Workgroup: IETF
Internet-Draft:
draft-rprice-ups-management-protocol-08
Published: 30 March 2022
Intended Status: Informational
Expires: 1 October 2022
Authors: R. Price, Ed.
Network UPS Tools Project

Uninterruptible Power Supply (UPS) Management Protocol -- Commands and Responses

Abstract

This document describes the command/response protocol currently used in the management of Uninterruptible Power Supply (UPS) units and other power devices often deployed in small offices, and in IT installations subject to an erratic public power supply. The UPS units typically interface to an Attachment Daemon in the system they protect. This Daemon is in turn polled by a Management Daemon which notifies users and system administrators of power supply incidents, and automates system shutdown decisions. The commands and responses described by this document are exchanged between the UPS Attachment Daemon and the Management Daemon. Current practice when this text was written risks weak security and this is addressed in the Security Considerations sections of this document.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 1 October 2022.

Table of Contents

1. Introduction

1.1. Current Practice

This document describes UPS management techniques and current UPS management practice published by the NUT Project. The document is based on version 2.8.0 of the NUT Project software which supports version 1.3 of the NUT protocol.

Since May 2002, the protocol described by this document has been operating on IANA port nut/3493 running over TCP.

1.1.1. NUT Software Project

The primary goal of the NUT (Network UPS Tools) Software Project [NUT] is to provide support for Power Devices, such as

Uninterruptible Power Supplies. The Project has been in operation since 1998 with a major rework in 2003. It operates through a user mailing list [nut-upsuser], a developer mailing list [nut-upsdev], a web site [NUT] and a GitHub repository [nut-repository]. See [githist] and [History] for a history of the project.

1.1.2. The "Shutdown Story"

Appendix "The Shutdown Story" describes the current UPS management practice for performing a managed shutdown of unattended infrastructure after an unscheduled failure of the public power supply in order to minimise the risk of corruption to data processed by this infrastructure. See Appendix B.

1.1.3. How to Read this Document

To lighten the text, the term "UPS" is used when "Managed Power Device" would be more complete. The reader should understand the simple "UPS" to include other managed power devices.

The statuses and events appearing in this document are named with short text-form names, some of which are abbreviations. A full list of the statuses can be found in Section 5.1 while the events are listed in Section 5.2.

This document refers to the "public power supply". Other texts frequently refer to "utility power", "input source power" or even "wall power".

1.2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.3. Additional Information

Additional information about the NUT Project is available in the project documentation [Documentation]. Requests for further information about this protocol and related technical matters may be addressed to the mailing list [nut-upsuser] of the NUT Project.

2. Terminology

The following technical terms appear in this document. They are listed in alphabetical order.

2.1. Attachment Daemon

The Attachment Daemon retrieves status from the UPS and sends commands to it often through a Driver specific to the hardware model and the connection medium, e.g., USB, serial. See Section 2.3. It maintains an abstracted view of the hardware through the use of hardware statuses. See Section 2.10. A Management Daemon may consult the abstracted view using the commands described in this document.

See Section 9.2 for details of the recommended minimum support of variables which calls for Attachment Daemon support of statuses OB, OL, LB and FSD.

The NUT Project implemented an Attachment Daemon as program upsd and a set of hardware specific drivers, all written in K&R C. The Attachment Daemon is launched as system user "root", but for better security, then drops privilege to run as a detached software service.

2.2. Administrative User

In current practice, the commands and other functions offered by the Attachment Daemon to each Management Daemon are made available to a set of Management Daemon users which authenticate to the Attachment Daemon with basic credentials (username and password). These users are not system users, they are specific to an Attachment Daemon and are listed in a text file (currently upsd.users) which is read by the Attachment Daemon and which assigns to each of them the password, Instant Commands and actions which are allowed, together with the Primary or Secondary status of the Management Daemon. For details, see Appendix E.1. For details of Primary see Section 2.7, and for details of Secondary see Section 2.8. Typically a high-level user will be able to send command FSD but a low-level user might only be allowed to access the test panel. The security provisions for administrative users are discussed in Appendix E.

2.3. Driver

A Driver is that part of an Attachment Daemon which is specific to the UPS hardware, the connection medium and the connection protocol, e.g., USB, serial. In current practice the Attachment Daemon has a driver for each hardware interface type it supports. Although this document considers the driver to be part of the Attachment Daemon, current practice is to see it as a separate software unit running as a Daemon "in front of" the Attachment Daemon. The protocol for data exchange between the Driver and the Attachment Daemon is outside the scope of this document.

2.4. Event

A UPS Event occurs in the Management Daemon when a change in UPS status is received from the Attachment Daemon. This event is internal to the Management Daemon. See Section 5.2.

2.5. Instant Command

A command which when sent to the Attachment Daemon is passed to the driver and sent to the hardware without any configured delay to perform a function. For example INSTCMD su700 test.panel.start . See Section 4.2.6.

2.6. Management Daemon

The Management Daemon is primarily responsible for managing the hardware and orchestrating system-wide actions after a power event. Using commands sent to the Attachment Daemon it follows the status of the UPS and determines when UPS events occur. It takes decisions based on the events, such as calling for a system shutdown. See Appendix B. Although the term includes the word "Daemon" nothing requires that it be implemented as a detached software service. The Management Daemon may also provide administrative functions such as a graphic interface to view the hardware activity.

There are several examples of a Management Daemon: the NUT Project provides upsmon which takes the system shutdown decision when the public power supply fails. Further configuration options such as timers were provided by helper program upssched.

Other programs represent the Management Daemon:

* upsc reports the values of the variables defined for a given UPS, see Table 7.
* upsrw reports on and changes the values of the readable and writable configuration variables defined for a given UPS, see Appendix A.2.
* upscmd reports on and executes the instant action commands defined for a given UPS, see Section 4.2.6.
* UPSmon.py is an experimental Python3 rewrite of upsmon and upssched which includes support for TLS 1.3 [RFC8446].

2.7. Primary

When a power device such as a UPS unit supplies power to more than one system, the computer running the driver is known as the Primary. The others are Secondaries. See figure 4. Common current practice for system administrators is to consider the Management Daemon in the Primary to be the Primary Management Daemon which is in charge of the shutdown of all the systems powered by the UPS. The Primary

Management Daemon sets status symbol FSD to order the secondaries to shut down.

Note: Historically, the Primary was known as the "Master".

2.8. Secondary

When a hardware device such as a UPS unit supplies power to more than one system, the system which communicates directly with the UPS unit e.g. using a USB, RS232, or network connection, is known as the Primariy. The other are Secondaries. There is no Attachment Daemon in a Secondary. See figure 4. Common current practice for system administrators is to consider the Management Daemon in a Secondary to be a Secondary Management Daemon which understands status symbol FSD as an order to shut down.

Note: Historically, the Secondary was known as the "Slave".

2.9. Session

The Management Daemon may initiate a TCP session with a specified device such as a UPS known to the Attachment Daemon. The session structure provides for audit and security as well as access to mission critical UPS functions. For example good practice requires a password protection for an Instant Command which turns off a UPS outlet. Other than the commands and responses used, the details of session management are outside the scope of this document.

2.10. UPS Status

The status of a hardware device such as a UPS unit is a symbolic description of the state of the unit. It consists of a space separated list of symbols from the set {ALARM BOOST BYPASS CAL CHRG COMM DISCHRG FSD LB NOCOMM OB OFF OL OVER RB TEST TRIM}. The symbols TICK and TOCK are experimental additions to the statuses and are not in common current practice. See Section 5.1 which specifies each of these symbols.

See Section 9.2 for details of the recommended minimum support of status symbols OB, OL, LB and FSD.

2.11. UPS Variable

The metrics and identifiers provided by each UPS are represented by variables giving the value representing that metric or identifier, The UPS variable is an abstraction of the UPS hardware configuration and activity maintained by the Attachment Daemon. See Appendix A which provides examples of variables. For example the variable battery.charge contains the current charge of the UPS battery as a percentage value.

Note: Some variables are constants, e.g. battery type, manufacturer.

See Section 9.2 for details of the recommended minimum support of variables. A full list of possible variables is available in source code file docs/nut-names.txt [gitvars] which serves as the Recording Document.

3. Protocol Overview

Figure 1 shows a reference configuration in which the command/response protocol applies. The UPS shown is representative of all managed power devices,

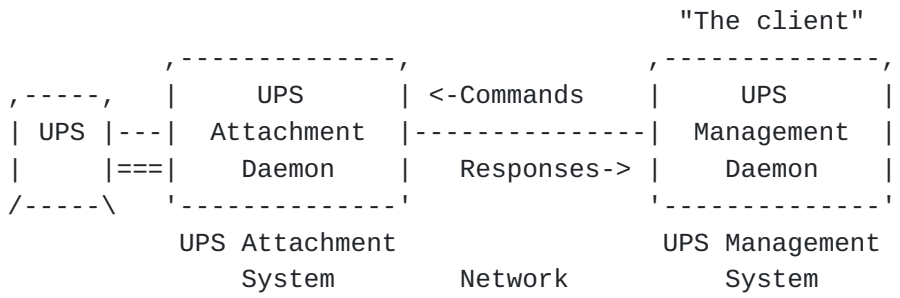

Figure 1: Reference Configuration

The reference configuration in figure 1 shows a single UPS unit which has a power supply link (===) and a data link (---) attached to a system running an Attachment Daemon. The UPS provides power supply protection to the system running the Attachment Daemon.

In practice there may be more than one UPS unit, and a unit may provide power protection to more than one system. The figure also shows a single Management Daemon. In practice there may be more than one Management Daemon, and any one Management Daemon may manage more than one UPS Attachment Daemon.

The protocol applies to connections between the Attachment Daemon and the Management Daemon which act as **server** and **client** respectively. The Management Daemon sends commands over TCP to the Attachment Daemon and receives responses over TCP from that Daemon.

The two Daemons may run in the same system, or may be connected through a local or wide area network. In simple cases, as shown in figure 2, the Attachment Daemon and the Management Daemon are in the same system, the one protected by the UPS. The commands and responses are exchanged through an internal loopback interface.

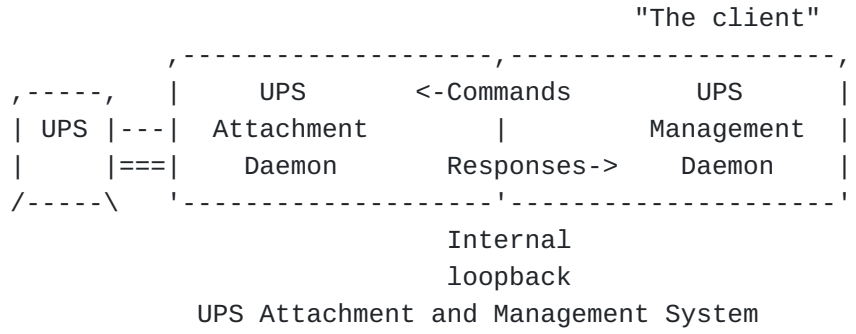

Figure 2: Simplified single-system configuration

The reference configuration does not require any specific design. For example figure 3 shows an arrangement in which the Attachment Daemon is closely associated with, or even included in the UPS system setup. This is becoming more prevalent with the availability of low cost processors able to run the Attachment Daemon thereby effectively creating a network attached UPS running a published protocol.

