REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
2.8.4.1058 2.8.4.1064- 1064+g62710feed	2026-01-03	Current release snapshot of Network UPS Tools (NUT).	
2.8.4	2025-08-07	Fixed a few regressions introduced by release v2.8.3. Some changes to docs and recipes, especially for parallel builds. Updated NUT SEMVER definition some more. Some rounds of code-hardening project. Numerous driver updates, some new ones introduced.	JK
2.8.3	2025-04-07	Some changes to docs and recipes, libupsclient API and functionality. Updated NUT SEMVER definition and added scripting around it. Groundwork for vendor-defined status and INSTCMD buzzwords like "ECO". Fixed some regressions and added improvements for certain new device series.	JK
2.8.2	2024-04-01	Some changes to docs and recipes, libnutscan API and functionality. Added nutconf (library and tool). Fixed some regressions and added improvements for certain new device series.	JK
2.8.1	2023-10-31	Some changes to API, docs and recipes, in particular to simplify local builds and tests (e.g. to help end-users check if current NUT codebase trunk has already fixed an issue they see with a packaged installation). Revived NUT for Windows effort, further improved other OS integrations. NUT became reference for "UPS management protocol", Informational RFC 9271. Documentation files refactored to ease maintenance. More drivers and new driver categories introduced.	JK

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
2.8.0	2022-04-26	Change of maintainer. Many changes to API, docs (both style and content), and recipes, with a stress on non-regression test-ability, run-time debug-ability, general codebase maintainability, as well as OS integrations (notably nut-driver-enumerator for systemd and SMF service instance maintenance). Added a lot in area of CI support and documented pre-requisite package lists for numerous platforms, and CI agent set-up. Added libusb-1.x support and many new driver categories (and drivers), and daisychain device connection support. Instant commands enhanced with TRACKING to enable protocol-based waiting for completion of a particular INSTCMD or SET operation.	JK
2.7.4	2016-03-09	NUT variables namespace updated, in particular for outlet groups, alarms and thresholds, ATS devices, and battery.charger.status to supersede CHRG and DISCHRG flags published in ups.status readings. NUT network protocol extended with NUMBER type; some API changes.	AQ
2.7.3	2015-04-22	Documentation revised, including some API changes. Added NUT DDL links. NUT variables namespace updated.	AQ
2.7.2	2014-04-17	The nut-website project was offloaded into a separate repository. FreeDesktop HAL support was removed (obsoleted in GNOME consumer). Introduced nutdrv_atcl_usb driver.	AQ
2.7.1	2013-11-19	NUT source codebase migrated from SVN to Git (and from Debian infrastructure to GitHub source code hosting). jNut binding split into a separate project. Introduced libnutclient (C++ binding), al175, apcupsd-ups and nutdrv_qx drivers, Mozilla NSS support for simpler licensing than OpenSSL, and a newer apcsmart implementation. Documentation support enhanced with a spell checker, contents massively updated to reflect project changes.	CL

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME
2.6.5	2012-08-08	New macosx-ups driver, new implementation of mge-shut driver. NUT variables namespace updated. Docs cleaned up and revised.	AQ
2.6.4	2012-05-31	New NUT network protocol commands (LIST CLIENTS, LIST RANGE and NETVER), and socket protocol commands (ADDRANGE, DELRANGE). NUT variables namespace updated. Introduced nut-recorder tool.	AQ
2.6.3	2012-01-04	No substantial changes to documentation.	AQ
2.6.2	2011-09-15	Introduced nut-scanner tool and nut-ipmipsu driver, systemd support, and a new apcsmart implementation.	AQ
2.6.1	2011-06-01	Introduced default.* and override.* optional settings in ups.conf, an ups.efficiency report, and outlet.0 special handling.	AQ
2.6.0	2011-01-14	First release of AsciiDoc documentation for Network UPS Tools (NUT).	AQ

Contents

1 I just upgraded, and . . .

You have read UPGRADING in the base directory of the distribution, right?

If not, go read it now, then come back to this file if your question wasn't answered in there.

