Uninterruptible Power Supply (UPS) Management Protocol -- Commands and Responses
draft-rprice-ups-management-protocol-15

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. The practice current when this protocol was first developed 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 19 November 2022.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.
Internet-Draft      UPS management protocol      May 2022

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1.  Introduction ..................................................  4
   1.1.  Current Practice .......................................  4
         1.1.1.  NUT Software Project  .......................  4
         1.1.2.  The "Shutdown Story" ....................  4
         1.1.3.  How to Read this Document ...............  5
   1.2.  Additional Information ..................................  5
   1.3.  Requirements Language ..................................  5
2.  Terminology ....................................................  5
   2.1.  Administrative User .....................................  5
   2.2.  Attachment Daemon .......................................  6
   2.3.  Driver ..................................................  6
   2.4.  Event ...................................................  6
   2.5.  Instant Command .........................................  6
   2.6.  Management Daemon .......................................  7
   2.7.  Primary ................................................  7
   2.8.  Secondary ..............................................  8
   2.9.  Session ................................................  8
   2.10. UPS Status ...............................................  8
   2.11. UPS Variable ............................................  8
3.  Protocol Overview ...............................................  9
   4.1.  Notation Used in this Specification .................... 11
   4.2.  Commands ................................................. 12
         4.2.1.  ATTACH ......................................... 12
         4.2.2.  DETACH ......................................... 13
         4.2.3.  FSD ............................................. 13
         4.2.4.  GET ............................................. 14
               4.2.4.1.  GET CMDDESC  ......................... 14
               4.2.4.2.  GET DESC  ............................ 14
               4.2.4.3.  GET NUMATTACH .............. 15
               4.2.4.4.  GET TYPE  ............................ 15
               4.2.4.5.  GET UPSDESC .................... 16
1. Introduction

1.1. Current Practice

This document describes UPS management techniques and current UPS management practice published by the NUT (Network UPS Tools) 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 3493/TCP (nut).

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"

"The Shutdown Story", see Appendix B, 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.

1.1.3. How to Read this Document

As a simplification to ease reading, 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. 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.
1.3. 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.

2. Terminology

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

2.1. Administrative User

In current practice, the commands and other functions offered by the Attachment Daemon are made available to a set of users known as Management Daemons. These Management Daemons authenticate to the Attachment Daemon with basic credentials (username and password). Although called "users", the administrative 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.

2.2. 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 8.2 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 has 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.

2.3. 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 are 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 6.
* 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 Primary. 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 8.2 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 8.2 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,

```
|-----|  Attachment Daemon | UPS | UPS |
|     |                   |     |
| UPS |-------------------|     |     |
|     |                   |     |     |
|     |                   |     |     |
```

"The client"

```
|-----|  Management Daemon | Management | UPS |
|     |                   | System     |
| UPS |-------------------| Network    |
```

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.
"The client"

<table>
<thead>
<tr>
<th>UPS</th>
<th>Attachment</th>
<th>&lt;-Commands</th>
<th>UPS</th>
<th>Management</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Daemon</td>
<td>Responses-&gt;</td>
<td>Daemon</td>
<td></td>
</tr>
</tbody>
</table>
/-----\                                   /-----|
| UPS Attachment and Management System     | UPS Management System |

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.

"The client"

<table>
<thead>
<tr>
<th>UPS</th>
<th>Attachment</th>
<th>&lt;-Commands</th>
<th>UPS</th>
<th>Management</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Daemon</td>
<td>Responses-&gt;</td>
<td>Daemon</td>
<td></td>
</tr>
</tbody>
</table>
/-----\                                   /-----|
| UPS Attachment System                  | UPS Management System |

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
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 requirement.
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 US-ASCII, see [RFC0020].

Multi-word elements are contained between QUOTATION MARK characters for easier parsing. E.g., "UPS on fire". Embedded quotation marks are escaped with REVERSE SLANT \ often known as backslashes. Embedded backslashes are also escaped by representing them as \\.

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

ABNF: see variable upsname in Figure 5.

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

* <upsname> ::= [<group>::]<ups>[@<hostname>[:<port>]]
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.
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.

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.

ABNF: See variable get in Figure 5.

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.