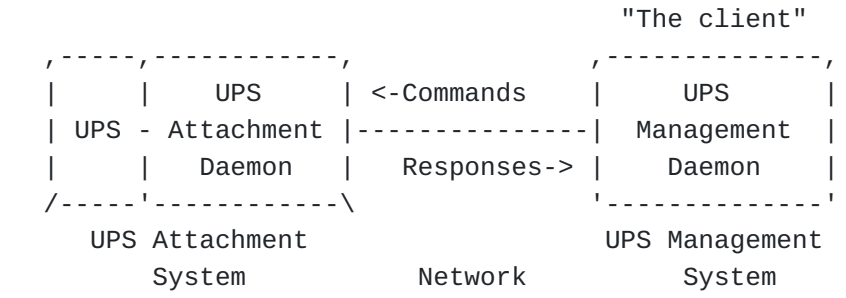

Figure 3: UPS and Attachment Daemon integration

As the power requirements for processors decrease, it is becoming increasingly common to use a single UPS to protect multiple systems as shown in figure 4. However there is only one data line (---) from the UPS to the Primary system. The others have only power connections (===) to the UPS, and are known as Secondaries. A Secondary does not run an Attachment Daemon, it connects over a network to the Attachment Daemon in the Primary. Figure 4 shows the Attachment Daemon and the Primary Management Daemon in the same system. This is common practice but it is not a technical requirement.

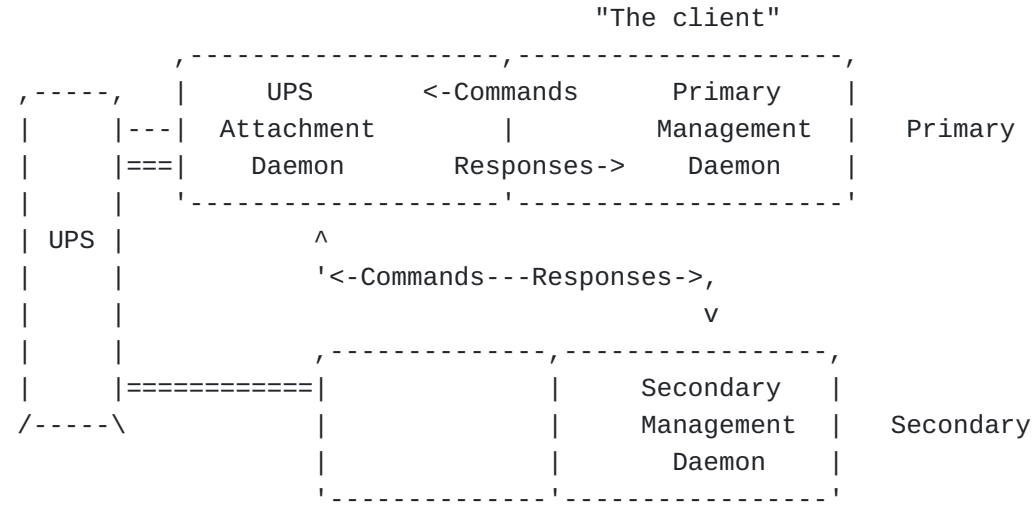

Figure 4: UPS protects multiple systems

Note: Should the Primary fail or go off-line, the fate of the Secondaries depends on the UPS status when the Primary failed. If the UPS had status OL the Secondary continues operation, but if the UPS had status OB the Secondary may choose to shut down as a precaution.

4. Protocol Specification

This specification includes only the commands and their responses. An implementation of the Attachment Daemon has an internal state machine, and some complex implementations of the Management Daemon include an internal state machine; for example to assist the system shutdown of a complex installation. The Management Daemon is required to remember the previous ups.status value it received from the Attachment Daemon and compare it with the next. Other than that the management protocol used between them is effectively stateless.

See for example Section 5.2 which shows a map of the new ups.status response and the previous ups.status response to an Event which is taken as the basis for Management Daemon action.

4.1. Notation Used in this Specification

The character set used for commands and responses is UTF-8 but current practice is to limit the character set used to the single byte UTF-8 characters 0-127.

Historical note: UTF-8 characters 0-127 were originally known as the American Standard Code for Information Interchange, US-ASCII, first published in 1963.

Multi-word elements are contained within U+0022 QUOTATION MARK characters for easier parsing. E.g., "UPS on fire". Embedded quotation marks are escaped with U+005C REVERSE SOLIDUS \ often

known as backslashes. Embedded backslashes are also escaped by representing them as \\.

Commands and responses have no leading or trailing whitespace, and are terminated with a single new line character U+000A LINE FEED (LF).

White space within commands and responses is reduced to one U+0020 SPACE (SP).

4.2. Commands

The commands address the UPS to which they apply by <upsname> where

- *<upsname> ::= <ups>[@<hostname>[:<port>]]
- *<ups> is defined by the Attachment Daemon configuration files.
- *The default <hostname> is localhost
- *The <port> is the number of the TCP port on which the Attachment Daemon is listening. The default is 3493. This is supported by all current Management Daemons.

Examples: myups, UPS-97B@bigserver.example.com

Note: Experimental Management Daemons use an extended form of <upsname> in configuration files and in program parameters, where

- *<upsname> ::= [<group>:]<ups>[@<hostname>[:<port>]]
- *<group> is an experimental extension to provide for groups of UPSs. It is not in common current practice.
- *<ups> is defined by the Attachment Daemon configuration files.
- *The default <hostname> is localhost

Examples: ups-1@example.com:3493, HB:heartbeat1@example.com:3493

In the current implementation, the names of commands and subcommands are not case sensitive. For example GET VAR may be written as Get var, but in this specification they are always written in upper case. Similarly, <upsname> and <varname> are not case sensitive. For example UPS341 ups.id may be written as ups341 Ups.Id, but in this specification <varname> is always written in lower case.

4.2.1. ATTACH

In a configuration such as that shown in Figure 4 in which a UPS protects more than one system, the Primary Management Daemon needs to know how many Secondaries are currently "*active*", i.e., powered by the UPS, either from the public power supply or from battery power. The Attachment Daemon supports this by keeping a count of all the "*active*" systems powered by a UPS. The count is initialised, one Secondary at a time by the ATTACH command, which should be

understood as "*count this Secondary as active*". ATTACH is one of three commands for Secondary counting: command DETACH decrements the count and a Management Daemon may read the count at any time using command NUMATTACH.

The ATTACH command is also sent to the Attachment Daemon for the Primary so during normal, fully protected operation, the count is 1 for the Primary + the number of secondaries. During a full system shutdown, the count drops as each Secondary Management Daemon executes command DETACH during its own shutdown. When the count drops to 1, only the Primary is "*active*" and it knows that all the secondaries have shut down.

Command: ATTACH <upsname>

If the command succeeds, the response is OK, otherwise see the error responses in Section 4.3.2.

Note: Historically, this command was known as LOGIN. Since that LOGIN was not the conventional user access to a shell or program the name was changed to avoid confusion.

4.2.2. DETACH

This companion command to ATTACH reduces the count of "active" Secondaries. It should be understood as "*this Secondary is no longer active*", and is usually used during system shutdown to decrement a count of how many Secondaries are still "active".

Command: DETACH

If the command succeeds, the response is OK Goodbye, otherwise see the error responses in Section 4.3.2.

Note: Historically, this command was known as LOGOUT.

4.2.3. FSD

A Management Daemon which is Primary and has the required authority, uses this command to set status symbol FSD meaning "Forced Shutdown" in the Attachment Daemon. In current practice the Primary Management Daemon uses the symbol to tell the Secondaries to shut down.

Command: FSD <upsname>

If the command succeeds, the response is OK FSD-SET, otherwise see the error responses in Section 4.3.2.

In current practice, commands such as FSD are made available only to a privileged administrative user authorized to send such a mission

critical command. The security provisions for administrative users are discussed in Appendix E.

Note: The symbol "FSD" is also used for an Event. See Table 5.

4.2.4. GET

Retrieve a single response from the Attachment Daemon. The possible sub-commands are:

4.2.4.1. GET CMDDESC

Retrieve a text description of a command.

Command: GET CMDDESC <upsname> <cmdname>

Response: CMDDESC <upsname> <cmdname> "<description>"

For example: GET CMDDESC su700 load.on and response CMDDESC su700 load.on "Turn on the load immediately"

This is like GET DESC, but it applies to an Instant Command;. See Section 4.2.4.2.

4.2.4.2. GET DESC

Retrieve a text description of a UPS variable. See Section 2.11.

Command: GET DESC <upsname> <varname>

Response: DESC <upsname> <varname> "<description>"

where <description> is a string that gives a brief explanation of the named variable. The Attachment Daemon **MAY** return "Unavailable" if the file which provides this description is not installed.

For example command GET DESC su700 ups.status and response DESC su700 ups.status "UPS status"

4.2.4.3. GET NUMATTACH

Retrieve the count kept by the Attachment Daemon of all the "*active*" systems protected by this UPS.

Command: GET NUMATTACH <upsname>

Response: NUMATTACH <upsname> <value>

where <value> is a count of the Primary and the number of Secondaries currently powered by this UPS.

For example command GET ATTACH su700 and response NUMATTACH su700 1

This information is needed by the Management Daemon to determine how many Secondaries are still connected during the system shutdown process.

Note: Historically, this sub-command was known as NUMLOGINS. Since LOGIN was not the conventional user access to a shell or program the name was changed to avoid confusion.

4.2.4.4. GET TYPE

Retrieve the type of a UPS variable. See Section 2.11.

Command: GET TYPE <upsname> <varname>

Response: TYPE <upsname> <varname> <type>...

where <type> can be one or more of the following tokens. Multiple types may be returned.

For example command GET TYPE su700 input.transfer.low and response TYPE su700 input.transfer.low ENUM

Type	Meaning
RW	This is a read/write variable. It may be read with command GET VAR, see Section 4.2.4.6, and set to a different value with command SET, see Section 4.2.11.
ENUM	An enumerated type, which supports specific predetermined values.
STRING:n	This is a string of maximum length n.
RANGE	This is a number, either integer or float, comprised in the range which may be seen with the command LIST RANGE, see Section 4.2.7.4.
NUMBER	This is a single numeric value, either integer or float.

Table 1: Variable Types

Notes:

*ENUM, STRING:n and RANGE are usually associated with RW, but not always. The default <type>, when omitted, is numeric, so either integer or float. Each Driver is then responsible for handling values as either integer or float.

*Current practice is to represent floating point values using locale C.utf8 which is a decimal (base 10) US English-based representation. Hexadecimal, exponents, and comma for thousands

separator are not allowed. For example: "1200.20" is valid, while "1,200.20" and "1200,20" are not valid.

4.2.4.5. GET UPSDESC

Retrieve a text description of a UPS.

Command: GET UPSDESC <upsname>

Response: UPSDESC <upsname> "<description>"

where <description> is defined by the Attachment Daemon configuration. If it is not set, current practice is for the Attachment Daemon to return "Unavailable".

For example command GET UPSDESC su700 and response UPSDESC su700 "Development box"

This can be used to provide human-readable descriptions instead of a cryptic ups@hostname string.

4.2.4.6. GET VAR

Retrieve the value of a UPS variable. See Section 2.11.

Command: GET VAR <upsname> <varname>

Response: VAR <upsname> <varname> "<value>"

For example command GET VAR su700 ups.status and response VAR su700 ups.status "OB LB"

4.2.5. HELP

Return a list of the commands supported by the Attachment Daemon. This command is intended for human as well as program use.

Command HELP

For example, the following command line sequence executed on an Attachment Daemon:

```
netcat localhost 3493
HELP
Commands: HELP VER GET LIST SET INSTCMD ATTACH DETACH
          USERNAME PASSWORD STARTTLS
```

Note: Historically, this command also returned LOGIN and LOGOUT. Since LOGIN was not the conventional user access to a shell or program, the command names were changed to ATTACH and DETACH to avoid confusion.

4.2.6. INSTCMD

Send an Instant Command to the UPS.

Command: INSTCMD <upsname> <cmdname>

where <upsname> is the name of the UPS and <cmdname> is the Instant Command to be issued to that UPS. See Appendix A.3 for examples of instant commands.