2 My UPS driver now says it's *broken*, and won't start. What now?

Or a variation like. . .

3 My favorite UPS driver disappeared after an upgrade. What now?

Drivers are occasionally removed from the tree if they are no longer receiving maintenance, or sometimes renamed to better reflect their hardware support scope or replaced by a more generic driver. There have been several architectural changes to the driver code in recent times, and drivers which were not converted by someone are eventually dropped.

This is called progress. We do this in order to avoid a situation where someone believes that a driver is being maintained when it is actually rotting slowly in the tree. It also keeps the tree free of old compatibility hacks for code that nobody actually uses anyway.

To get a driver back into current releases, you need to convert it yourself or get someone to do it for you. This is not difficult. The hardest part of any driver is decoding the protocol, and that's already been done in the old version.

4 My UPS driver program won't work. I'm starting it as root, and root owns the device, so what's the problem?

Answer 1

The drivers drop root privileges long before the serial or USB port is opened. You'll need to change the permissions on that port so that their new user id can access it. Normally this is *nobody*, but it may be changed at compile-time by using `configure --with-user` or in the `ups.conf` file for driver settings with `user=...`.

Read the error message. If you have a permissions mismatch, then you'll see something like this:

```
Network UPS Tools - APC Smart protocol driver 0.60 (1.1.7)
This program is currently running as youruid (UID 1234)
/dev/ttyS2 is owned by user root (UID 0), mode 0600
Change the port name, or fix the permissions or ownership
of /dev/ttyS2 and try again.
Unable to open /dev/ttyS2: Permission denied
```

Now is a good time to point out that using *nobody* is a bad idea, since it's a hack for NFS access. You should create a new role account (perhaps called *ups* or *nut*), and use that instead.

Also, scroll down to the "security domains" question to see an even better way of restricting privileged operations. Neither the drivers nor `upsd` ever need *root* powers, and that answer tells you how to make it work.

Answer 2

You can also specify a user with `user=` in the global part of ups.conf(5). Just define it before any of your `[sections]`:

```
        user = nut

        [myups]
                driver = mge-shut
                port = /dev/ttyS0
```

5 upsc, upsstats, and the other clients say *access denied*. The device communication port (serial, USB or network) permissions are fine, so what gives?

In this case, "access denied" means the access to upsd(8), not the device communication port. You're being denied since the system has no permission to speak to `upsd` according to its access controls.

There can be due to various reasons. To fix it, check:

- the LISTEN directive in upsd.conf(5). It should allow your local or remote access method,
- your firewall rules. Port *3493/tcp* must be opened to incoming connections,
- your **tcp-wrappers** configuration (`hosts.allow` and `hosts.deny`) if used.

Refer to the upsd(8) and upsd.conf(5) man pages for more information.

6 I get a *not listening on...* error from upsd.

Verify your LISTEN directive. It should be one of the valid IP addresses for the computer running `upsd` (or `0.0.0.0`, which is `INADDR_ANY`), not an address for a client.

The LISTEN directive lets you pick which interface `upsd` listens on. If you are trying to limit the clients which can connect to `upsd`, you either need to use tcp-wrappers or kernel firewall rules.

This isn't a NUT-specific limitation — it applies equally to your web server or mailer daemon.

7 Which UPS should I buy?

One with a no-questions-asked money-back guarantee. Seriously. The NUT developers cannot take responsibility for recommending an UPS (see the LICENSE file for more details on the explicit lack of warranty), only to find out that the manufacturer has changed the internals of the UPS without changing the model name.

That said, from time to time, certain vendors have helped out by providing hardware for testing, results of their testing efforts, or protocol specifications. We try to publish this information on the NUT website, so you can take this into consideration when selecting an UPS brand.

8 I have an APC Smart-UPS connected with a grey APC serial cable and it won't work.

The Back-UPS type in the genericups driver works but then I don't get to use all the nifty features in there. Why doesn't the right driver work?