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
<table>
<thead>
<tr>
<th>Type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>RW</td>
<td>This is a read/write variable. It may be read with command GET VAR, see</td>
</tr>
<tr>
<td></td>
<td>Section 4.2.4.6, and set to a different value with command SET, see Section 4.2.11.</td>
</tr>
<tr>
<td>ENUM</td>
<td>An enumerated type, which supports specific predetermined values.</td>
</tr>
<tr>
<td>STRING:n</td>
<td>This is a string of maximum length n.</td>
</tr>
<tr>
<td>RANGE</td>
<td>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.</td>
</tr>
<tr>
<td>NUMBER</td>
<td>This is a single numeric value, either integer or float.</td>
</tr>
</tbody>
</table>

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 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

ABNF: See variable help in Figure 5.

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

ABNF: See variable instcmd in Figure 5.

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.

ABNF: See variable list in Figure 5.

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:
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 QUOTATION MARKS 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

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 QUOTATION MARKS 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 8.2 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
QUOTATION MARKS 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 QUOTATION MARKS are
part of the response.
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 QUOTATION MARKS 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 8.2 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.

ABNF: See variable password in Figure 5.

4.2.9. PRIMARY

In current practice, the Attachment Daemon records in local file upsds.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 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:
Notes:

1. There are no QUOTATION MARKS in the response.
2. The version of the protocol returned by PROTVER is different to the implementation version of the Attachment Daemon returned by VER.
3. To ease migration, NUT version 2.8.0 also supports the equivalent NETVER command used in previous releases. See Section 8.2.4.

ABNF: See variable protver in Figure 5.

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 QUOTATION MARKS 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

ABNF: See variable set in Figure 5.

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.

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] and updates. See Appendix D for details of the encryption of communications in previous release 2.7.4.

ABNF: See variable starttls in Figure 5.

4.2.12.1. Key Infrastructure and Self-signed Certificates

_The very restricted nature of UPS management makes it of interest to consider self-signed certificates._

In the World Wide Web, there are millions of servers and hundreds of millions of potential clients for each one. The servers do not know who their clients will be, so they entrust the management of a Public Key Infrastructure (PKI) to Certificate Authorities that they trust, for some value of trust. The encryption of communications between client and server requires that the browsers carry a list of Certificate Authorities which the clients have to trust. _This is a many-to-many relationship._

The management of UPS units is not a many-to-many relationship, it is frequently one-to-one. In the closely restrained world of UPS management, there are a very limited number of clients for each server, rarely more than three, and unlike the World Wide Web the server administrators know exactly who they are. These clients visit very few servers, typically one only. This situation is totally different to the World Wide Web. The use of external Certificate Authorities is a potential security weakness that must be accepted for the World Wide Web, but which can be avoided for UPS management
by either generating locally the private and public keys, or for larger organisations, using a Private Key Infrastructure.

The security policies for UPS management may be subordinate to an organisation's own internal IT security plans and procedures, possibly based on [RFC7030] and [RFC8894], but in simple cases it is possible to obtain better security using self-signed certificates.

4.2.13. USERNAME

The Attachment Daemon limits access to clients whose credentials match those in the file ups.d.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.

ABNF: See variable username in Figure 5.

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 QUOTATION MARKS in the response.
2. The implementation version of the Attachment Daemon returned by VER is different to the protocol version returned by PROTVER.

ABNF: See variable ver in Figure 5.

4.3. Summary of Responses

4.3.1. Response when Command Succeeds

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

<table>
<thead>
<tr>
<th>Command</th>
<th>Response</th>
<th>Reference</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTACH</td>
<td>OK</td>
<td>Section 4.2.1</td>
<td>Was LOGIN</td>
</tr>
<tr>
<td>DETACH</td>
<td>OK Goodbye</td>
<td>Section 4.2.2</td>
<td>Was LOGOUT</td>
</tr>
<tr>
<td>FSD</td>
<td>OK FSD-SET</td>
<td>Section 4.2.3</td>
<td></td>
</tr>
<tr>
<td>GET</td>
<td>Sub command specific</td>
<td>Section 4.2.4</td>
<td></td>
</tr>
<tr>
<td>HELP</td>
<td>List of commands</td>
<td>Section 4.2.5</td>
<td></td>
</tr>
<tr>
<td>INSTCMD</td>
<td>OK</td>
<td>Section 4.2.6</td>
<td></td>
</tr>
<tr>
<td>LIST</td>
<td>Sub command specific</td>
<td>Section 4.2.7</td>
<td></td>
</tr>
<tr>
<td>PASSWORD</td>
<td>OK</td>
<td>Section 4.2.8</td>
<td></td>
</tr>
<tr>
<td>PRIMARY</td>
<td>OK</td>
<td>Section 4.2.9</td>
<td></td>
</tr>
</tbody>
</table>
Table 2: Response if command succeeds