If the command succeeds, the response is OK, otherwise see the error responses, Section 4.3.2.

For example the command: INSTCMD su700 test.panel.start and the response OK

4.2.7. LIST

All the LIST commands produce a response with a common format. The response will begin with BEGIN LIST and then repeat the initial query. A list then follows, with as many lines as are necessary. The response ends with END LIST followed by the initial query.

The formatting may seem a bit redundant, but it makes a different form of client possible. A client can send a LIST command and then wait for the response. When it arrives, the Management Daemon doesn't need a complicated state machine to remember which list is which.

Note: The current NUT Project implementation of the Attachment Daemon, upsd, sends back the response to the LIST command as a sequence of messages. The Management Daemon should continue reading these messages until it receives the line beginning END LIST.

The possible subcommands are:

4.2.7.1. LIST CLIENT

The command calls for the Attachment Daemon to report all the current Management Daemon clients of a given UPS.

Command: LIST CLIENT <upsname>

The response is

```
BEGIN LIST CLIENT <upsname>
CLIENT <upsname> <client_IP_address>
...
END LIST CLIENT <upsname>
```

For example, the command LIST CLIENT ups1 and the response:

```
BEGIN LIST CLIENT ups1
CLIENT ups1 ::1
CLIENT ups1 203.0.113.1
END LIST CLIENT ups1
```

4.2.7.2. LIST CMD

The command calls for the Attachment Daemon to report a list of the Instant Commands which the Management Daemon may send to the Attachment Daemon. This Instant Command list is the abstracted view of the UPS hardware capabilities. An economical UPS will support few or no Instant Commands but a professional model should support more.

Command: LIST CMD <upsname>

The response is:

```
BEGIN LIST CMD <upsname>
CMD <upsname> <cmdname>
...
END LIST CMD <upsname>
```

where <upsname> is the name of the UPS, and <cmdname> is the name of the Instant Command which may be issued to the UPS.

For example the command: LIST CMD su700 and the response:

```
BEGIN LIST CMD su700
CMD su700 load.on
CMD su700 test.panel.start
...
END LIST CMD su700
```

4.2.7.3. LIST ENUM

The command calls for the Attachment Daemon to report the set of possible values of a UPS variable which has predetermined values.

Command: LIST ENUM <upsname> <varname>

The response is:

```
BEGIN LIST ENUM <upsname> <varname>
ENUM <upsname> <varname> "<value>"
...
END LIST ENUM <upsname> <varname>
```

where <upsname> is the name of the UPS, <varname> is the UPS variable and <value> is one of the possible values of that UPS variable. Note that in current practice the output is an unordered list. Note also that the U+0022 QUOTATION MARK characters are part of the response.

For example the command: LIST ENUM su700 input.transfer.low and the response:

```
BEGIN LIST ENUM su700 input.transfer.low
ENUM su700 input.transfer.low "103"
ENUM su700 input.transfer.low "100"
...
END LIST ENUM su700 input.transfer.low
```

4.2.7.4. LIST RANGE

The command calls for the Attachment Daemon to report the interval in which valid values of UPS variable lie.

Command: LIST RANGE <upsname> <varname>

The response is:

```
BEGIN LIST RANGE <upsname> <varname>
RANGE <upsname> <varname> "<min>" "<max>"
...
END LIST RANGE <upsname> <varname>
```

where <upsname> is the name of the UPS, <varname> is the UPS variable and {<min>,<max>} is the interval of valid values of that UPS variable. Note that the U+0022 QUOTATION MARK characters are part of the response.

For example, the command LIST RANGE su700 input.transfer.low and the response:

```
BEGIN LIST RANGE su700 input.transfer.low
RANGE su700 input.transfer.low "90" "105"
END LIST RANGE su700 input.transfer.low
```

4.2.7.5. LIST RW

The command calls for the Attachment Daemon to report a list of the UPS variables associated with a given UPS which may be read and written by the Management Daemon. These variables are the abstracted view of the UPS hardware capabilities. An economical UPS may support few variables but a professional model should support at least the variables which are needed for an automatic shutdown and restart, see Appendix B. See also Section 9.2 for details of the recommended minimum support of variables. A full list of variables is available in source code file docs/nut-names.txt [gitvars] which serves as the Recording Document.

Command: LIST RW <upsname>

The response is:

```
BEGIN LIST RW <upsname>
RW <upsname> <varname> "<value>"
...
END LIST RW <upsname>
```

where <upsname> is the name of the UPS, <varname> is the UPS variable and <value> is the value of that UPS variable. Note that the U+0022 QUOTATION MARK characters are part of the response.

For example the command: LIST RW su700 and the response:

```
BEGIN LIST RW su700
RW su700 output.voltage.nominal "115"
RW su700 ups.delay.shutdown "020"
...
END LIST RW su700
```

4.2.7.6. LIST UPS

The command calls for the Attachment Daemon to report a list of the UPS units to which it is attached.

Command: LIST UPS

The response is:

```
BEGIN LIST UPS
UPS <upsname> "<description>"
...
END LIST UPS
```

where <upsname> is the name of a UPS, and <description> is the description maintained by the Attachment Daemon if available. It is set to "Unavailable" otherwise. Note that the U+0022 QUOTATION MARK characters are part of the response.

This command can also be used to determine what values of <upsname> are valid before calling other functions on the server. This is also a good way to handle situations where a single Attachment Daemon supports multiple UPS's. It is also useful for clients which perform a UPS discovery process.

For example, the response:

```
BEGIN LIST UPS
UPS su700 "Development box"
END LIST UPS
```

4.2.7.7. LIST VAR

The command calls for the Attachment Daemon to report a list of all the UPS variables which it maintains for a given UPS, and the values of those UPS variables.

Command: LIST VAR <upsname>

The response is:

```
BEGIN LIST VAR <upsname>
VAR <upsname> <varname> "<value>"
...
END LIST VAR <upsname>
```

where <upsname> is the name of the UPS, <varname> is the UPS variable and <value> is the value of that variable. Note that the U+0022 QUOTATION MARK characters are part of the response.

The response to this command lists the UPS variables available for this UPS and their current values. For example the command LIST VAR su700 and the response:

```
BEGIN LIST VAR su700
VAR su700 ups.mfr "Example Mfg"
VAR su700 ups.mfr.date "10/17/96"
...
END LIST VAR su700
```

See Section 9.2 for details of the recommended minimum support of variables. A full list of variables is available in source code file docs/nut-names.txt [gitvars] which serves as the Recording Document.

4.2.8. PASSWORD

This command is a companion to USERNAME, and is used by a Management Daemon to specify the password required to enter a Session with the Attachment Daemon, see Section 2.9.

Command: PASSWORD <password>

If the command succeeds, the response is OK, otherwise see the error responses, Section 4.3.2.

For examples of the use of commands USERNAME and PASSWORD by administrative users, see Appendix E.2.

4.2.9. PRIMARY

In current practice, the Attachment Daemon records in local file upsd.users that an administrative user is a Primary. See Appendix E.1 for an example. When a Management Daemon starts up and opens a Session with the Attachment Daemon, it lays claim to being a Primary by sending command PRIMARY to the Attachment Daemon, thus claiming that it has the required authority to perform such critical actions as setting status symbol FSD.

Command: PRIMARY <upsname>

where <upsname> is the name of the UPS.

If the Attachment Daemon has the authority, the response is OK, otherwise see the error responses in Section 4.3.2.

Note: Historically, this command was known as MASTER.

4.2.10. PROTVER

Return the implementation version of the command/response protocol used by the Attachment Daemon. This command is intended for human as well as program use.

Command PROTVER

For example, the following command line sequence in the Attachment Daemon:

```
netcat localhost 3493
PROTVER
1.3
```

Notes:

1. There are no U+0022 QUOTATION MARK characters in the response.
2. Historically, this command was known as NETVER and current practice is to use NETVER instead of PROTVER.
3. The implementation version of the protocol returned by PROTVER is different to the implementation version of the Attachment Daemon returned by VER.
4. The protocol version 1.3 requires support of the commands used in previous protocol version 1.2 aliases.

4.2.11. SET

The command calls for the Attachment Daemon to set a UPS variable to a given value. Whether this has an effect on the UPS hardware is specific to the Driver and the UPS model. Some variables are read-only due to the design of the UPS or its driver.

Command: SET VAR <upsname> <varname> "<value>"

where <upsname> is the name of the UPS, <varname> is the UPS variable and <value> is the value to be assigned to that variable. Note that the U+0022 QUOTATION MARK characters are part of the command.

If the command succeeds, the response is OK, otherwise see the error responses in Section 4.3.2.

For example the command: SET VAR su700 ups.id "My UPS" and the response OK

4.2.12. STARTTLS

The client tells the Attachment Daemon to switch to TLS [RFC8446] encrypted communication. When the client receives OK it also switches to TLS encryption.

Command: STARTTLS

If the command succeeds, the response is OK STARTTLS, otherwise see the error responses in Section 4.3.2.

If the client does not send command STARTTLS to the Attachment Daemon communication continues unencrypted, however an Attachment Daemon **MAY** refuse unencrypted communication.

Notes:

1. NUT 2.8.0 supports the encryption of communications between the Attachment Daemon and the Management Daemon using TLS 1.3 [RFC8446] with X.509 v3 certificates as defined by RFC5280 [RFC5280] and updates.

2. In the closely restrained world of UPS management, it may be possible to obtain better security usig self-signed certificates.

4.2.13. USERNAME

The Attachment Daemon limits access to clients whose credentials match those in the file upsd.users. There is no anonymous access. A Management Daemon program or script uses command USERNAME and its companion command PASSWORD to open a Session with the Attachment Daemon for an administrative user, Note that this command is for program or script use and is not the familiar login command typed on a command line to gain access to a shell.

Command: USERNAME <username>

If the command succeeds, the response is OK, otherwise see the error responses in Section 4.3.2.

For examples of the use of commands USERNAME and PASSWORD by administrative users, see Appendix E.2.

4.2.14. VER

Return the implementation version of the Attachment Daemon. This command is intended for human as well as program use.

Command VER

For example, the following command line sequence:

```
netcat localhost 3493
VER
Network UPS Tools upsd 2.8.0 - http://www.networkupstools.org/
```

Notes:

1. There are no U+0022 QUOTATION MARK characters in the response.
2. The implementation version of the Attachment Daemon returned by VER is different to the protocol version returned by PROTVER.
3. NUT version 2.8.0 supports the commands used in previous version 2.7.4 as aliases.

4.3. Summary of Responses

4.3.1. Response when Command Succeeds

If the command succeeds, the response has the following command-dependent form:

Command	Response	Note	Reference
ATTACH	OK	Was LOGIN	Section 4.2.1
DETACH	OK Goodbye	Was LOGOUT	Section 4.2.2
FSD	OK FSD-SET		Section 4.2.3
GET	Sub command specific		Section 4.2.4
HELP	List of commands		Section 4.2.5
INSTCMD	OK		Section 4.2.6
LIST	Sub command specific		Section 4.2.7
PASSWORD	OK		Section 4.2.8
PRIMARY	OK		Section 4.2.9
PROTVER	Protocol version	Was NETVER	Section 4.2.10
SET	OK		Section 4.2.11
STARTTLS	OK STARTTLS		Section 4.2.12
USERNAME	OK		Section 4.2.13
VER	Program version		Section 4.2.14

Table 2: Response if command succeeds

4.3.2. Error Responses

Error responses have the following format:

```
ERR <error-name> [<extra>]
```

where <error-name> is a single word token taken from the 27 characters A-Z and U+002D HYPHEN-MINUS. Implementations **MAY** if needed add an additional optional <extra>. Current practice does not make use of this possibility.