The problem lies in your choice of cable. APC's grey cables generally only do "dumb" signalling — very basic yes/no info about the battery and line status. While that is sufficient to detect a low battery condition while on battery, you miss out on all the goodies that you paid for.

Note that the 940-0095B happens to be a grey cable, but it is actually a dual mode cable and can be used in smart mode. If you have this cable, you need to edit your ups.conf to look like this:

```
[myups]
        driver = apcsmart
        port = /dev/whatever
        cable = 940-0095B
```

All other grey cables from APC are assumed to be "dumb".

If your grey cable isn't the 940-0095B, the solution is to dump that cable and find one that supports APC's "smart" signalling. Typically these come with the UPS and are black. If your smart cable has wandered off, one can be built rather easily with some connectors and cable — there's no fancy wiring or resistors.

See this URL for a handy diagram: https://www.networkupstools.org/cables/940-0024C.jpg

There is also a text version of that diagram in the docs/cables directory of the NUT source distribution. Either one should allow you to build a good clone of APC's 940-0024C cable.

There are simpler solutions involving 3 wires that work just fine too, but Powerchute won't find the loopback DTR-DCD and RTS-CTS and will be annoyed. If you don't ever plan to use Powerchute, 3 wires (RxD, TxD, GND) are sufficient.

It should also be noted that the genericups driver has no way to detect the UPS, so it will fire up quite happily if it can open the serial port. Merely having it start up is not necessarily an indication of success. You should start it and then check the status with upsc or similar to be sure that it's reading the hardware properly.

9 Why doesn't upsd implement the functionality of upsmon? I have to run THREE programs to monitor my UPS!

Answer 1

We try to follow the "tool for the job" philosophy of Unix. It may mean more programs running, but the flexibility you get is usually worth it.

Yes, the machine with the UPS attached will generally have 3 processes (driver, `upsd`, `upsmon`) running, but this design allows a much bigger setup. Imagine a data room with a bunch of machines all drawing power from the same UPS. Only one can be connected by a serial or USB cable, and the rest of them just run an `upsmon` client.

Besides, if `upsmon` were rolled into `upsd`, `upsd` would get even bigger than it is now. You'd have one less process, but the RAM consumption would be pretty close to what you have now — but consumed on each system involved.

See the "Data Room" section in docs/config-notes.txt for more configuration ideas and explanations.

Answer 2

If this really bothers you, roll up your sleeves and use the sockdebug(8)] code to write a "upsmon" type program that sits on top of the state sockets. It won't work over the network, but it means you don't need `upsd`. It also means only one host can monitor the UPS.

This is also a good option to consider if you can't use networked monitoring code for security or safety reasons.

Answer 3

There are plans in the queue... for a long time... to extend `upsd` data server and the NUT clients with ability to use a local Unix socket for the NUT Networked protocol. This would also allow to forgo the dependency on TCP/IP stack for communications within one machine.

See the TODO file for more on this and other related topics.

10 Why isn't upssched part of upsmon?

Most users will never have any reason to use `upssched`. It's complicated, and getting it right for your situation can be tricky. Having it live in a separate program saves resources and lets most people avoid it completely.

It is also coherent with the answer to the previous question.

11 Why doesn't upsmon send a SIGPWR signal to init so it can deal with power events?

Answer 1

New versions of the `init` man page taken from the **sysvinit** package are saying that usage of `SIGPWR` is discouraged, since `/dev/initctl` control channel is the preferred way of communication.

Answer 2

The name of the game is portability. Not everyone's `init` handles that kind of signalling gracefully. What's more, some admins might want to do things differently even if they have that kind of `init` running.

So, to be compatible, upsmon just invokes a shell command. If you want to use init's `SIGPWR` stuff, just put the right "kill" line in a shell script and make your `upsmon` call it. Everyone wins.

12 Why won't bestups talk to my Best Fortress UPS?

There are at least two different protocols being used for hardware with very similar names. The bestups driver tends to support the units built around the newer "PhoenixTec" protocol, and the bestfortress driver supports the older Best hardware.