### 4.3.2. Error Responses

Error responses have the following format:

ERR &lt;error-name&gt; [&lt;extra&gt;]

where &lt;error-name&gt; is a single word token taken from the 27 characters A-Z and HYPHEN (MINUS). Implementations MAY if needed add an additional optional &lt;extra&gt;. Current practice does not make use of this possibility.

The &lt;error-name&gt; may have one of the following values:

<table>
<thead>
<tr>
<th>The error name token</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCESS-DENIED</td>
<td>The client's host and/or authentication details supplied by USERNAME and PASSWORD are not sufficient to execute the requested command.</td>
</tr>
<tr>
<td>ALREADY-ATTACHED</td>
<td>The client has already sent a successful ATTACH command for a given UPS and can't do it again.</td>
</tr>
</tbody>
</table>
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.
<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEATURE-NOT-CONFIGURED</td>
<td>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.</td>
</tr>
<tr>
<td>FEATURE-NOT-SUPPORTED</td>
<td>This instance of Attachment Daemon does not support the requested feature. In current practice this error response is possible only for command STARTTLS.</td>
</tr>
<tr>
<td>INSTCMD-FAILED</td>
<td>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.</td>
</tr>
<tr>
<td>INVALID-ARGUMENT</td>
<td>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.</td>
</tr>
<tr>
<td>INVALID-PASSWORD</td>
<td>The client sent a non valid PASSWORD.</td>
</tr>
<tr>
<td>INVALID-USERNAME</td>
<td>The client sent an non valid USERNAME.</td>
</tr>
<tr>
<td>INVALID-VALUE</td>
<td>The value specified in the request is not valid. This usually applies to a SET of an ENUM type which is using a value</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PASSWORD-REQUIRED</td>
<td>The command requires a PASSWORD for authentication, but the client hasn't provided one.</td>
</tr>
<tr>
<td>READONLY</td>
<td>The requested variable in a SET command is not writable.</td>
</tr>
<tr>
<td>SET-FAILED</td>
<td>The Attachment Daemon failed to deliver the SET request to the Driver. This is similar to INSTCMD-FAILED.</td>
</tr>
<tr>
<td>TLS-ALREADY-ENABLED</td>
<td>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.</td>
</tr>
<tr>
<td>TLS-NOT-ENABLED</td>
<td>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.</td>
</tr>
<tr>
<td>TOO-LONG</td>
<td>The requested value in a SET command is too long.</td>
</tr>
<tr>
<td>UNKNOWN-COMMAND</td>
<td>The Attachment Daemon doesn't recognize the command.</td>
</tr>
<tr>
<td>UNKNOWN-UPS</td>
<td>The UPS specified in the request is not known to the Attachment Daemon. This usually means that it didn't match anything in the Attachment Daemon configuration.</td>
</tr>
<tr>
<td>USERNAME-REQUIRED</td>
<td>The command requires a USERNAME for authentication, but the client hasn't provided one.</td>
</tr>
<tr>
<td>VAR-NOT-SUPPORTED</td>
<td>The specified UPS doesn't support the UPS variable in the command.</td>
</tr>
</tbody>
</table>
+-------------------+------------------------------------------------------------------+

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 ASCII [RFC0020].

Current practice tolerates mixed case command names, but it is RECOMMENDED to use upper case only for commands. See Figure 5.

;-------------------------------------------------------------------
; This grammar is case sensitive. Terminal keywords SHOULD be written in upper case as shown.
; The following basic rules written with upper case names are taken from RFC5234 Appendix B.1.

