
```

9.1.2 Configure SNMPv2 Read Access

In order to enable SNMPv2 read access, a community string has to be configured. This is done as follows:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv2 read enabled community very-secret-abc*123
updu-100499(config-snmp)#
```

SNMPv2 read access is disabled as follows:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv2 read disabled
updu-100499(config-snmp)#
```

9.1.3 Configure SNMPv2 Write Access

In order to enable SNMPv2 write access, a community string has to be configured. This is done as follows:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv2 write enabled community very-secret-abc*123
updu-100499(config-snmp)#
```

SNMPv2 write access is disabled as follows:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv2 write disabled
updu-100499(config-snmp)#
```

9.1.4 SNMP Version 3

SNMP version 3 can be enabled or disabled as follows:

- Enable SNMPv3:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv3 enabled
updu-100499(config-snmp)#
```

- Disable SNMPv3:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# snmpv3 disabled
updu-100499(config-snmp)#
```

When SNMPv3 is enabled, all users which are enabled for SNMPv3 have access according to their user role.

9.1.5 Set SNMP SysContact

The MIB-II System Group contact is configured as follows:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# syscontact system-admin@best-datacenter.domain
updu-100499(config-snmp)#
```

9.1.6 Set SNMP SysLocation

The MIB-II System Group location is configured as follows:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# syslocation "Cabinet 1, Row 4, Floor 2"
updu-100499(config-snmp)#
```

9.1.7 Enable or Disable SNMP MIBs

It is possible to enable or disable MIBs with the `mib` command. The following MIBs are available:

- `updu-mib2`: The UPDU MIB2, defined in the file `RNX-UPDU-MIB2.mib`. This is the latest, recommended MIB and is enabled by default.
- `updu-mib1`: The UPDU MIB1, defined in the file `RNX-UPDU-MIB1.mib`. This MIB is obsolete and not recommended to use for new installations. It is enabled by default.
- `e3meter-mib`: A subset of the RNX E3METER IPS PDUs, making it possible to monitor the PDU with existing monitoring software. Described in `e3meter-ipm-stripped.mib`. This MIB is disabled by default.

Note that the actual `.mib` files are available from the firmware archive and can be downloaded from the UPDU's web interface next to the SNMP configuration section.

Examples:

- Disable the `updu-mib1`:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# mib updu-mib1 disabled
updu-100499(config-snmp)#
```

- Enable the `updu-mib1`:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# mib updu-mib1 enabled
updu-100499(config-snmp)#
```

9.1.8 Configure SNMP Notifications

Notifications can be sent to up to three SNMP receivers using SNMP Trap or Inform messages. Notifications are sent for events from the monitoring system and for audit events. Thus, in order to receive SNMP notifications, refer to *UPDU Object Configuration* for details on how to configure monitoring settings.

Notes:

- Inform messages require the receiver to acknowledge reception. If not acknowledged within two seconds the UPDU tries to re-send the messages up to five times.
- In order for notifications to be sent, the `updu-mib2` MIB and the corresponding SNMP version needs to be enabled. For SNMPv3, the mentioned user must be configured.
- For SNMPv3 Traps, the UPDU's SNMP Engine ID has to be configured at the receiver.
- Only a single notification setting can be enabled per host address. This is because the host address is used as uniqe identifier for the notification. **An existing configuration for the same host address is replaced without confirmation**.

Examples:

- Send SNMPv2 Trap messages to 192.168.1.1 with community `alert`:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# notify-host 192.168.1.1 snmpv2-trap community alert
```

- Send authenticated and encrypted SNMPv3 Inform messages to `192.168.1.123` with the user `event`. Note that a system user has to be created in order to use the SNMPv3 authentication against a host.

```
updu-100499> configure
updu-100499(config)# users
updu-100499(config-users)# user event
Created new user event.
updu-100499(config-users-event)# snmpv3 enabled
updu-100499(config-users-event)# snmpv3 auth sha1 password yes-its-me
updu-100499(config-users-event)# snmpv3 privacy aes password very-secret
updu-100499(config-users-event)# exit
updu-100499(config-users)# exit
updu-100499(config)# snmp
updu-100499(config-snmp)# notify-host 192.168.1.123 snmpv3-inform user event
```