There is a similar problem with the tripplite_usb driver: it only supports the older, proprietary protocol. Newer standards-compliant Tripp Lite UPS models are supported by usbhid-ups. We name drivers based on the information available at the time the driver was first written, which often is incomplete.

13 What's this about *data stale*?

It means your UPS driver hasn't updated things in a little while. upsd refuses to serve up data that isn't fresh, so you get the errors about staleness.

If this happens to you, make sure your driver is still running. Also look at the syslog. Sometimes the driver loses the connection to the UPS, and that will also make the data go stale.

This might also happen on certain virtualization platforms. If you cannot reproduce the problem on a physical machine, please report the bug to the virtualization software vendor.

If this happens a lot, you might consider cranking up DEADTIME in the upsmon.conf to suppress some of the warnings for shorter intervals. Use caution when adjusting this number, since it directly affects how long you run on battery without knowing what's going on with the UPS.

Note: some drivers occasionally need more time to update than the default value of MAXAGE (in `upsd.conf`) allows. As a result, they are temporarily marked stale even though everything is fine. This can happen with MGE Ellipse equipment — see the mge-shut(8) or usbhid-ups(8) man pages. In such cases, you can raise the value of MAXAGE to avoid these warnings; try a value like 25 or 30.

14 Why do the client programs say *Driver not connected* when I try to run them?

Answer 1

This means that `upsd` can't connect to the driver for some reason. Your `ups.conf` entry might be wrong, or the driver might not be running. Maybe your state path is not configured properly.

Check your syslog. upsd will complain regularly if it can't connect to a driver, and it should say why it can't connect.

Note: if you jumped in with both feet and didn't follow the INSTALL.nut document, you probably started `upsd` by itself. You have to run `upsdrvctl start` (explicitly — on legacy systems only) to start the drivers after configuring `ups.conf`.

On operating systems with a supported service management framework, you might wrap your NUT drivers into individual services instances with `upsdrvsvcctl resync` and then manage those with commands like `upsdrvsvcctl stop` and `upsdrvsvcctl start` (note that on other systems this tool may be not pre-installed via packaging).

In fact, service instances prepared by the `nut-driver-enumerator` script and service based on contents of your `ups.conf` file and automatically maintained by the respective framework can conflict with manual execution of drivers, so `upsdrvctl` would emit a warning in NUT builds with that capability (can be silenced by exporting a `NUT_QUIET_INIT_NDE_WARNING` environment variable with any value).

Answer 2

Some USB UPS devices have unreliable USB to serial interfaces. In that case, it's advised to unplug / plug the device and try again.

If that resolves the issue, you should consider resetting the USB hub the device is attached to before starting the nut driver, using `usb_resetter` script (for Linux) from https://github.com/netinvent/usb_resetter

See files under `scripts/usb_resetter/` in NUT sources for more information.

15 Why don't the pathnames in your documentation match the package I installed?

Each distribution has conventions for where specific file types should be stored. The NUT project cannot possibly track all of these conventions, so the documentation assumes the default installation directory prefix of `/usr/local/ups` when describing file locations. It also allows custom builds of NUT to minimally conflict with files of a packaged installation.

The distributions tend not to change the base name of the files, so you can search for drivers and configuration files in the package database of installed files. For instance, on Debian or Ubuntu derivatives, you can use `dpkg --search usbhid-ups` to see where the drivers are stored.

16 Everything works perfectly during the shutdown, and the UPS comes back on, but my system stays off. What's happening?

Assuming you don't have the problem in the next question, then you probably have an ATX motherboard, have APM or ACPI enabled in your kernel (assuming Linux here), and are reaching the *halt* at the bottom of your shutdown scripts.

Your machine obeys and shuts down, and stays down, since it remembers the *last state* when the UPS restarts.