SP = 1*%x20 ; At least one SPACE
LF = %x0A ; Linefeed
DIGIT = %x30-39 ; Digit 0 through 9
ALPHA = %x41-5A / %x61-7A ; A-Z / a-z
DQUOTE = %x22 ; Double quote 
VCHAR = %x21-7E ; Visible (printing) characters

; Additional basic rules needed by this grammar
LC = %x61-7A ; Letter a through z
DOT = 1%x2E ; Exactly one .
COLON = 1%x3A ; Exactly one : 
AT = 1%x40 ; Exactly one @
SEP = 1"-" / 1"_" / 1"." ; A single - or _ or .
JOIN = COLON / AT ; A single : or @

; Frequently used in this grammar
cmdname = 1*LC *62(DOT 1*LC) ; E.g. load.off.delay
upschar = DIGIT / ALPHA / SEP
ups = 1*ALPHA *62upschar ; E.g. Example-Mfg-999
group = ups ; E.g. HB (Not in common use)
hostname = ups ; E.g. example.com
port = 1*5DIGIT ; E.g. 3493
upsname = [group COLON] ups [AT hostname [COLON port]] ; Fully Qualified UPS name ; E.g. HB:heartbeat1@example.com:3493

Price                   Expires 19 November 2022               [Page 31]

Internet-Draft           UPS management protocol                May 2022

username = ups ; E.g. Power-Dept.6
varname = 1*LC *62( DOT 1*(DIGIT / LC) ) ; E.g. outlet.1.status

commandLine = command LF ; LF is a single %x0A
command = attach / detach / fsd / get / help / instcmd /
          list / password / primary / protver / set /
          starttls / username / ver

attach = "ATTACH" SP upsname

detach = "DETACH"

fsd = "FSD" SP upsname

get = "GET" SP getsubcommand
getsubcommand = getcmddesc / getdesc / getnumattach /
                gettype / getupsdesc / getvar

getcmddesc = "CMDDESC" SP upsname SP cmdname
getdesc = "DESC" SP upsname SP varname
getnumattach = "NUMATTACH" SP upsname
gettype = "TYPE" SP upsname SP varname
getupsdesc = "UPSDESC" SP upsname
getvar = "VAR" SP upsname SP varname

help = "HELP"

instcmd = "INSTCMD" SP upsname SP cmdname

list = "LIST" listsubcommand
listsubcommand = listclient / listcmd / listenum / listrange /
                listrw / listups / listvar

listclient = "CLIENT" SP upsname
listcmd = "CMD" SP upsname
listenum = "ENUM" SP upsname SP varname
listrange = "RANGE" SP upsname SP varname
listrw = "RW" SP upsname
listups = "UPS"
listvar = "VAR" SP upsname

password = "PASSWORD" SP *63VCHAR
; A sequence of printable characters defined in a server configuration file. Local
; security practices may mandate a minimum and maximum number of characters.

Figure 5: ABNF for the Commands

Notes:

1. _Implementation note:_ The ABNF is written using the provisions of [RFC5234] [RFC7405] which are US-ASCII based [RFC0020].

2. The grammar is case sensitive. The terminal key words SHOULD be written in upper case as specified.

3. The repetition factor in front of an expression has the form <min>*<max> where <min> is the minimum number of repetitions and <max> is the maximum number.
4. If <min> is omitted its value is 0. If <max> is omitted, its value is infinity.

5. The notation n*n meaning "exactly n copies" may be written as n.

6. Square brackets around an expression mean that the expression is optional. This could be written as 0*1.

4.4.1. Responses to Commands

The responses to the commands are encoded in US-ASCII [RFC0020] and fall into two groups:

1. Short replies to action commands, see Section 4.3.

2. Long replies to requests for information. In this case the reply is sent in a sequence of messages. The last message will contain a line beginning END LIST. See for example Section 4.2.7.1.

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 such as VAR su700 ups.status "OB LB" to discover changes in the UPS status. These changes will indicate UPS events.

+================================+=
| Status     | 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 |
| **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 its 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.