The <error-name> may have one of the following values:

The error name token <error-name>	Meaning
ACCESS-DENIED	The client's host and/or authentication details supplied by USERNAME and PASSWORD are not sufficient to execute the requested command.
ALREADY-ATTACHED	The client has already sent a successful ATTACH command for a given UPS and can't do it again. Note: Historically, this error response was ALREADY-LOGGED-IN.
ALREADY-SET-PASSWORD	The client has already supplied a PASSWORD and is attempting to repeat the command in the same Session.
ALREADY-SET-USERNAME	The client has already supplied a USERNAME, and is attempting to repeat the command within the same Session.
CMD-NOT-SUPPORTED	The specified UPS doesn't support the Instant Command.
DATA-STALE	The Attachment Daemon is connected to the Driver for the UPS, but that driver isn't providing regular updates or has specifically marked the data as stale. Current practice is for the Attachment Daemon to refuse to provide the Management Daemon with variables on stale units to avoid false readings. This generally means that the Driver is running, but it has lost communication with the hardware. Check the physical connection to the equipment.
DRIVER-NOT-CONNECTED	The Attachment Daemon can't perform the requested command, since the Driver for that UPS is not connected. This usually means that the driver is not running, or if it is, is misconfigured.
FEATURE-NOT-CONFIGURED	This instance of the Attachment Daemon hasn't been configured properly to allow the requested feature to operate. In current practice this error response is possible only for command STARTTLS.

The error name token <error-name>	Meaning
FEATURE-NOT-SUPPORTED	This instance of Attachment Daemon does not support the requested feature. In current practice this error response is possible only for command STARTTLS.
INSTCMD-FAILED	The Attachment Daemon failed to deliver the Instant Command request to the Driver. No further information is available to the client. This typically indicates a dead or broken driver.
INVALID-ARGUMENT	The client sent an argument to a command which is not recognized or is otherwise not valid in this context. This is typically caused by sending a valid command such as GET with a subcommand which is not valid.
INVALID-PASSWORD	The client sent a non valid PASSWORD.
INVALID-USERNAME	The client sent an non valid USERNAME.
INVALID-VALUE	The value specified in the request is not valid. This usually applies to a SET of an ENUM type which is using a value not in the list of allowed values. See Section 4.2.7.3.
PASSWORD-REQUIRED	The command requires a previous PASSWORD for authentication, but the client hasn't provided one.
READONLY	The requested variable in a SET command is not writable.
SET-FAILED	The Attachment Daemon failed to deliver the SET request to the Driver. This is similar to INSTCMD-FAILED.
TLS-ALREADY-ENABLED	TLS mode is already enabled on this connection, so the Attachment Daemon can't start it again. Note: Historically, this message was ALREADY-SSL-MODE.
TLS-NOT-ENABLED	TLS mode is required but has not yet been enabled on this connection, so the Attachment Daemon can't send commands. Note: This message is experimental and not in current common use.
TOO-LONG	The requested value in a SET command is too long.
UNKNOWN-COMMAND	The Attachment Daemon doesn't recognize the command.
UNKNOWN-UPS	The UPS specified in the request is not known to the Attachment Daemon. This

The error name token <error-name>	Meaning
	usually means that it didn't match anything in the Attachment Daemon configuration.
USERNAME-REQUIRED	The command requires a USERNAME for authentication, but the client hasn't provided one.
VAR-NOT-SUPPORTED	The specified UPS doesn't support the UPS variable in the command.

Table 3: Error responses

4.4. An ABNF of the Commands

This section repeats the syntax of Section 4.2, but in Augmented Bachus-Naur Form (ABNF). It does not define any additional feature. For further details of each command and the response, see Section 4.2.

The commands may be presented in ABNF [RFC5234], [RFC7405], and represented using Unicode [ISOIEC10646] and encoded in UTF-8 [RFC5198], a transformation format of ISO 10646.

In this ABNF, non-terminals are written in lower case letters, terminals are written in upper case letters. User supplied values are written in lower case letters and are enclosed between Less-Than Sign U+003C and Grater-Than Sign U+003E: <cmdname> is defined in Section 4.2.4.1, <value> is defined in Section 4.2.11, <varname> is defined in Section 4.2.4.1, <upsname> is defined in Section 4.2.1 and <username> is defined in Section 4.2.13. Current practice tolerates mixed case command names, but it is RECOMMENDED to use upper case only for commands. See Figure 5.

```
command = attach / detach / fsd / %s"GET" getsubcommand / help /
          instcmd / %s"LIST" listsubcommand / password / primary /
          protver / set / starttls / username / ver
attach = %s"ATTACH" <upsname>
detach = %s"DETACH"
fsd = %s"FSD" <upsname>
getsubcommand = getcmddesc / getdesc / getnumattach / gettype /
                getupsdesc / getvar
getcmddesc = %s"CMDDESC" <upsname> <cmdname>
getdesc = %s"DESC" <upsname> <varname>
getnumattach = %s"NUMATTACH" <upsname>
gettype = %s"TYPE" <upsname> <varname>
getupsdesc = %s"UPSDESC" <upsname>
getvar = %s"VAR" <upsname> <varname>
help = %s"HELP"
instcmd = %s"INSTCMD" <upsname> <cmdname>
listsubcommand = listclient / listcmd / listenum / listrange / listrw /
                 listups / listvar
listclient = %s"CLIENT" <upsname>
listcmd = %s"CMD" <upsname>
listenum = %s"ENUM" <upsname> <varname>
listrange = %s"RANGE" <upsname> <varname>
listrw = %s"RW" <upsname>
listups = %s"UPS"
listvar = %s"VAR" <upsname>
password = %s"PASSWORD" <password>
primary = %s"PRIMARY" <upsname>
protver = %s"PROTVER"
set = %s"SET" %s"VAR" <upsname> <varname> U+0022<value>U+0022
starttls = %s"STARTTLS"
username = %s"USERNAME" <username>
ver = %s"VER"
```

Figure 5: ABNF for the Commands

5. Statuses and Events

5.1. Status Symbols

These symbols resume the abstracted view of the UPS hardware maintained by the Attachment Daemon. The variable ups.status contains one or more space-separated status symbols which together describe the UPS state at that instant. In current practice the Management Daemon will poll variable ups.status every 5 seconds with a command such as GET VAR su700 ups.status and response VAR su700 ups.status "OB LB" to discover changes in the UPS status. These changes will indicate UPS events.

Status Symbol	Meaning
ALARM	The UPS reports that it requires intervention.
BOOST	The UPS has determined that the voltage level of the public power supply is too low, and is boosting it to the required level. The UPS continues to supply the protected system from the public power supply.
BYPASS	The UPS is feeding current directly from the public power supply to the protected system. The backup facilities are disconnected. This state allows maintenance personnel to change the batteries without interrupting the protected system.
CAL	The UPS is calibrating itself, for example to determine at what charge the LB status is raised or lowered.
CHRG	The UPS battery is charging. This usually implies that the UPS also has status OL, but may not be the case if the UPS also has status OFF.
COMM	The Attachment Daemon has effective contact with the UPS.
DISCHRG	The UPS battery is discharging. This usually implies that the UPS also has status OB, but may not be the case if the UPS also has status OFF.
FSD	This "Forced Shutdown" status signals that the final shutdown sequence has begun.
LB	Low Battery. The battery level of the UPS is below a chosen limit. The UPS may be in status OL or OB.
NOCOMM	The Attachment Daemon has no effective contact with the UPS.
OB	On Battery. The UPS is taking energy from it's battery. The battery is discharging. A UPS must have status OB or OL, otherwise it is deemed dead.
OFF	The UPS is in state "Off". It does not react to failure in the public power supply. The exact meaning depends on the model.
OL	Online. The UPS is online, receiving energy from the public power supply. The battery is charging. A UPS must have status OB or OL, otherwise it is deemed dead.
OVER	Overloaded. The UPS reports that the load on it is beyond it's normal operating maximum.
RB	Replace battery. The UPS reports that it's battery/ batteries should be replaced.
TEST	Under test. The UPS is currently undergoing a test, which may have been called for manually or internally.
TICK	Heartbeat. A software UPS in the Attachment Daemon provides a regular signal monitored by the Management Daemon as a way of verifying effective end-to-end management. TICK and TOCK are companions, they are considered experimental.
TOCK	Heartbeat. See TICK
TRIM	

Status Symbol	Meaning
	The UPS has determined that the voltage level of the public power supply is too high, and is reducing it to the required level. The UPS continues to supply the protected system from the public power supply.

Table 4: UPS Status Symbols

5.2. Events

A Management Daemon detects the occurrence of a UPS Event from a change in the UPS status received from the Attachment Daemon. The following table summarizes the process. A status of "none" means that the status symbol is not present in the variable ups.status.

The Management Daemon should retrieve the variable ups.status from the Attachment Daemon at regular intervals. If the interval is too short, compute and network resources will be wasted, but if the interval is too large, the Management Daemon risks missing short-lived changes in the UPS status.

A default value of 5 seconds is **RECOMMENDED**, but an implementation **MAY** make this value configurable. By default the "old" status is therefore the previous value retrieved 5 seconds ago.

Current practice is for the Management Daemon to assign names to certain events. These is shown in the table in parentheses.

Old status	New status	Event		Old status	New status	Event
none	ALARM	Alarm on		ALARM	none	Alarm off
none	BOOST	Boosting voltage		BOOST	none	Not boosting
none	BYPASS	Bypass on		BYPASS	none	Bypass off
none	CAL	Calibrating		CAL	none	Not calibrating
none	CHRG	Charging		CHRG	none	Not charging
none	COMM	UPS communicating (Event COMMOK)		COMM	none	See note 4
none	DISCHRG	Discharging		DISCHRG	none	Not discharging
none	FSD	System shutdown (Events FSD, SHUTDOWN)		FSD	none	Shutdown abandoned. See note 1
none	LB	Low battery. See note 2 (Event LOWBATT)		LB	none	Battery not low
none	NOCOMM	UPS dead? See note 4 (Events COMMBAD, NOCOMM)		NOCOMM	none	See note 4

Old status	New status	Event		Old status	New status	Event
none	OFF	UPS turned off		OFF	none	UPS not turned off
OB	OL	Receiving power (Event ONLINE)		OL	OB	Power lost (Event ONBATT)
none	OVER	UPS overloaded		OVER	none	Overload gone
none	RB	Replace battery (Event REPLBATT)		RB	none	Replacement canceled
none	TEST	Test starts		TEST	none	Test finished
none	TICK	Heartbeat event. See note 3		TICK	none	No heartbeat. See note 3
none	TOCK	Heartbeat event. See note 3		TOCK	none	No heartbeat. See note 3
none	TRIM	Trimming voltage		TRIM	none	Not trimming

Table 5: Event deduction from status changes

Notes

1. Current practice does not include this event.
2. If the status OB is present, current practice takes Management Daemon reception of LB as an order to perform an emergency system shutdown.
3. The use of a software defined UPS to provide a heartbeat is experimental and is not part of common current practice.
4. Current practice is: if the UPS has not responded for 15 seconds, the Management Daemon assumes that the UPS is "*dead*" (Event NOCOMM), and if the last known OL/OB status was OB a system shutdown, command FSD, is called for.

6. Security Considerations

6.1. Current General Security Practice

Experience over the last 20 years shows that new UPS management software releases are not frequent, and when installed, stay unmodified for some years. This is probably because UPS management is a mature hardware dependent activity. A limited number of system administrators have access to the UPS hardware and software and tend to assume a certain "security by obscurity" since many installations have a configuration as shown in figure 6 which uses port nut/3493 between the two Daemons running in the same system. The traffic is often not encrypted, and when encrypted uses deprecated early versions of SSL/TLS.