One solution is to change your shutdown scripts so you never reach that point. You **want** the system to die without reaching the part where the kernel tells it to shut down. A possible script might look like this (see `scripts/systemd/nutshutdown` in NUT sources for a more streamlined and modern variant):

```
        # other shutdown stuff here (mount -o remount,ro ...)
        # `upsmon -K` if available on still mounted filesystems
        # at this point is more portable than the `test` below

        if (test -f /etc/killpower || /sbin/upsmon -K)
        then
                /sbin/upsdrvctl shutdown

                sleep 600       # this should never return

                # uh oh, we never got shut down! (power race?)
                reboot
        fi

        halt -p
```

For more details and a better integrated example, please see https://github.com/networkupstools/nut/blob/master/scripts/systemd/-nutshutdown.in (a copy tailored for your OS distribution may be or not be included in NUT packages, if you installed from those).

The other solution is to change your BIOS setting to "always power on" instead of "last state", assuming that's possible.

17 My operating system has no shutdown scripts, how can I tell the UPS to go down and/or avoid power races?

Modern operating systems offer service management frameworks, like the Solaris 10/11 and illumos SMF, or Linux systemd. These frameworks drive the life cycle of the operating system and enforce their opinions on the shutdown end-game in particular. For example, any "user-land" processes which remain alive after a certain (configurable or hard-coded) timeout would be killed off to not block the shutdown or reboot request.

While systemd has a concept of shutdown hooks, so the logic above can be placed as a script into a special directory `/usr/lib/syster` and called after all the service daemons have been killed off, the SMF does not commonly offer such a facility.

Instead, the question is turned around: "What would break in your system if it is suddenly turned off (not only due to power, but also kernel panic, hardware failure, etc.?)" and the suggested solutions follow up from there.

The copy-on-write filesystems, like ZFS extensively used in Solaris/illumos as well as other operating systems (including some *BSD and Linux distributions), do not care about sudden reboots (assuming the storage hardware does not lie about honouring orderly metadata flushes). At worst, an incomplete transaction would be lost, but the filesystem structure itself remains valid and does not need any lengthy `fsck` after a reboot.

It may be or not be similar with databases (depending on how yours would log incoming write transactions), mail systems, etc.

Chances are, with a complete stack of well-engineered software for which hardware failure is always an option (considered in design), you do not even need to shut down anything in case of reported power failure and the UPS going on battery.

In practice, if you are not after maximizing the service uptime but rather want some peace of mind that your application data is not corrupted, you can script your `NOTIFYCMD` implementation to stop just those services (containers, VMs...), maybe `umount` non-transactional file systems (FAT EFI partition, USB storage), and revive them when/if the UPS becomes "on-line" again and this box is still alive. Or it just boots up if the power did get lost (whether you requested the UPS to power-off or power-cycle or not, whether it honoured such request or not). Probably `upssched` as the notification handler can help implement this logic consistently, so it would react to different events. Using service grouping or milestones as a way to consistently start/stop "fragile" services should also help (in this case the framework ensures the action only happens once, regardless of how many times your scripts requested it).

Notably, with this approach you DO NOT tell the OS to actually shut down, and so suffer any forced kill of user-land processes after any timeouts. You choose which services might be hurt by the outage and should go down, and which can run as long as they can (or in which mode, e.g. turning your mail, DNS or LDAP server read-only just in case until the storm passes). The still-running stub of your OS is the environment which talks to the UPS and takes care of eventually restarting the services or the whole box, as you deem fit.

18 My operating system uses (immutable) images, how can I tell the UPS to go down and/or avoid power races?

One approach was proposed in discussion with Lennart Poettering of systemd fame, that operating system concepts constructed with (immutable) run-time images are actually layered like onions, with code in another file system (usually an initrd) picking the run-time image to use and `chroot`'ing into it... and possibly regaining control when the run-time operating environment exits. Sort of like containers, where the operating system which you interact with is the inside of the container, and you do not directly see its hypervisor.

The problem is that while NUT services can run as part of the run-time image and decide to power-off or power-cycle the UPS in case of trouble, there is no space for classic NUT approach to do this in the end-game of that run-time image (calling `upsmon -K` to detect the `POWERDOWNFLAG` file, and launching some new instance of the driver to talk to the UPS itself) "after the filesystem was remounted read-only", because that life-cycle concept has no such spot in the state machine of the run-time image — its storage is to be completely unmounted.