<table>
<thead>
<tr>
<th>OVER</th>
<th>Overloaded. The UPS reports that the load on it is beyond its normal operating maximum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>RB</td>
<td>Replace battery. The UPS reports that its battery/batteries should be replaced.</td>
</tr>
<tr>
<td>TEST</td>
<td>Under test. The UPS is currently undergoing a test, which may have been called for manually or internally.</td>
</tr>
<tr>
<td>TICK</td>
<td>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.</td>
</tr>
<tr>
<td>TOCK</td>
<td>Heartbeat. See TICK</td>
</tr>
<tr>
<td>TRIM</td>
<td>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.</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Old status</th>
<th>New status</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>ALARM</td>
<td>Alarm on</td>
</tr>
<tr>
<td>none</td>
<td>BOOST</td>
<td>Boosting voltage</td>
</tr>
<tr>
<td>none</td>
<td>BYPASS</td>
<td>Bypass on</td>
</tr>
<tr>
<td>none</td>
<td>CAL</td>
<td>Calibrating</td>
</tr>
<tr>
<td>none</td>
<td>CHRG</td>
<td>Charging</td>
</tr>
<tr>
<td>none</td>
<td>COMM</td>
<td>UPS communicating</td>
</tr>
<tr>
<td>none</td>
<td>DISCHRG</td>
<td>Discharging</td>
</tr>
<tr>
<td>none</td>
<td>FSD</td>
<td>System shutdown</td>
</tr>
<tr>
<td>none</td>
<td>LB</td>
<td>Low battery</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Old status</th>
<th>New status</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALARM</td>
<td>none</td>
<td>Alarm off</td>
</tr>
<tr>
<td>BOOST</td>
<td>none</td>
<td>Not boosting</td>
</tr>
<tr>
<td>BYPASS</td>
<td>none</td>
<td>Bypass off</td>
</tr>
<tr>
<td>CAL</td>
<td>none</td>
<td>Not calibrating</td>
</tr>
<tr>
<td>CHRG</td>
<td>none</td>
<td>Not charging</td>
</tr>
<tr>
<td>COMM</td>
<td>none</td>
<td>See note 4</td>
</tr>
<tr>
<td>DISCHRG</td>
<td>none</td>
<td>Not discharging</td>
</tr>
<tr>
<td>FSD</td>
<td>none</td>
<td>Shutdown abandoned</td>
</tr>
<tr>
<td>LB</td>
<td>none</td>
<td>Battery not low</td>
</tr>
</tbody>
</table>

Price
Expires 19 November 2022

[Page 36]
<table>
<thead>
<tr>
<th>Event</th>
<th>Status</th>
<th>Description</th>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWBATT</td>
<td>none</td>
<td>UPS dead?</td>
<td>NOCOMM</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Events COMMBAD, NOCOMM)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>UPS turned off</td>
<td>OFF</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Event ONLINE)</td>
<td></td>
<td>UPS not turned off</td>
</tr>
<tr>
<td></td>
<td>OB</td>
<td>Receiving power</td>
<td>OL</td>
<td>OB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Event ONLINE)</td>
<td></td>
<td>Power lost</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>OVER</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>OVER</td>
<td>UPS overloaded</td>
<td></td>
<td>Overload gone</td>
</tr>
<tr>
<td></td>
<td>RB</td>
<td>Replace battery</td>
<td>RB</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Event REPLBATT)</td>
<td></td>
<td>Replacement canceled</td>
</tr>
<tr>
<td></td>
<td>TEST</td>
<td>Test starts</td>
<td>TEST</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Test finished</td>
</tr>
<tr>
<td></td>
<td>TICK</td>
<td>Heartbeat event. See note 3</td>
<td>TICK</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No heartbeat. See note 3</td>
</tr>
<tr>
<td></td>
<td>TOCK</td>
<td>Heartbeat event. See note 3</td>
<td>TOCK</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No heartbeat. See note 3</td>
</tr>
<tr>
<td></td>
<td>TRIM</td>
<td>Trimming voltage</td>
<td>TRIM</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Not trimming</td>
</tr>
</tbody>
</table>

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


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 activity, part of site management. 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 3493/TCP (nut) 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.

\[
\text{Listens on port 3493/TCP for localhost}
\]

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.

\[
\text{Listens on port 3493/TCP}
\]

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 IEC Technical Specification 62351 [IEC62351-1]. In addition to equipment security, cyber security is now an essential consideration.

Quoting from IEC 62351-1[IEC62351-1], Introduction to the standard, clause 5.2.3.5:

| 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.


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 insertion, 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.

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.

6.2.1. Certificate security

In long-lived installations such as those found in UPS management, careful certificate management is essential, whether the certificate is provided by a Certificate Authority, or is a self-signed certificate. For example the specification of expiration times of both the certificate containing the public key and the signing certificate.