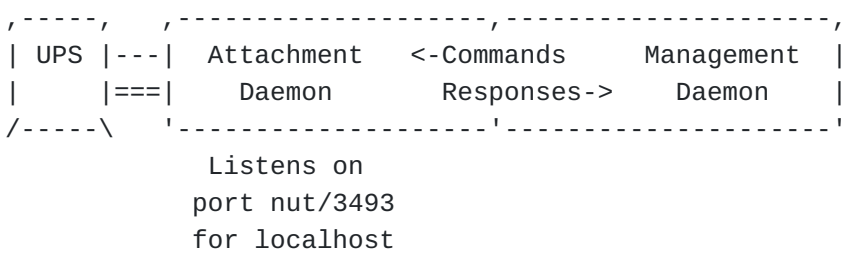

Figure 6: Common single-system configuration

This situation is now changing as low cost processors become available, costing significantly less than a UPS unit. This evolution makes it interesting to shift to a configuration as shown in figure 7, but it also exacerbates the security weakness of figure 6 since the traffic between the Daemons is now over an exposed network.

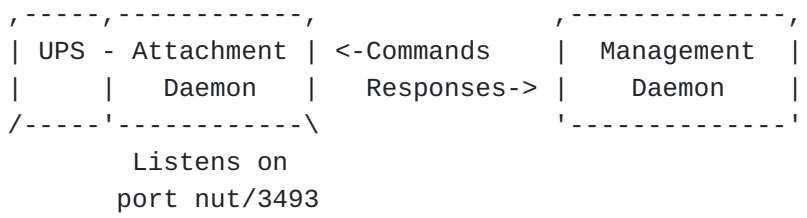

Figure 7: Integration of UPS and Attachment Daemon

These security issues raised by UPS management are those of the power industry in general: they are addressed in detail in Technical Specification IEC 62351-1 [IEC62351-1]. In addition to equipment security, cyber security is now an essential consideration.

Quoting from IEC 62351-1 clause 5.2.3.5 [IEC62351-1]:

> With the computer systems for power operations presumably kept isolated from the Internet, many utility personnel do not see any reason for adding security measures to these systems. However, as clearly seen from these Subclauses, this may not be true anymore as networking becomes more prevalent and additional information access requirements grow.

Clause 5.3.5 [IEC62351-1] lists typical security attacks: Eavesdropping, Masquerade, Man-in-the-Middle, Replay, Resource Exhaustion. RFC3552 [RFC3552] adds message insersion / deletion / modification, and denial of service.

Let's look more closely at these requirements:

*Eavesdropping, see Section 6.3.1

*Man-in-the-Middle, see Section 6.3.2

*Masquerade, see Section 6.3.3

*Message insersion, deletion, modification, see Section 6.3.4

*Replay, see Section 6.3.5

*Resource Exhaustion, Denial of Service, see Section 6.3.6

6.2. Communication Security Requirements

Enforcing secure communication requires tightening up the Attachment Daemon to require the use of command STARTTLS for commands sent over the global Internet. In such a situation an Attachment Daemon listening for traffic other than from the localhost:

1. **SHOULD** require and accept command STARTTLS,

2. **MUST** encrypt all communication with a Management Daemon,

3. **SHALL** refuse all non-encrypted commands except an initial STARTTLS.

Notes:

*The **SHOULD** rather than **MUST** in Section 6.2, Paragraph 2, Item 1 above allows system administrators to enforce secure communication using other techniques which do not involve the STARTTLS command.

*If an Attachment Daemon requires that all commands be encrypted as required by the **MUST** in Section 6.2, Paragraph 2, Item 2 above, then automatically each Management Daemon **MUST** encrypt as well, since it has to do so in order to gain access.

*The **SHALL** in Section 6.2, Paragraph 2, Item 3 above applies to traffic from the global Internet. An Attachment Daemon **MAY** accept unencrypted commands from localhost if the local installation's security practices allow it, for example in a dedicated appliance.

*Note that the separate management of strongly secure traffic from the global Internet and weakly secure traffic from localhost can be achieved by using two ports: nut/3493 for the current weakly secure traffic from localhost, and some other port, perhaps ups/TBD1, for enforced secure communication, much in the manner of http and https.

Firewalls **SHOULD** be used to restrict the communication between the Attachment Daemon and the accepted Management Daemons, prohibiting and discarding traffic from any systems that are not part of the envisioned power management setup. Note: See Section 6.2, Paragraph 4, Item 1 above on the use of **SHOULD**.

6.3. Attacks and Defences

6.3.1. Eavesdropping

The defence against eavesdropping is encryption of the commands and responses passed between client Management Daemon and server Attachment Daemon. The protocol provides command STARTTLS, see Section 4.2.12, which calls on the Attachment Daemon to support TLS encryption of the communication. If this command is accepted, the Management Daemon also encrypts.

In current NUT Project practice, the use of TLS is optional, however a Management Daemon may refuse to accept unencrypted communication. This is done by setting declarations FORCESSL to 1 and CERTVERIFY to 1 in the Management Daemon configuration file.

6.3.1.1. Misplaced declarations requiring TLS

A further weakness is that the FORCESSL and CERTVERIFY declarations which enforce use of encryption are in the client Management Daemon configuration file and not in the Attachment Daemon. Secure practice requires enforcement by the server Attachment Daemon rather than a possibly rogue client Management Daemon out on the Internet.

This weakness may be mitigated with strict firewall rules which would prevent the rogue client Management Daemon from accessing the Attachment Daemon.

6.3.1.2. Weak protection in previous version 2.7.4

Although version 2.8.0 of NUT supports TLS v.3 [RFC8446] with X.509 v3 certificates as defined by RFC5280 [RFC5280], previous version 2.7.4 only supported earlier SSL/TLS versions. To overcome this weakness, The following techniques have been used:

*Shims, see Appendix D.1

*TLS tunnel, see Appendix D.2

*Virtual Private Network, VPN, see Appendix D.3

*Virtual Local Area network, VLAN, see Appendix D.4

6.3.2. Man in the Middle

The protocol relies on TLS encryption to prevent man-in-the-middle attacks. See Appendix D for defense methods used for previous NUT version 2.7.4.

6.3.3. Masquerade Attack: Agent Verification

The protocol allows a Management Daemon to send command FSD to an Attachment Daemon to shut down a working system and it's power supply as described in The Shutdown Story, see Appendix B. A malicious client acting as a Management Daemon could turn off the UPS power outlets causing the system to fail.

The protocol provides commands USERNAME, see Section 4.2.13, and PASSWORD, see Section 4.2.8, which allow an administrative user in a Management Daemon to authenticate itself to the Attachment Daemon, as a defence against masquerade attacks. The administrative user name and password need protection against local malicious users. This is done by restricting access to the configuration files.

6.3.4. Message insertion, deletion, modification

The protocol relies on TLS encryption to prevent message insertion, deletion and modification attacks. See Appendix D for defense methods used for previous NUT version 2.7.4.

6.3.5. Replay

There are two cases:

1. The replay is from a system other than an approved Management Daemon: the protocol relies on a firewall to block the traffic.

2. The replay is from an approved Management Daemon: the protocol relies on the Management Daemon's own security to prevent unauthorised access.

6.3.6. Denial of Service

The protocol relies on a very tightly specified firewall to prevent denial of service attacks. Only designated client Management Daemons should be able to reach the server Attachment Daemon.

7. Codepoint Management

This document raises five matters of codepoint management:

1. The namespaces occupied by the protocol commands described in this document, see Section 4.2.

2. The namespaces occupied by the protocol responses described in this document, see Section 4.3.2.

3. The namespace occupied by UPS status names,

4. The namespace occupied by UPS variable names,

5. The port name and port number used to manage UPS units.

7.1. Namespaces used by Commands, Responses, Statuses and Variables

Current NUT Project experience after more than 20 years is that the UPS management area advances slowly, and that there are few requests to modify or extend the Commands, Responses, Statuses and Variables. When this does occur, the NUT Project has been able to settle the matter without difficulty in the project mailing list. It is therefore proposed to not burden IANA with this namespace management and to continue with the current process in which the project in its mailing list acts as a Working Group.

The Commands, Responses, Statuses and Variables are currently recorded as follows:

Namespace	Recording document	Reference
Commands and Responses	This document	Commands, see Section 4.2, Responses, see Section 4.3
(Idem, historical record)	Project Developer Guide Ch 9	Developer Guide [devguide]
Statuses	This document	Section 5.1
(Idem, historical record)	Source code clients/ status.h	GitHub repository [gitstats]
Variables	Source code file docs/nut-names.txt	GitHub repository [gitvars]

Table 6: Project records of namespace allocation

7.2. Port Name and Number used to Manage UPS Units

See the IANA Registry [Registry] for the latest situation.

7.2.1. Port nut/3493

In 2002 IANA assigned port nut/3493 to project lead Russell Kroll, and updated the assignment to the NUT Project itself in 2020.

7.2.2. Port ups/401

In 2008 IANA assigned ups/401 "Uninterruptible Power Supply" to Mr. Charles Bennett as both assignee and contact. We have been unable to find any protocol document or other published activity report for this port other than the One Windows Trojan. Mr. Bennett himself died in 2015, see obituary [Bennett]. Since his email address was registered by IANA as bennettc@ohio.edu it is possible that the University of Ohio is a successor in interest. The editor tried to contact the IT support department of the university by email and telephone but was rejected. Ed: My non-contact was Mr. Keith Brock, IT Support Senior Specialist, brock@ohio.edu +1 740 597 2136

7.2.3. NUT Project Requirement

The NUT Project needs to address the the current weak security, see Section 6.1, of UPS management deployments, for example

* by implementing the "shim" technique of Figure 9 described in Appendix D.1 for providing secure access to the Attachment Daemon,

* or by providing a choice of ports through which an Attachment Daemon may receive commands: one for "legacy" traffic, the other for fully secured traffic.

The project needs a second registered port. Since ports are a limited resource, it would be better to re-use an existing port rather than request a new one, and the project is interested in using existing port ups/TBD1. Let's look more closely at this:

* The port name "ups" satisfies the Principle of Least Surprise. It is not surprising for a port called "ups" to be used to manage UPSs. It is unlikely to be used for anything else.
* There are no other known users of this port and no other published protocols or usage reports.
* The currently assigned port number 401 is for a system port. The project has no imperative need for such a port; a user port, TBD1, would be sufficient. The Attachment Daemon is a system activity, but it can be launched by root and dropped to a non-privileged user perfectly well on a user port.

  Note: In Unix-like systems a port with a number below 1024 is privileged and requires elevated permissions to manage.
* System ports are more likely to attract malicious scans than user ports.
* The project does not need to be assigned this port. The need is to be able to use port "ups".

8. IANA Considerations

8.1. Port Name ups: Reference to this Document

The NUT Project has a requirement to use a second port, see "NUT Project Requirement" Section 7.2.3, Paragraph 3, and would like to use port name ups as well as port nut. The project requests that IANA authorise such use, perhaps by updating the Service Name and Transport Protocol Port Number Registry [Registry] for ports ups and nut/3493 to include a reference to this document.

UPS management does not need a system port. If port number 401 were freed and the name ups assigned to user port TBD1, that would be equally effective.

The document shepherd is requested to replace the port number TBD1 by any number that IANA assigns to port name ups.

8.2. Change of Registrant

The NUT Project advises IANA that port ups/401 has no effective registrant, see Section 7.2.2. The project does not have an imperative need to be the registrant but will accept to become the registrant if IANA deems such change desirable. Such a change in registrant could be accompanied by an allocation of a user port number TBD1.