Instead, the proposed idea was to have a build of NUT included also into that "managing" initrd image, with perhaps a `/run` (or similar) location passed (bind-mounted?) from the initial environment into the launched run-time system. Into such a location both the `POWERDOWNFLAG` file can be written and, as part of `NOTIFYCMD` handling, the most-recent copies of NUT configuration files which may be managed in the interactively accessible operating environment.

When the run-time image exits and the logic in initrd image which launched it regains control, it can check for existence of the `POWERDOWNFLAG` file and call **its own** copies of NUT drivers to talk to the UPS, and/or implement the long sleep and reboot to avoid a power race condition, if needed.

See also: https://github.com/networkupstools/nut/issues/2836

19 My system has an ATX power supply. It will power off just fine, but it doesn't turn back on. What can I do to fix this?

This depends on how clueful your motherboard manufacturer is, and isn't a matter of the OS. You have to do one of the following things depending on what's supported:

- Set a jumper on the motherboard that means "return after outage"
- Set something in the BIOS that says "power up after power failure"
- Try using something (like a capacitor) across the power button to "push" it for you — this might not work if it needs a delay
- Hack the cable between the power supply and the motherboard to fool it into powering up whenever line power is present
- Teach a monkey to watch the machine and press the power button when the outage is over. This might work, but it creates high produce bills.

If you can't use one of the first two options, give the board to an enemy. Let them worry about it.

20 My Mac won't power back up by itself into Linux after the UPS shuts down. What can I do about this?

This is about the same situation as the ATX question above, only worse. Earlier Macs apparently supported a hack where you could cat some magic characters at `/dev/adb` to enable "server mode". This would instruct the system to reboot while unattended.

From Usenet post <6boftzxz51.fsf@ecc-office.sp.cs.cmu.edu>:

```
        # Send packet over the ADB bus to the PowerMac CUDA chip
        # telling it to reboot automatically when power is restored
        # after a power failure.

        cat /etc/local/autoboot.adb > /dev/adb

    # autoboot.adb contains these three bytes (in hex):  01 13 01
```

Later PowerPC Macs with a PMU and the appropriate kernel driver can achieve the same effect with the following command:

```
echo server_mode=1 > /proc/pmu/options
```

The following pages have some slightly more kludgy answers which involve the use of `setpci`, and are highly model-specific:

- https://www.mythic-beasts.com/support/servers/colo/macminicolo_howto
- http://superuser.com/questions/212434/reboot-after-power-failure-for-mac-running-ubuntu
- http://ubuntuforums.org/showthread.php?t=1209576

Note: this question has been in the FAQ for several years now, and there's still no clean answer. Let me guess: everyone who runs a server on Mac hardware has a team of trained monkeys, and feeds them by growing bananas in the tropical environment formed by waste heat from the equipment.

The rest of us are still waiting for the answer. Booting into the Mac OS to frob the "file server" panel is not an acceptable solution.

21 My Mac won't power back up by itself into Mac OS X after the UPS shuts down. What can I do about this?

This is relatively simple to fix. If you have console or VNC access, log in as an administrator, go to System Preferences, click on Energy Saver, click on the options tab, check "Restart automatically after a power failure".

Alternatively, you can connect via SSH and run `sudo pmset autorestart 1` to achieve the same effect.

22 I want to keep the drivers and upsd in their own security domains. How can this be accomplished?

Using a few role accounts and a common group, you can limit access to resources such as the serial port(s) leading to the UPS hardware.

This is just an example. Change the values to suit your systems.

- Create a user called *nutdev* and another called *nutsrv*. Put them both in a group called *nut*.
- Change the owner of any serial ports that will be used to nutdev, and set the mode to 0600. Then change the ownership of your state directory (usually /var/state/ups) to nutdev.nut.