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 1.3 [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
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 malicious client acting as an Management Daemon to send command FSD to an Attachment Daemon to shut down a working system and its power supply as described in The Shutdown Story, see Appendix B. Similarly, a malicious client 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. IANA Considerations

The protocol specified by this text runs over port 3493/TCP (nut) registered by the NUT (Network UPS Tools) project.

This document will be added to the registration's reference field.

8. 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.
* 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.
* April 2003: The initial set of command and variable names was designed.

See [githist] and [History] for a detailed history of the NUT Project.
# 8.1. 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.

# 8.2. 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.

## 8.2.1. 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.

## 8.2.2. 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

8.2.3. 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.

8.2.4. Support for Earlier Versions

In order to ease migration from software version 2.7.4 which supported protocol version 1.2, software version 2.8.0 also supports the technical terms used in protocol version 1.2. See Appendix C for the differences.

9. 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.

Many helpful comments were received from Eliot Lear, Bart Smit, David Zomaya, Joyce Norris, and Ted Mittelstaedt.

10. Normative References
11. Informative References

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

[Documentation]
"Network UPS Tools Documentation",

[githist] "GitHub Network UPS Tools code repository, project history",

[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",
Internet-Draft            UPS management protocol          May 2022


[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>.

Price                   Expires 19 November 2022               [Page 46]

Internet-Draft           UPS management protocol                May 2022


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. The example is a 1600Va 1000W UPS.

A.1. Typical UPS Variables

| +==========================================|=================|-----------------|}
<p>| | Variable | Typical | Default | value description | description |</p>
<table>
<thead>
<tr>
<th>battery.charge</th>
<th>100</th>
<th>&quot;Battery charge (percent of full)&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>battery.charge.low</td>
<td>20</td>
<td>&quot;Remaining battery level when UPS switches to LB (percent)&quot;</td>
</tr>
<tr>
<td>battery.runtime</td>
<td>1481</td>
<td>&quot;Battery runtime (seconds)&quot;</td>
</tr>
<tr>
<td>battery.type</td>
<td>PbAc</td>
<td>&quot;Battery chemistry&quot;</td>
</tr>
<tr>
<td>device.mfr</td>
<td>Example Mfg</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>device.model</td>
<td>Economy 1600</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>device.serial</td>
<td>1234567890</td>
<td>&quot;&quot;</td>
</tr>
</tbody>
</table>