To delete a notification receiver, use the `delete` command:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# delete notify-host 192.168.1.1
```

9.1.9 Enable or Disable SNMP Options

The following SNMP functions can be controlled using the `option` command in the SNMP configuration mode:

- `write-object-info`: When enabled, properly authenticated SNMP write requests (i.e. carrying the correct SNMPv2 write community or matching SNMPv3 username/authentication/privacy information) can be used to modify object name and description, which are exposed as `upduMeterCustomName` and `upduMeterDescription` in the RNX-UPDU MIB. This functionality is disabled by default. It is only shown in the `show config` output if enabled.

  When disabled, SNMP write requests can only be used to switch outlets on and off.

Examples:

- Enable `write-object-info`:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# option write-object-info enabled
updu-100499(config-snmp)#
```

- Disable `write-object-info`:

```
updu-100499> configure
updu-100499(config)# snmp
updu-100499(config-snmp)# option write-object-info disabled
updu-100499(config-snmp)#
```

9.1.10 Show SNMP EngineID

The `show snmp engineid` command shows the UPDU's unique EngineID:

```
updu-100499> show snmp engineid
8000d74403d466a810091d
```

9.2 Modbus/TCP Configuration

9.2.1 Enable or Disable Modbus/TCP

The Modbus/TCP service can be configured using the `modbus/tcp` configuration sub-mode.

- Disable the Modbus/TCP service:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# disabled
updu-100499(config-modbus/tcp)#
```

- Enable the Modbus/TCP service:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# enabled
updu-100499(config-modbus/tcp)#
```

9.2.2 Modbus/TCP Transport

The Modbus/TCP server supports two transport modes:

- `plain`: Connections without encryption or authentication

- `tls` : Modbus/TCP over TLS (also known as Modbus Security or MBAPS)

With `plain` transport mode, all connections use the configured `default-role` (see *Modbus/TCP Authorization*). It is not recommended to use `plain` transport mode except in fully isolated networks. In `plain` mode the server listens on port 502.

With `tls` transport mode, all communication with the Modbus/TCP server is encrypted. The server identifies itself using the certificate configured with the `certificate` command (see *Configure TLS Server Certificate*, allowing clients to make sure they are communicating with the correct server. In `tls` mode, the server listens on port 802.

If client authentication is configured (see *Configure Client Authentication*), clients have to identify themselves with a certificate signed with one of the certificates configured in `auth-certificate`. Other connections are refused. Depending on the `default-role` setting (see *Modbus/TCP Authorization*, client certificates have to specify the role of the client in an *x.509v3 certificate extension* (*OID Role: 1.3.6.1.4.1.50316.802.1*, refer to section 8.4 within the *Modbus/TCP Security Protocol Specification V36*).

The transport is configured with the `transport` command:

- Configure plain Modbus/TCP:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# transport plain
```

- Configure Modbus/TCP over TLS:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# transport tls
```

9.2.3 Configure TLS Server Certificate

The Modbus/TCP server can either use the built-in `updu.io` wildcard certificate or a custom certificate uploaded to the UPDU (see *SSL/TLS Certificates Configuration*).

The certificate is selected using the `certificate` command in the `modbus/tcp` configuration sub-mode.

Examples:

- Use the built-in `*.updu.io` wildcard certificate:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# certificate built-in
```

- Use the certificate configured in certificate store 1:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# certificate store-1
```

9.2.4 Configure Client Authentication

To enable client authentication, the list of certificates used for signing the client certificates is stored on the UPDU in one of the certificate stores (see *SSL/TLS Certificates Configuration*).

The store containing the authentication certificates is then configured using the `auth-certificate` command:

- Authenticate client connections using the certificates in `store-1` :

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# auth-certificate store-1
```

- Disable client authentication and accept any client:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# auth-certificate none
```

9.2.5 Modbus/TCP Authorization

Modbus/TCP authorizes connections using roles embedded in x.509 client certificates used for authentication. The role transmitted to the UPDU correspond to the configured roles (see *Roles & Permissions*). In order to read Modbus/TCP registers, the `modbus-read` permission is required.

If no client authentication is configured, the role configured with the `default-role` command is used:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# default-role set guest
```

To make client authorization mandatory (i.e. only grant access to authenticated clients having a role with the `modbus-read` permission in the client certificate), set the default role to `none`:

```
updu-100499> configure
updu-100499(config)# modbus/tcp
updu-100499(config-modbus/tcp)# default-role none
```

The above example is the factory default setting.

9.2.6 Show Modbus/TCP Register Map

The `show modbus/tcp` command shows which Modbus/TCP register ranges correspond to which object. To get the detailed list of registers, use `show modbus/tcp detail`.

9.3 Webserver Configuration

The `webserver` configuration sub-mode can be used to modify the built-in webserver settings.

9.3.1 Enable or Disable HTTP

Enable HTTP:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# http enabled
updu-100499(config-webserver)#
```