9. Implementation Status

This section presents a very short summary of the status of the Network UPS Tools project

*May 1996: The first hack as a cron job.
*September 1997: The first server-client code.
*March 1998: First public release.
*June 1999: Code rewrite with a UPS driver smartups, an Attachment Daemon upsd and a simple Management Daemon.
*September 1999: The project became "Network UPS Tools". The Management Daemon upsmon supported primary/secondary configurations.
*June 2001: Common core for multiple drivers.
*May 2002: IANA granted port nut/3493. August: release 1.0.0. November: OpenSSL support.
*April 2003: The initial set of command and variable names was designed.
*February 2005: Arnaud Quette took over the project lead from Russell Kroll.
*March 2016: Version 2.7.4 released, supported over 100 device manufacturers and hundreds of UPS models.

*November 2020: Evgeny "Jim" Klimov took over project lead from Arnaud Quette.
*March 2022: Version 2.8.0 released, supporting protocol version 1.3.

See [githist] and [History] for a detailed history of the NUT Project.

9.1. Inclusion in Software Distributions

The programs upsd, upsmon, upssched, upsc, upscmd and upsrw have been included in the package known as "nut" in the package systems of many distributions: all the major Linux distributions, and Unix distributions such as OpenBSD and OpenSolaris. A Microsoft Windows version has been developed but was not maintained.

9.2. Recommended Minimum Support

The features provided by current UPS units vary widely. However experience shows that a minimum feature set is needed for satisfactory use of the NUT Project software. A full list of variables is available in source code file docs/nut-names.txt [gitvars] which serves as the Recording Document.

9.2.1. Desktop PC Variables

The following variables form a minimum set suitable for Desktop PC. It is expected that on public power supply failure, the PC will be halted. It will not restart automatically when power returns.

*battery.charge

*battery.charge.low

*device.mfr

*device.model

*ups.status with the minimum status symbol set OL OB LB FSD, see Section 5.1.

9.2.2. Unattended Servers, Additional Variables

The following additional variables are needed in a minimum set suitable for an unattended server. It is expected that on public power supply failure, the server will be halted. It will restart automatically when power returns.

*battery.date

*device.serial

*ups.delay.shutdown

*ups.delay.start

9.2.3. Commands and other Technical Terms

Satisfactory use of the NUT Project software requires support for all the commands specified in protocol version 1.3, software version 2.8.0. The software should also support the technical terms used in protocol version 1.2, software version 2.7.4. See Appendix C for the differences.

10. Acknowledgments

This document is based on the NUT Project documentation [devguide]. The editor acknowledges the work of Charles Lepple, Arjen de Korte, Arnaud Quette, Jim Klimov, Russell Kroll, Manuel Wolfshant, Greg Troxel, Mark Hansen and many others who contribute to the nut-upsuser [nut-upsuser]. and nut-upsdev [nut-upsdev] mailing lists.

The source for this document is marked up using an SGML DTD [SGML] and an XML meta-DTD as defined by HyTime Annex A [HyTimeA]. Unlike XML, SGML offers markup minimisation, and the source document takes advantage of this. The osgmlnorm [sgmlnorm] program generates XML which program xml2rfc [RFC7991] uses to prepare the HTML and text renderings. The editor acknowledges the help received from Carsten Bormann and Julian Reschke in the xml2rfc mailing list.

The editor thanks the Independent Submissions Editors Adrian Farrel and Eliot Lear for advice received during the preparation of this document. Many helpful comments were received from Bart Smit, David Zomaya, Joyce Norris, Ted Mittelstaedt and Manuel Wolfshant.

11. Normative References

[ISOIEC10646] "ISO/IEC 10646:2020 Information technology - Universal coded character set (UCS)", ISO/IEC Internatioanl Standard ISO/IEC 10646:2020(E), 2804 pages, CHF 195, Technical Committee JTC1/SC2 Coded character sets, December 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/

RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, DOI 10.17487/RFC5198, March 2008, <https://www.rfc-editor.org/info/rfc5198>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <https://www.rfc-editor.org/info/rfc7405>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

12. Informative References

[Bennett] "Charles Bennett Obituary", Publisher: Jagers and Sons Funeral Home, 24 Morris Ave., Athens OH, <https://www.legacy.com/obituaries/name/charles-bennett-obituary?pid=174356861>.

[devguide] "Network UPS Tools (NUT) Project Developer Guide", <https://networkupstools.org/docs/developer-guide.chunked/ar01s09.html>.

[Documentation] "Network UPS Tools Documentation", <https://networkupstools.org/documentation.html>.

[githist] "GitHub Network UPS Tools code repository, project history", <https://github.com/networkupstools/nut/blob/master/docs/history.txt>.

[gitstats] "GitHub Network UPS Tools code repository, status names", <https://github.com/networkupstools/nut/blob/master/clients/status.h>.

[gitvars] "GitHub Network UPS Tools code repository, variable names", <https://github.com/networkupstools/nut/blob/master/docs/nut-names.txt>.

[History] "Network UPS Tools User Manual, Appendix J Project history", <https://networkupstools.org/docs/user-manual.pdf>.

[HyTimeA] "International Standard ISO/IEC 10744 -- Hypermedia/Time-based Structuring Language, Annex A, SGML Extended Facilities", ISO/IEC JTC 1/SC 34 Document description and processing languages, 1997.

[IEC62351-1] "IEC TS 62351-1 Power systems management and associated information exchange -- Data and communications security. Part 1: Communication network and system security -- Introduction to security issues", IEC Technical Specification Reference number IEC/TS 62351-1:2007(E), 35 pages, CHF 205, Technical Committee TC 57 - Power systems management and associated information exchange, 15 May 2007, <https://nanopdf.com/download/technical-iec-specification-ts-62351-1_pdf>.

[Library] "GitHub Network UPS Tools, Devices Dumps Library", <https://networkupstools.org/ddl/>.

[NUT] "Network UPS Tools (NUT) Project", <https://www.networkupstools.org>.

[nut-repository] "GitHub Repository for the Network UPS Tools (NUT) Project", <https://github.com/networkupstools/nut/>.

[nut-upsdev] "Network UPS Tools (NUT) Project Mailing List for developers", <https://alioth-lists.debian.net/cgi-bin/mailman/listinfo/nut-upsdev>.

[nut-upsuser] "Network UPS Tools (NUT) Project Mailing List for users", <https://alioth-lists.debian.net/cgi-bin/mailman/listinfo/nut-upsuser>.

[Registry] "Service Name and Transport Protocol Port Number Registry", Publisher: IANA, <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation

List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC7942] Sheffer, Y. and A. Farrel, "Improving Awareness of Running Code: The Implementation Status Section", BCP 205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://www.rfc-editor.org/info/rfc7942>.

[RFC7991] Hoffman, P., "The "xml2rfc" Version 3 Vocabulary", RFC 7991, December 2016, <https://www.rfc-editor.org/info/rfc7991>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[SGML] Goldfarb, Charles F., "The SGML Handbook", ISBN 0-19-853737-9, 1990.

[sgmlnorm] Clark, James., "SGMLNORM An SGML System Conforming to International Standard ISO 8879 -- Standard Generalized Markup Language", <http://www.jclark.com/sp/sgmlnorm.htm>.

[stunnel] Trojnara, Michal., "Stunnel proxy adds TLS encryption functionality to existing clients and servers", <https://www.stunnel.org/>.

Appendix A. Variables

The UPS variables represent the abstracted state of the UPS unit. Certain variables represent not only the state of some hardware feature, but also provide tunable values and instant commands, see Section 2.5. The full set of variables is recorded in the reference document for variable names [gitvars].

The number of variables used in a given deployment depends on the sophistication of the UPS product: this annex shows a typical example of the subset of variables used for a reasonably complete "consumer grade" UPS. The NUT Project maintains a large library of the variable subsets [Library] used by different UPS models.

Note that successive versions of a given product may add or delete features causing a change in the subset of variables used. An example is the removal of ups.delay.start from a "consumer grade" UPS. The manufacturer reserves the feature for the "professional" product.

An implementation of a Management Daemon acting as a utility program may provide a listing of the variables available for a given

product, for example utility program upsc as included in the NUT package, see Section 2.6, Paragraph 3.

The following sections illustrate the use of variables by taking the values associated with a typical product example of a 1600Va 1000W UPS.

A.1. Typical UPS Variables

Variable	Typical value	Default description
battery.charge	100	"Battery charge (percent of full)"
battery.charge.low	20	"Remaining battery level when UPS switches to LB (percent)"
battery.runtime	1481	"Battery runtime (seconds)"
battery.type	PbAc	"Battery chemistry"
device.mfr	Example Mfg	""
device.model	Economy 1600	""
device.serial	1234567890	""
device.type	ups	""
driver.name	usbhid-ups	"Driver name"
driver.parameter.lowbatt	37	"Driver parameter: <name>"
driver.parameter.offdelay	30	"Driver parameter: <name>"
driver.parameter.ondelay	40	"Driver parameter: <name>"
driver.parameter.pollfreq	30	"Driver parameter: <name>"
driver.parameter.pollinterval	2	"Driver parameter: <name>"
driver.parameter.port	auto	"Driver parameter: <name>"
driver.parameter.synchronous	no	"Driver parameter: <name>"
driver.parameter.vendorid	0999	"Driver parameter: <name>"
driver.version	2.8.0	"Driver version - NUT release"
driver.version.data	HID 1.39	""
driver.version.internal	0.41	"Internal driver version"
input.transfer.high	264	"High voltage transfer point (V)"

Variable	Typical value	Default description
input.transfer.low	184	"Low voltage transfer point (V)"
outlet.1.desc	PowerShare Outlet 1	"Outlet description"
outlet.1.id	2	"Outlet system identifier"
outlet.1.status	on	"Outlet switch status"
outlet.1.switchable	no	"Outlet switch ability"
outlet.2.desc	PowerShare Outlet 2	"Outlet description"
outlet.2.id	3	"Outlet system identifier"
outlet.2.status	on	"Outlet switch status"
outlet.2.switchable	no	"Outlet switch ability"
outlet.desc	Main Outlet	"Outlet description"
outlet.id	1	"Outlet system identifier"
outlet.power	25	""
outlet.switchable	no	"Outlet switch ability"
output.frequency.nominal	50	"Nominal output frequency (Hz)"
output.voltage	230.0	"Output voltage (V)"
output.voltage.nominal	230	"Nominal output voltage (V)"
ups.beeper.status	enabled	"UPS beeper status"
ups.delay.shutdown	20	"Interval to wait after shutdown with delay command (seconds)"
ups.delay.start	30	"Interval to wait before (re)starting the load (seconds)"
ups.firmware	02	"UPS firmware"
ups.load	20	"Load on UPS (percent of full)"
ups.mfr	Example Mfg	"UPS manufacturer"
ups.model	Economy 1600	"UPS model"
ups.power.nominal	1600	"UPS power rating (VA)"
ups.productid	ffff	"Product ID for USB devices"
ups.serial	000000000	"UPS serial number"
ups.status	OL	"UPS status"
ups.temperature	27	"UPS temperature (C)"
ups.timer.shutdown	0	"Time before the load will be shutdown (seconds)"

Variable	Typical value	Default description
ups.timer.start	0	"Time before the load will be started (seconds)"
ups.vendorid	0999	"Vendor ID for USB devices"

Table 7: Typical UPS Variables

A.2. Typical UPS Readable and Writable Variables

Some of the features of a UPS are represented by variables which may be tuned by the user. The following variables are typical of such tunable features. The precise list depends on the model of UPS. An implementation of a Management Daemon acting as a utility program may provide a listing of the variables available, as well as acting on them, for example utility program upsrw as included in the NUT package, see Section 2.6, Paragraph 3.