<p>| device.type       | ups | &quot;&quot; |
| driver.name       | usbhid-ups | &quot;Driver name&quot; |
| driver.parameter.lowbatt | 37 | &quot;Driver parameter: &lt;name&gt;&quot; |
| driver.parameter.offdelay | 30 | &quot;Driver parameter: &lt;name&gt;&quot; |
| driver.parameter.ondelay  | 40 | &quot;Driver parameter: &lt;name&gt;&quot; |
| driver.parameter.pollfreq | 30 | &quot;Driver parameter: &lt;name&gt;&quot; |
| driver.parameter.pollinterval | 2 | &quot;Driver parameter: &lt;name&gt;&quot; |</p>
<table>
<thead>
<tr>
<th>driver.parameter.port</th>
<th>auto</th>
<th>&quot;Driver parameter: &lt;name&gt;&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>driver.parameter.synchronous</td>
<td>no</td>
<td>&quot;Driver parameter: &lt;name&gt;&quot;</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>driver.parameter.vendorid</td>
<td>0999</td>
<td>&quot;Driver parameter: &lt;name&gt;&quot;</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>driver.version</td>
<td>2.8.0</td>
<td>&quot;Driver version: NUT release&quot;</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>driver.version.data</td>
<td>HID 1.39</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>driver.version.internal</td>
<td>0.41</td>
<td>&quot;Internal driver version&quot;</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>input.transfer.high</td>
<td>264</td>
<td>&quot;High voltage transfer point (V)&quot;</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>input.transfer.low</td>
<td>184</td>
<td>&quot;Low voltage transfer point (V)&quot;</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>outlet.1.desc</td>
<td>PowerShare</td>
<td>&quot;Outlet description&quot;</td>
</tr>
<tr>
<td>outlet.1.id</td>
<td>2</td>
<td>&quot;Outlet system identifier&quot;</td>
</tr>
<tr>
<td>outlet.1.status</td>
<td>on</td>
<td>&quot;Outlet switch status&quot;</td>
</tr>
<tr>
<td>outlet.1.switchable</td>
<td>no</td>
<td>&quot;Outlet switch ability&quot;</td>
</tr>
<tr>
<td>outlet.2.desc</td>
<td>PowerShare</td>
<td>&quot;Outlet description&quot;</td>
</tr>
<tr>
<td>outlet.2.id</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Price Expires 19 November 2022
Internet-Draft UPS management protocol May 2022
<p>| outlet.2.id | 3 | &quot;Outlet system identifier&quot; |
| outlet.2.status | on | &quot;Outlet switch status&quot; |
| outlet.2.switchable | no | &quot;Outlet switch ability&quot; |
| outlet.desc | Main Outlet | &quot;Outlet description&quot; |
| outlet.id | 1 | &quot;Outlet system identifier&quot; |
| outlet.power | 25 | &quot;&quot; |
| outlet.switchable | no | &quot;Outlet switch ability&quot; |
| output.frequency.nominal | 50 | &quot;Nominal output frequency (Hz)&quot; |
| output.voltage | 230.0 | &quot;Output voltage (V)&quot; |
| output.voltage.nominal | 230 | &quot;Nominal output voltage (V)&quot; |
| ups.beeper.status | enabled | &quot;UPS beeper status&quot; |
| ups.delay.shutdown | 20 | &quot;Interval to wait after shutdown with delay command (seconds)&quot; |
| ups.delay.start | 30 | &quot;Interval to wait before (re)starting the load (seconds)&quot; |</p>
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ups.firmware</td>
<td>02</td>
<td>&quot;UPS firmware&quot;</td>
</tr>
<tr>
<td>ups.load</td>
<td>20</td>
<td>&quot;Load on UPS (percent of full)&quot;</td>
</tr>
<tr>
<td>ups.mfr</td>
<td>Example Mfg</td>
<td>&quot;UPS manufacturer&quot;</td>
</tr>
<tr>
<td>ups.model</td>
<td>Economy 1600</td>
<td>&quot;UPS model&quot;</td>
</tr>
<tr>
<td>ups.power.nominal</td>
<td>1600</td>
<td>&quot;UPS power rating (VA)&quot;</td>
</tr>
<tr>
<td>ups.productid</td>
<td>ffff</td>
<td>&quot;Product ID for USB devices&quot;</td>
</tr>
<tr>
<td>ups.serial</td>
<td>0000000000</td>
<td>&quot;UPS serial number&quot;</td>
</tr>
<tr>
<td>ups.status</td>
<td>OL</td>
<td>&quot;UPS status&quot;</td>
</tr>
<tr>
<td>ups.temperature</td>
<td>27</td>
<td>&quot;UPS temperature (C)&quot;</td>
</tr>
<tr>
<td>ups.timer.shutdown</td>
<td>0</td>
<td>&quot;Time before the load will be shutdown (seconds)&quot;</td>
</tr>
<tr>
<td>ups.timer.start</td>
<td>0</td>
<td>&quot;Time before the load will be started (seconds)&quot;</td>
</tr>
<tr>
<td>ups.vendorid</td>
<td>0999</td>
<td>&quot;Vendor ID for USB devices&quot;</td>
</tr>
</tbody>
</table>

Table 6: 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.
<table>
<thead>
<tr>
<th>Variable</th>
<th>Typical value</th>
<th>Default description provided as response to the command GET DESC</th>
</tr>
</thead>
<tbody>
<tr>
<td>battery.charge.low</td>
<td>20</td>
<td>&quot;Remaining battery level when UPS switches to LB (percent)&quot;</td>
</tr>
<tr>
<td>input.transfer.high</td>
<td>264</td>
<td>&quot;High voltage transfer point (V)&quot;</td>
</tr>
<tr>
<td>input.transfer.low</td>
<td>184</td>
<td>&quot;Low voltage transfer point (V)&quot;</td>
</tr>
<tr>
<td>outlet.1.desc</td>
<td>PowerShare Outlet 1</td>
<td>&quot;Outlet description&quot;</td>
</tr>
<tr>
<td>outlet.2.desc</td>
<td>PowerShare Outlet 2</td>
<td>&quot;Outlet description&quot;</td>
</tr>
<tr>
<td>outlet.2.switchable</td>
<td>no</td>
<td>&quot;Outlet switch ability&quot;</td>
</tr>
<tr>
<td>outlet.desc</td>
<td>Main Outlet</td>
<td>&quot;Outlet description&quot;</td>
</tr>
<tr>
<td>outlet.power</td>
<td>25</td>
<td>&quot;Description unavailable&quot;</td>
</tr>
<tr>
<td>output.voltage.nominal</td>
<td>230</td>
<td>&quot;Nominal output voltage (V)&quot;</td>
</tr>
<tr>
<td>ups.delay.shutdown</td>
<td>20</td>
<td>&quot;Interval to wait after shutdown with delay command (seconds)&quot;</td>
</tr>
<tr>
<td>ups.delay.start</td>
<td>30</td>
<td>&quot;Interval to wait before (re)starting the load (seconds)&quot;</td>
</tr>
</tbody>
</table>