Disable HTTP:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# http disabled
updu-100499(config-webserver)#
```

9.3.2 Enable or Disable HTTPS

Enable HTTPS:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# https enabled
updu-100499(config-webserver)#
```

Disable HTTPS:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# https disabled
updu-100499(config-webserver)#
```

9.3.3 Webserver Redirection

The `redirect` configuration directive allows to configure the webserver to redirect requests as follows:

- `redirect http disabled`: No redirection of unencrypted requests (URLs starting with `http://`).
- `redirect https disabled`: No redirection of encrypted requests (URLs starting with `https://`).
- `redirect http https`: All unencrypted requests are redirected to an encrypted `https://` URL. No other unencrypted data is sent by the webserver. This is the factory default. If HTTPS is disabled, this setting has no effect.
- `redirect https updu.io`: All requests to an https URL containing an IP address are redirected to the corresponding URL under the `updu.io` domain name. This setting has no effect if a custom SSL/TLS certificate is active.

Examples:

- Disable all redirections:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# redirect http disabled
updu-100499(config-webserver)# redirect https disabled
```

- Redirect http to https and to the updu.io domain:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# redirect http https
updu-100499(config-webserver)# redirect https updu.io
```

This causes e.g. requests to `http://192.168.1.100` to be redirected to `https://192-168-1-100.updu.io`.

9.3.4 Enable or Disable HTTP Strict Transport Security (HSTS)

When HTTP Strict Transport Security is enabled, clients will no longer communicate with the UPDU over non-encrypted HTTP connections but directly use HTTPS.

HSTS is disabled by default. When enabled, HSTS is used with `max-age=31536000` (one year), subdomains are not included.

Enable HSTS:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# hsts enabled
updu-100499(config-webserver)#
```

Disable HSTS:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# hsts disabled
updu-100499(config-webserver)#
```

9.3.5 Select the SSL/TLS Certificate

By default, the webserver uses a built-in wildcard certificate for the `updu.io` domain. In addition to this static certificate, the webserver can be configured to use a custom certificate uploaded to the PDU (see *SSL/TLS Certificates Configuration*).

The certificate is selected using the `certificate` command in the webserver configuration sub-mode.

Examples:

- Use the built-in `*.updu.io` wildcard certificate:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# certificate built-in
```

- Use the certificate configured in certificate store 1:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# certificate store-1
```

- Use the certificate configured in certificate store 2:

```
updu-100499> configure
updu-100499(config)# webserver
updu-100499(config-webserver)# certificate store-2
```

If the chosen certificate is incomplete (i.e. the private key or the certificate is missing or invalid), the webserver will automatically fall-back to the built-in `updu.io` certificate. As soon as the chosen certificate is corrected, it is activated immediately.

9.3.6 Using the updu.io certificate

As mentioned before, the UPDU comes with an embedded certificate which can be used to access the device securely and without browser warnings. RNX operates a special DNS resolver which allows to use the `updu.io` domain to access the UPDU via HTTPS.

This works as followed: If your UPDU is configured to have the IP address `10.10.0.33`, the hostname to access the UPDU would be `10-10-0-33.updu.io`. For this to work, the client (e.g. PC) which connects to the UPDU must have internet access. The UPDU itself is not required to have internet access becuase the DNS lookup happens on the client.

Given that the `updu.io` certificate is signed and trusted by all major browsers, there will not be any warning message when accessing the web-interface.

9.4 SSL/TLS Certificates Configuration

In addition to the built-in wildcard SSL/TLS certificate, up to four custom certificates, stored in the certificate stores `store-1` to `store-4` can be configured by the user.

To configure a certificate, two text files containing PEM formatted data are needed:

- A private key (e.g. pdu.domain.key), with a format as follows:

```
-----BEGIN PRIVATE KEY-----
<base64 data>
-----END PRIVATE KEY-----
```

- A certificate (e.g. pdu.domain.crt), with a format as follows:

```
-----BEGIN CERTIFICATE-----
<base64 data>
-----END CERTIFICATE-----
```

The certificate can contain intermediate certificates, in which case multiple `BEGIN/END CERTIFICATE` lines are present.

These two data files are configured in the `certificates` configuration sub-mode:

- Enter the `certificates` configuration sub-mode and choose the one of the certificate stores:

```
updu-100499> configure
updu-100499(config)# certificates
updu-100499(config-certificates)# store-1
```

- Optionally enter a certificate description, for documentation purposes:

```
updu-100499(config-certificates)# description "Datacenter team wildcard cert"
```

- Paste the private key file content:

```
updu-100499(config-certificates-store-1)# key-data
Paste private key in PEM format, end with an empty line or CTRL-c or CTRL-d.
# -----BEGIN PRIVATE KEY-----
# ...
# -----END PRIVATE KEY-----
#
```

- Paste the certificate file content:

```
updu-100499(config-certificates-store-1)# crt-data
Paste certificate in PEM format, end with an empty line or CTRL-c or CTRL-d.
# -----BEGIN CERTIFICATE-----
# ...
# -----END CERTIFICATE-----
# -----BEGIN CERTIFICATE-----
# ...
# -----END CERTIFICATE-----
#
updu-100499(config-certificates-store-1)#
```

The `store-1` certificate can now be used by the webserver, see *Select the SSL/TLS Certificate*.

Having multiple certificate stores allows the certificates to be exchanges seamlessly.

9.4.1 Clearing SSL/TLS Certificates

To clear certificate data, use the `clear` command in the certificates store sub-mode:

- Clear everything (description, key, and certificate):

```
updu-100499> configure
updu-100499(config)# certificates
updu-100499(config-certificates)# store-1
updu-100499(config-certificates)-store-1# clear all
```

- Clear description:

```
updu-100499(config-certificates)-store-1# clear description
```

- Clear key:

```
updu-100499(config-certificates)-store-1# clear key
```

- Clear certificate:

```
updu-100499(config-certificates)-store-1# clear certificate
```

9.5 SSH Configuration

The `ssh` configuration sub-mode allows to control the SSH service.

- Enable the SSH server:

```
updu-100499> configure
updu-100499(config)# ssh
updu-100499(config-ssh)# enabled
updu-100499(config-ssh)#
```

- Disable SSH server:

```
updu-100499> configure
updu-100499(config)# ssh
updu-100499(config-ssh)# disabled
updu-100499(config-ssh)#
```

9.5.1 SSH Session Timeout

By default, SSH sessions are automatically terminated after 15 minutes of inactivity. This timeout can be changed as follows:

- Disable SSH session timeout:

```
updu-100499> configure
updu-100499(config)# ssh
updu-100499(config-ssh)# session-timeout off
updu-100499(config-ssh)#
```

- Terminate inactive SSH sessions after 1 hour:

```
updu-100499> configure
updu-100499(config)# ssh
updu-100499(config-ssh)# session-timeout 60
updu-100499(config-ssh)#
```

Changing the timeout doesn't apply to established sessions, only to new sessions.

9.5.2 Clear SSH Host Keys

The `clear ssh-hostkey` normal mode command clears the host key used by the SSH server. A new key pair is generated the next time the SSH server is restarted (i.e. using the `disabled` / `enabled` SSH configuration commands or using a reboot of the UPDU).

Example:

```
updu-100499> clear ssh-hostkey
SSH host key cleared. A new key pair is generated the next time
the SSH server is started (e.g. after rebooting the UPDU).
```

9.5.3 Show SSH Server Host Key Fingerprint

The `show ssh-hostkey` shows the fingerprint of the SSH host key in OpenSSH format.

```
updu-100499> show ssh-hostkey
Fingerprint: SHA256:WG3ha0d4z/myt1IqNfCb5pGBYB/MTioYHvFU+W4mLVg
```

9.6 SMTP Configuration

The UPDU can be configured to send notifications by email using the Simple Mail Transfer Protocol (SMTP). This is done in the `smtp` configuration context.

9.6.1 Enable or Disable SMTP

The SMTP service can be enabled or disabled without losing the configuration:

- Enable SMTP:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# enabled
```

- Disable SMTP:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# disabled
```

9.6.2 Configure Mail Server

The `server` command configures the address and port of the SMTP server and the connection method. The following connection methods are available:

- `plain`: Unencrypted TCP connections, typically to ports 25 or 587.
- `tls`: Encrypted TLS connections, typically to port 465.
- `starttls`: Also called opportunistic TLS connections, are plain TCP connections which are upgraded to TLS after the connection is established. Typically used on ports 25 or 587.

Examples:

- Connect using STARTTLS to the submission port (587) of the mail server at `smtp.domain.com`:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# server starttls smtp.domain.com port 587
```

- Don't use a mail server:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# server none
```

9.6.3 Configure Authentication

Depending on the mail server, the UPDU has to authenticate itself before being able to send emails. The UPDU supports SMTP authentication using username/password credentials.

Examples:

- Disable SMTP authentication:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# smtp-auth disabled
```

- Authenticate using username `foo` and a password:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# smtp-auth login foo "secret-pass"
```

9.6.4 Configure the Sender Email Address

The `sender` command configures the `From` address of outbound emails.

Example:

- Send emails with `updu@domain.com` in the `From` address:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# sender updu@domain.com
```

9.6.5 Test SMTP

In order to confirm that the configured SMTP configuration works, the `test smtp` CLI command can be used. It sends a test email message to the specified recipient:

```
updu-100499> test smtp testuser@domain.com
```

9.6.6 Configure SMTP Notifications

When SMTP is configured, the UPDU can send notifications from the monitoring system as well as audit events by email. This is done with the `notify-recipient` command. Up to three email recipients can be configured.