Variable	Typical value	Default description provided as response to the command GET DESC
battery.charge.low	20	"Remaining battery level when UPS switches to LB (percent)"
input.transfer.high	264	"High voltage transfer point (V)"
input.transfer.low	184	"Low voltage transfer point (V)"
outlet.1.desc	PowerShare Outlet 1	"Outlet description"
outlet.2.desc	PowerShare Outlet 2	"Outlet description"
outlet.2.switchable	no	"Outlet switch ability"
outlet.desc	Main Outlet	"Outlet description"
outlet.power	25	"Description unavailable"
output.voltage.nominal	230	"Nominal output voltage (V)"
ups.delay.shutdown	20	"Interval to wait after shutdown with delay command (seconds)"
ups.delay.start	30	"Interval to wait before (re)starting the load (seconds)"

Table 8: Typical readable and writable UPS Variables

A.3. Typical UPS Instant Commands

Some of the features of a UPS are actions known as instant commands, see Section 2.5, which may be ordered by the user. The following variables represent such instant commands. The precise list depends on the model of UPS. An implementation of a Management Daemon acting as a utility program may provide a listing of the variables available, as well as acting on them, for example utility program upscmd as included in the NUT package, see Section 2.6, Paragraph 3.

Command	Meaning
beeper.disable	Disable the UPS beeper
beeper.enable	Enable the UPS beeper
beeper.mute	Temporarily mute the UPS beeper
load.off	Turn off the load immediately
load.off.delay	Turn off the load with a delay (seconds)
load.on	Turn on the load immediately
load.on.delay	Turn on the load with a delay (seconds)
shutdown.return	Turn off the load and return when power is back
shutdown.stayoff	Turn off the load and remain off
shutdown.stop	Stop a shutdown in progress

Table 9: Typical Instant Commands

Appendix B. The Shutdown Story for System and UPS

This appendix provides background material helpful for a general understanding of the relation between system and UPS. It does not define any feature of the command-response protocol.

We consider the steps involved in the shutdown and restart of a long-running unattended server protected by a single UPS. The Management Daemon runs in the server as shown in Figure 8.

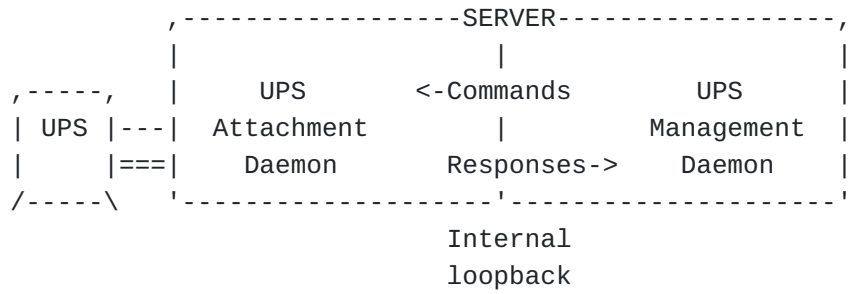

Figure 8: Long-running unattended server

1. *The public power supply is on* -- The system runs normally. Every 5 seconds, variable ups.status reports OL. -- *Days, weeks, months go by...*

2. *Winter storm. Tree falls on power lines. The public power supply fails* -- The server remains operational running on the UPS battery. The Management Daemon polls the Attachment Daemon, and detects status change OL->OB.

3. The Management Daemon logs warning messages. The server is still operational running on the UPS battery. -- *Minutes go by...*

4. The battery discharges below the level specified by variable battery.charge.low. The server remains operational, but the UPS battery will not last much longer. The Management Daemon polls the Attachment Daemon, and detects status change OB->OB+LB.

5. The Management Daemon logs the low battery event.

6. The Management Daemon decides to call for a system shutdown. It sets status FSD in the Attachment Daemon to call on any secondaries to shut down and waits for command GET NUMATTACH to report one single attachment, i.e. the Primary itself. The Management Daemon then issues the system shutdown command for itself.

7. The operating system's shutdown process takes over. During the system shutdown, a NUT Project specific script or an equivalent systemd service unit runs the command upsdrvctl shutdown. This tells the UPS that it is to shut down N seconds later where the default is N=20. Note that the "shutdown" of a UPS removes power from the outlet sockets, but may not turn the UPS off completely. A delayed shutdown is sometimes audible, and the characteristic beeping of the UPS stops.

8. The system shuts down and powers down, hopefully before the N=20 seconds have passed.

9. *N seconds after item 7* -- The UPS shuts down, i.e., it turns off its outlet sockets when N=20 seconds have passed. With some UPS units, there is an audible "clunk".

   What if the public power supply returns before the UPS shuts down? The UPS unit should be able to wait a configurable time with default 30 seconds. These two timers start from the moment the UPS receives the upsdrvctl shutdown command. -- *Minutes, hours, days go by...*

10. *Some time later, maybe much later, the public power supply returns* -- The UPS reconnects it's outlets to send power to the protected system.

11. The system BIOS option "Restore power on AC return" or "Restore to previous state" has hopefully been selected and the system powers up. The bootstrap process of the operating system begins.

12. The operating system starts the Attachment Daemon and the Management Daemon. The Attachment Daemon starts the Driver and scans the UPS. The UPS status becomes OL+LB.

13. After some time, the battery charges above the battery.charge.low threshold and the Attachment Daemon declares the status change OL+LB->OL. We are now back in the same situation as 1 above.

Appendix C. Technical Terms: Historical Differences

This appendix lists the major differences between the technical terms used in NUT software release 2.8.0 and documented in this text, and those used in previous version 2.7.4 of the NUT Project.

Term in previous release NUT 2.7.4	Term in this document, release NUT 2.8.0	Reference
ALREADY-LOGGED-IN	ALREADY-ATTACHED	Table 3
ALREADY-SSL-MODE	TLS-ALREADY-ENABLED	Table 3
LOGIN	ATTACH	Section 4.2.1
LOGOUT	DETACH	Section 4.2.2
Master	Primary	Section 2.7
NETVER	PROTVER	Section 4.2.10
NUMLOGINS	NUMATTACH	Section 4.2.4.3
Slave	Secondary	Section 2.8

Table 10

Appendix D. Security Defences in Release 2.7.4

Previous NUT version 2.7.4 did not provide support for TLS v.3 [RFC8446]. The following subsections describe mitigating techniques.

D.1. Shims

Previous version 2.7.4 of NUT did not support TLS v.3 [RFC8446]. Where such protection is needed for version 2.7.4, a possible technique introduces shims between the Attachment Daemon and the network, and between the network and the Management Daemon as shown in figure 9. These shims provide TLS v.3 support, thus allowing the Attachment Daemon and Management Daemon to continue temporarily without native TLS. The technique has been successfully tested, but the principal difficulty is that the shims make use of a second port which is not currently available.

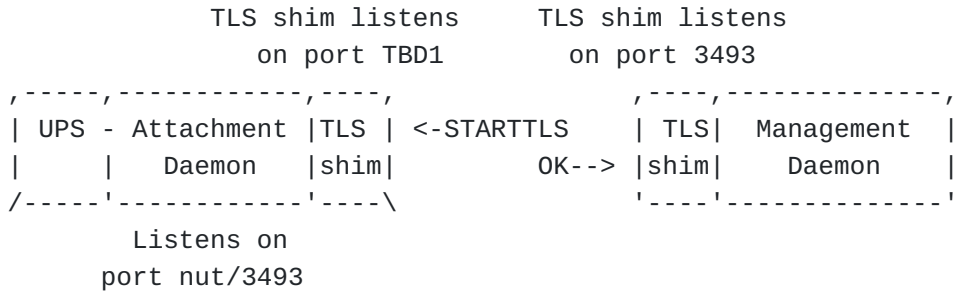

Figure 9: Shims provide TLS support during migration

D.1.1. Attachment Daemon Shim

The shim in front of the Attachment Daemon listens to incoming traffic on a port to be specified. When it receives the command STARTTLS it

1. Returns OK to the client and sets up TLS encapsulation.
2. Does not send STARTTLS to the Attachment Daemon port nut/3493.

All other commands and responses are passed through.

D.1.2. Management Daemon Shim

The shim in front of the Management Daemon listens for incoming traffic on port nut/3493. When it receives the command STARTTLS it

1. Returns FEATURE-NOT-CONFIGURED to the client.
2. Sends STARTTLS to the Attachment Daemon on a port to be specified.

All other commands and responses are passed through.

D.2. TLS Tunnels

Another technique is the use of TLS tunnels [RFC8446], using a software such as stunnel [stunnel] which adds OpenSSL-based TLS support without modifying the Attachment Daemon or Management Daemon. The NUT Project has no procedure to enforce this on sites.

D.3. VPN

A further option to secure communications is very similar to TLS tunnelling [RFC8446] and consists of routing the NUT traffic through a Virtual Private Network, VPN.

D.4. VLAN

A fourth option is to isolate the UPS management traffic at the network switching level using a Virtual LAN, VLAN technique.

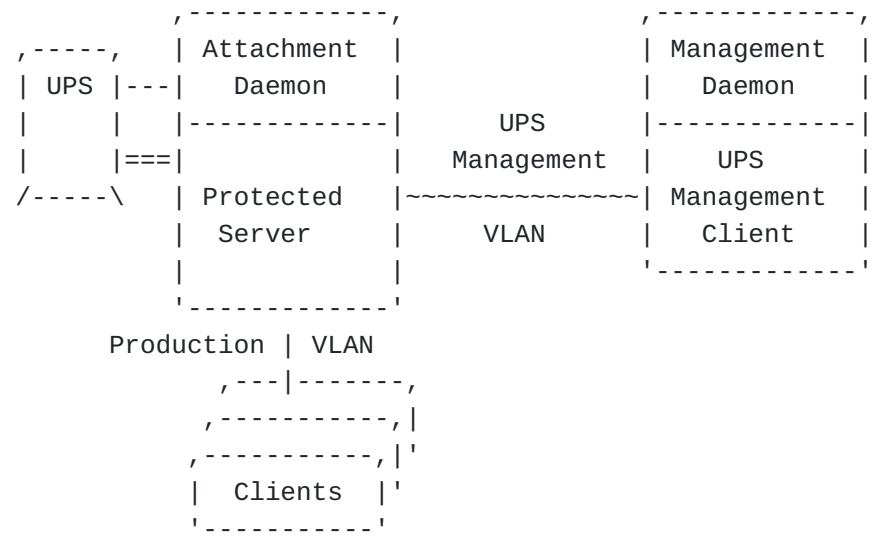

Figure 10: UPS Management Protocol runs over its own VLAN

In Figure 10 there are two VLANS: The main traffic between the protected server and its clients uses the production VLAN. The UPS management traffic between the attachment and management Daemons uses the UPS management VLAN marked as ~~~~~~~~~~~~~.

Appendix E. Administrative Security

Administrative commands such as FSD, INSTCMD and SET are powerful and can have a deep effect on system integrity, For example, the command FSD is involved in mission critical system shutdown decisions. Access to them needs to be managed and restricted. This clause presents the current practice.

E.1. Management of Administrative Users

The Attachment Daemon maintains a file (currently upsd.users) defining each administrative user. Note that these users are independent of those recorded in file /etc/passwd. Each administrative user gets its own section in file upsd.users. The declarations in that section set the parameters which define that user's privileges. The section begins with the name of the user enclosed in square brackets, U+005B LEFT SQUARE BRACKET [and U+005D RIGHT SQUARE BRACKET], and continues until the next user name in brackets or EOF.