Table 7: 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.

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

---

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 its 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.

<table>
<thead>
<tr>
<th>Term in previous release NUT 2.7.4</th>
<th>Term in this document, release NUT 2.8.0</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALREADY-LOGGED-IN</td>
<td>ALREADY-ATTACHED</td>
<td>Table 3</td>
</tr>
<tr>
<td></td>
<td>TLS-ALREADY-ENABLED</td>
<td>Table 3</td>
</tr>
<tr>
<td>LOGIN</td>
<td>ATTACH</td>
<td>Section 4.2.1</td>
</tr>
<tr>
<td>LOGOUT</td>
<td>DETACH</td>
<td>Section 4.2.2</td>
</tr>
</tbody>
</table>
Appendix D. Security Defences in Release 2.7.4

Previous NUT version 2.7.4 did not provide support for TLS 1.3 [RFC8446]. The following subsections describe mitigating techniques.

D.1. Shims

Previous version 2.7.4 of NUT did not support TLS 1.3 [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 1.3 support, thus allowing the Attachment Daemon and Management Daemon to continue temporarily without native TLS. The technique has been successfully tested.

<table>
<thead>
<tr>
<th>UPS - Attachment Daemon</th>
<th>TLS shim listens on port TBD1/TCP</th>
<th>TLS shim listens on port 3493/TCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPS - Attachment</td>
<td>TLS</td>
<td>&lt;-STARTTLS</td>
</tr>
<tr>
<td></td>
<td>Daemon</td>
<td>shim</td>
</tr>
</tbody>
</table>

Listens on
port 3493/TCP

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 port TBD1/TCP. 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 3493/TCP.

All other commands and responses are passed through.

Note: Port TBD1/TCP is not specified by this text.

D.1.2. Management Daemon Shim

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

1. Returns FEATURE-NOT-CONFIGURED to the client.
2. Sends STARTTLS to the Attachment Daemon on port TBD1/TCP.

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

![Diagram of UPS management protocol and VLANs](image)

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, OPENING BRACKET [ and CLOSING 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:
<table>
<thead>
<tr>
<th>Declaration</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>actions</strong></td>
<td>Allow the user to do certain things in the Attachment Daemon. To specify multiple actions, use multiple instances of the declaration. Valid actions are:</td>
</tr>
<tr>
<td></td>
<td>* FSD Set the &quot;Forced Shutdown&quot; flag for this UPS. See Section 4.2.3.</td>
</tr>
<tr>
<td></td>
<td>* SET Change the value of a UPS variable. See Section 4.2.11.</td>
</tr>
<tr>
<td><strong>instcmds</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>password</strong></td>
<td>Set the password for this user. Your password should be more secure than the examples shown._</td>
</tr>
<tr>
<td><strong>upsmon</strong></td>
<td>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 EQUALS =.</td>
</tr>
</tbody>
</table>
Table 10: 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.

<table>
<thead>
<tr>
<th>Price</th>
<th>Expires 19 November 2022</th>
<th>[Page 61]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet-Draft</td>
<td>UPS management protocol</td>
<td>May 2022</td>
</tr>
</tbody>
</table>

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
```

Option `-s` specifies the variable and the value, option `-u` specifies the `USERNAME`, option `-p` specifies the `PASSWORD`, and `UPS-1@example.com` is the UPS. The `USERNAME` and `PASSWORD` commands are issued within the client `upsrw` and the Session is of short duration.

Note: Your password should be stronger than the example shown.

E.2.2. 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 still uses the former term master.

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

Author's Address

Roger Price (editor)
Network UPS Tools Project
France
Email: ietf@rogerprice.org