Examples:

- Send notification emails to `alarm@domain.com`:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# notify-recipient alarm@domain.com
```

- Delete notification emails to `alarm@domain.com`:

```
updu-100499> configure
updu-100499(config)# smtp
updu-100499(config-smtp)# delete notify-recipient alarm@domain.com
```

9.7 Telnet Configuration

The `telnet` configuration sub-mode allows to control the telnet service.

- Enable telnet.

```
updu-100499> configure
updu-100499(config)# telnet
updu-100499(config-telnet)# enabled
updu-100499(config-telnet)#
```

- Disable telnet:

```
updu-100499> configure
updu-100499(config)# telnet
updu-100499(config-telnet)# disabled
updu-100499(config-telnet)#
```

By default, telnet is disabled.

9.7.1 Telnet Session Timeout

By default, telnet sessions are automatically terminated after 15 minutes of inactivity. This timeout can be changed as follows:

- Disable telnet session timeout:

```
updu-100499> configure
updu-100499(config)# telnet
updu-100499(config-telnet)# session-timeout off
updu-100499(config-telnet)#
```

- Terminate inactive telnet sessions after 1 hour:

```
updu-100499> configure
updu-100499(config)# telnet
updu-100499(config-telnet)# session-timeout 60
updu-100499(config-telnet)#
```

Changing the timeout doesn't apply to established sessions, only to new sessions.

10 Licenses

Some UPDU features require purchasing additional licenses.

10.1 Show Licenses

The `show license` command shows all licenses installed in the PDU.

To get the raw base64-encoded license data, use the `show license raw` command. The output is suitable for the `add license` command.

10.2 Activate Licenses

License can be activated using the `add license` command:

```
updu-100499> add license
Enter license-data, exit with two empty lines or CTRL-c or CTRL-d.
> gwFYQFTP1N90RizCZ8JuHB+AbFSnIhzBCfNrylyVsB1DtFSidMyS0YZfyzRJA85rD+uVK6qLGwP9
> kvCXexpHmaRqUQdUhAEaB1vNFRoAD0cSgYIBZFRlc3Q=
Valid license parsed: S/N 123456789
>
>
Note that newly added licenses will only show up after a reboot.
updu-100499>
```

The `add license` command accepts base64-encoded license data. It ignores comments included with the license data and licenses for other PDUs. When a valid license is found, it is stored in the PDU and a message is printed.

The `add license` command can be quit using two consecutive empty lines or CTRL-c or CTRL-d.

Once activated, licenses cannot be deactivated.

In order to activate the licensed feature, a reboot is required after adding the license.

11 Troubleshooting

11.1 Tracing

Some aspects of the UPDU can be traced for debugging purposes. When tracing is enabled, additional debugging log messages are logged. By default, tracing is switched off for all modules.

11.1.1 Enable Tracing

To enable tracing a functionality, use the `trace enable` command.

Example: Enable tracing RADIUS authentication:

```
updu-100499> trace enable radius
```

To obtain a list of all tracable modules, type `trace enable` and hit the `<TAB>` button.

11.1.2 Disable tracing

To disable tracing a functionality, use the `trace disable` command. To disable all tracing, use `trace disable all`.

Example: Disable tracing RADIUS authentication:

```
updu-100499> trace disable radius
```

11.2 Factory Reset

The `factory-reset` command resets settings to their factory defaults and reboots the UPDU. If requested, network settings (interface configuration and spanning-tree protocol settings) can be preserved in order not to lose connectivity when the factory reset is initiated remotely.

Examples:

- Do a factory reset:

```
updu-100499> factory-reset full
Enter "YES" if you really want to reset all settings to their
factory defaults and reboot the UPDU:
```

- Do a factory reset but preserve the current network settings:

```
updu-100499> factory-reset preserve-network
Enter "YES" if you really want to reset all settings (except network
settings) to their factory defaults and reboot the UPDU:
```

By specifying the `force` parameter, the interactive confirmation can be skipped (e.g. for scripting).

The `factory-reset` command cannot be abbreviated.

A reboot of the UPDU follows immediately after issuing this command.