For example the following file declares two administrative users admin and pfy:

```
[admin]
    password = sekret
    upsmon master
    actions = SET
    instcmds = ALL
[pfy]
    password = sekret
    instcmds = test.panel.start
    instcmds = test.panel.stop
```

Within each section the administrative user declarations are:

Declaration	Meaning
actions	Allow the user to do certain things in the Attachment Daemon. To specify multiple actions, use multiple instances of the declaration. Valid actions are: *FSD Set the "Forced Shutdown" flag for this UPS. See Section 4.2.3. *SET Change the value of a UPS variable. See Section 4.2.11.
instcmds	Let a user initiate specific instant commands. See Section 4.2.6. Use value ALL to grant all commands automatically. To specify multiple commands, use multiple instances of the instcmds field. For the full list of what a given UPS supports, use client upscmd -l supplied by the NUT Project. The LIST CMD command is issued within the client upscmd.
password	Set the password for this user. *Your password should be more secure than the examples shown.*
upsmon	Add the necessary actions for a Management Daemon to process a system shutdown. In current practice the value is still master or slave. Note that there is no U+003D EQUALS SIGN =.

Table 11: Administrative user declarations

E.2. An Administrative User of a Client Management Daemon

The following examples show the current security practices for administrative users of a client Management Daemon They also illustrate the command pair USERNAME and PASSWORD. See Section 4.2.13 and Section 4.2.8.

E.2.1. An Administrative User Logs into a Short Session

In this simple example of current practice, the system administrator sets the battery level at which an Attachment Daemon will raise the status LB, represented by variable battery.charge.low, to 35% of full charge. A system administrator types the following command to call the client upsrw supplied by the NUT Project.

```
upsrw -s battery.charge.low=35 -u admin -p sekret UPS-1@example.com
```

The USERNAME and PASSWORD commands are issued within the client upsrw and the Session is of short duration.

E.2.1.1. An Administrative User Logs into a Long Session

In this second example of current practice, the long-running Management Daemon upsmon which is responsible for initiating system shutdowns and which is provided by the NUT Project issues commands USERNAME and PASSWORD when it starts up. The data values needed for the USERNAME and PASSWORD commands are provided by a configuration file upsmon.conf which contains the line

```
MONITOR UPS-1@example.com 1 admin sekret master
```

This says that the UPS to be monitored is UPS-1@example.com, it provides 1 single power supply, the administrative user is admin with password sekret. The Management Daemon acts as a Primary, although current practice uses the term master.

The USERNAME and PASSWORD commands are contained within the client upsmon and the Session is of long duration.

Appendix F. Change Log

This section is to be removed before publishing as an RFC.

Ed: To be removed on publication.

F.1. Changes in Version 01

1. There is exactly one newline (4.1) at the end of commands and responses.
2. Added descriptions to variables in Annex (A).
3. Added clause Events (5.2).

F.2. Changes in Version 02

1. Extended acknowledgments.
2. Added reference to possible use of RFC1628 between driver and Attachment Daemon.
3. Clarified response to command LIST CLIENT.

F.3. Changes in Version 03

1. Clarified description of Attachment Daemon.
2. Added Implementation status section as recommended by RFC 7942 [RFC7942].
3. Rewrote Section 7.2.3.
4. Clarified Appendix A as being merely an example of variables used for a specific UPS product.
5. Added definition of <upsname> in Section 4.2.

F.4. Changes in Version 04

There are many changes in this version following the ISE review. See reply to ISE review: http://rogerprice.org/NUT/ISE-comments-2021-06-14.reply.html Among other changes are:

1. Section (7) becomes "Codepoint Management".
2. Editorial cleanup. All <aside> elements labelled as notes.
3. Added implementation note to Section 4.2.7.
4. Error message ALREADY-SSL-MODE becomes ALREADY-TSL-ENABLED.
5. Added error message TSL-NOT-ENABLED.
6. Typo in clause UPS status (Section 2.10).
7. Removed all reference to use of RFC1628 between driver and Attachment Daemon.
8. In Section 4.2 field [:<port>] is always available in <upsname>
9. Added technical term administrative user.
10. Added appendix Technical terms: Historical differences (C)
11. Added table of "successful" responses: Response when command succeeds (4.3.1)
12. Three commands change name LOGIN -> ATTACH, LOGOUT -> DETACH and NUMLOGINS -> NUMATTACH.
13. Error message ALREADY-LOGGED-IN becomes ALREADY-ATTACHED.

F.5. Changes in Version 05

This version includes changes made following comments by the reviewers.

1. Abstract: Change "takes" -> "automates".
2. Abstract: Change "Current practice" -> "Current practice when this text was written".
3. Abstract: Change "leads to" -> "risks".

4. Section (1.1.3), Added sentence on use of "public power supply".
5. Section (1.1), Change "shutting down..." -> "performing a managed shutdown of unattended..."
6. Section (1.3) added link to NUT documentation.
7. Section (2) added "They are listed in alphabetical order."
8. Section (2.1) Change "talks to the UPS" -> "retrieves status from the UPS and sends commands to it".
9. Section (2.1) Change "... launched as system user root and drops privilege ..." -> "... launched as system user root to allow direct access to the hardware (e.g. /proc, /dev). For better security, the daemon then drops privilege ..."
10. Section (2.2) Change "Management Daemon users." -> "Management Daemon users which authenticate to the attachment daemon with basic credentials (username and password)."
11. Section (2.2) Change "... and are defined by a file in the ..." -> "... and are listed in a text file which is read by the ...".
12. Section (2.2) Added link to Appendix E.1.
13. Section (2.3) Change "specific to the hardware" -> "specific to the UPS hardware"
14. Section (2.5) Change "causes the hardware" -> "is passed to the driver and sent to the hardware".
15. Section (2.5) Change "to immediately" -> "without any configured delay to".
16. Section (2.6) Change "the system reaction to power loss." -> "orchestrating system-wide actions after a power event."
17. Section (1.1.1) Title "NUT Project" -> "NUT Software Project".
18. Section (2.7) Change "the system to which the data lead is connected" -> "the computer running the driver".
19. Section (2.8) Replaced and clarified "data lead" not present with secondaries.
20. Section (2.9) Change "may open a session" -> "may initiate a TCP session".
21. Section (2.10) Change "are considered fundamental and are" -> "MUST be".
22. Section (2.10) Change "other statuses depend" -> "other statuses are OPTIONAL and depend".
23. Section (2.11) Change "The features" -> "The metrics and identifiers".
24. Section (2.11) Change "current value attached to that feature" -> "value representing that metric or identifier".
25. Section (2.11) Added a note: "Note: Some variables are constants, e.g. battery type, manufacturer."
26. Section (3) Rewrote paragraph to clarify "the Attachment Daemon and the Management daemon which act as **server** and **client** respectively."

27. Section (3) Change "run the Attachment Daemon (2.1)." -> "run the Attachment Daemon (2.1), thereby effectively creating a network attached UPS running a standard protocol."
28. Figure (4) In the note: replaced "but if the UPS had status OB the Secondary shuts down." by "but if the UPS had status OB the Secondary may choose to shut down as a precaution."
29. Section (4.1) Added scholarly historical note.
30. Section (4.2) Change "of the port" -> "of the TCP port".
31. Section (4.2.1) Change "the count is 1 (the Primary (2.8)) + the number of Secondaries (2.9)" -> "the count is 1 for the primary + the number of secondaries". Change "a trio of" to "three".
32. Section (4.2.3) Clarify that "FSD" means "Forced Shutdown".
33. Section (4.2.3) Change "only to a high-level" -> "only to a privileged".
34. Section (4.2.4) Added prefix GET to all the subcommands.
35. Section (4.2.7) Added prefix LIST to all the subcommands.
36. Section (4.2.7) Change "common container" -> "common".
37. Section (4.2.7) Change "then go off and wait for the response" to "wait".
38. Section (4.2.10) Added note differentiating Section 4.2.10 and Section 4.2.14.
39. Section (4.2.11) Changed "and the UPS model." -> "and the UPS model. Some variables are read-only due to the design of the UPS or its driver."
40. Section (4.2.12) Changed "The choice of TLS version is a matter for site security policy and is not specified in this document." -> "The parameters and versions of cryptographic libraries are those of the Attachment Daemon's underlying OS and are outside the scope of this document."
41. Section (4.2.13) Change "provides facilities to limit access to the UPS unit(s) to which it is attached." -> "limits access to clients whose credentials match those in the file upsd.users. There is no anonymous access."
42. Section (4.2.14) Added note differentiating Section 4.2.14 and Section 4.2.10.
43. Section (5.1) Change "public supply", "wall power" and "input supply" -> "input power supply", nine places.
44. Section (5.1) Remove notes from CHRG and DISCHARG.
45. Section (5.1) OB: Removed "offline".
46. Section (5.2) Change "deduces" -> "detects".
47. Section (5.2) Change "valuable resources" -> "compute and network resources" .
48. Section (5.2) Change "will not have up-to-date information about the UPS status" -> "risks missing short-lived changes in the UPS status"
49. Section (D.1) After "imposed by a wave of the hand" added "it cannot be implemented quickly and without impact to many deployed systems".

50. Section (D.3) Added section.
51. Section (D.4) Added section with figure.
52. Section (7.2.3) Added "Note: In Unix-like systems a port with a number below 1024 is privileged and requires elevated permissions to manage."
53. Section (8.2) Change "but accepts" -> "but will accept".
54. Appendix (A) Change "domestic" -> "consumer grade".
55. Appendix (B) item (7) Change *Note that the "shut down" of a UPS does not turn the UPS off completely. It disconnects the outlet sockets. Such a delayed shutdown is audible since the characteristic beeping of a UPS stops. -> Note that the "shutdown" of a UPS removes power from the outlet sockets, but may not turn the UPS off completely. A delayed shutdown is sometimes audible, and the characteristic beeping of the UPS stops.*

F.6. Changes in Version 06

This version includes changes made following comments by the reviewers.

1. Section (4.2.12) Added sentence to permit unencrypted communication.
2. Added Section (9.2) Recommended Minimum Support
3. Changed "long-running Attachment Daemon upsmon" -> "long-running Management Daemon upsmon". Text now moved to appendix.
4. Appendix (B) item (6). Added clarification of the use of status symbol FSD and command Section 4.2.11 NUMATTACH.
5. Section (3) Figure 2 Changed "a standard protocol" -> "a published protocol".

F.7. Changes in Version 07

This version includes changes made following comments by reviewers at the request of the IETF's Independent Submissions Editor.

1. Section (2.2) Cosmetic change.
2. Section (1.1.1) Added reference to GitHub repository [nut-repository].
3. Section (4.2) Added paragraph to explain that NUT is not case sensitive.
4. Section (4.2.7) Rewrote first sentence: "All the LIST commands..."
5. Section (6.3.3) Rewrote 2nd sentence as follows: The administrative user name and password need protection from sniffing and local malicious users. This is done by encrypting the traffic and properly restricting access to the configuration files. (Text has since been thoroughly reworked.)
6. Section (D.2) Replace SSL by TLS.

7. Section (D.3) Replace SSH by TLS. Remove <aside>.
8. Appendix (A) table (7) Added ups.temperature
9. Appendix (B) The Shutdown Story, BIOS options. I removed wording from Appendix B, Paragraph 4, Item 9 and completed the wording in Appendix B, Paragraph 4, Item 11.

F.8. Changes in Version 08

This version includes further changes made following comments by reviewers at the request of the IETF's Independent Submissions Editor.

F.8.1. Technical changes

1. Section 9, Paragraph 2, Item 9. Arnaud took over in 2005, not 2001. Added reference to project history on GitHub [githist].

F.8.2. Editorial changes following initial ISE comments

1. Removed counter attribute from <xref> to prevent ()s.
2. Removed most content from <xref>.
3. References become more literary and less "hypertext".
4. Most keywords and symbols are no longer clickable.

Author's Address

Roger Price (editor)
Network UPS Tools Project
France

Email: ietf@rogerprice.org