For my development system this yields the following /dev entries:

```
0 crw-------   1 nutdev   tty        4,  64 Sep  3 17:11 /dev/ttyS0
0 crw-------   1 nutdev   tty        4,  65 Sep  3 17:11 /dev/ttyS1
```

- Switch to root, then start the drivers:

```
# upsdrvctl -u nutdev start
```

- The listing for /var/state/ups then looks like this:

```
4 drwxrwx---   2 nutdev   nut          4096 Aug 20 18:37 .
4 drwxr-xr-x   4 root     root         4096 May 14 21:20 ..
4 srw-rw----   1 nutdev   nut             0 Sep  3 17:10 apcsmart-ups1
4 srw-rw----   1 nutdev   nut             0 Sep  3 17:10 blazer_ser-ups2
```

You may have to remove old socket or state files first if you are changing to this security scheme from an older version. The drivers will create new files with the right owners and modes.

Note that `/var/state/ups` is group writable since `upsd` will place the `upsd.pid` file here by default.

You may have to change the groups of `upsd.conf` and `upsd.users` to make them readable to the NUT `upsd` program. These files should not be owned nor writable by `nutsrv`, since someone could compromise the daemon and change the config files. Instead, put `nutsrv` in a group (`nut` in this example), then make the files owned by `root.nut`, with POSIX bits mode `0640`.

Once the config files are ready, start upsd:

```
:; upsd -u nutsrv
```

Check your syslog to be sure everything's happy, then be sure to update your startup scripts so it uses this procedure on your next boot.

If you like this, you'll probably also find the chroot(2) process to be useful and interesting. See security.txt for more details.

23 What's the point of that *security domains* concept above?

The point is limiting your losses. If someone should happen to break into `upsd` in that environment, they should only gain access to that one user account. Direct access to the serial device is not possible, since that is owned by another user.

There is also the possibility of running the drivers and `upsd` in a chroot jail. See the `chroot` option in security.txt, `upsd` and driver documentation.

Why give would-be vandals any sort of help?

Put it this way — I **wrote** good chunks of this stuff, and I still run the programs this way locally. You should definitely consider using this technique.

24 How can I make upsmon shut down my system after some fixed interval?

You probably don't want to do this, since it doesn't maximize your runtime on battery. Assuming you have a good reason for it (see the next entry), then look at scheduling.txt or the upssched(8) man page for some ideas.

25 Why doesn't upsmon shut down my system? I pulled the plug and nothing happened.

Wait. The `upsmon` client doesn't consider a UPS to be critical until it's **both** *on battery* and *low battery* at the same time. This is by design. Nearly every UPS supports the notion of detecting the low battery all by itself. When the voltage drops below a certain point, it *will* let you know about it.

If your system has a really complicated or time-consuming shutdown procedure, you might need to shut down before the UPS raises the low battery flag. For most users, however, the default behavior is adequate.

Ask yourself this: why buy a nice big UPS with the matching battery and corresponding runtime and then shutdown early? If anything, I'd rather have a few more minutes running on battery during which the power might return. Once the power's back, it's business as usual with no visible interruption in service.

If you purposely shut down early, you guarantee an interruption in service by bringing down the box.

See upssched.txt for information on how you can shutdown early if this is what you really want to do.

26 The CGI programs report *access to that host is not authorized* — what's going on?

Those programs need to see a host in your hosts.conf before they will attempt communications. This keeps people from feeding it random `host=` settings, which would annoy others with outgoing connection attempts from your system.

If your hosts.conf(5) turns out to be configured correctly with proper `MONITOR` entries and all that, check the permissions. Your web server may be running the CGI programs as a user that can't read the file.

If you run your web server in a chroot jail, make sure the programs can still read `hosts.conf`. You may have to copy it into the jail for this to work. If you do that, make sure it's not writable by any of the user accounts which run inside the jail.

27 upsd is running, so why can't I connect to it?

Assuming you haven't changed the TCP port number on the command line or at compile-time, then you may have some sort of firewall blocking the connection.

`upsd` listens on TCP port *3493* by default. If you do not specify a `LISTEN` directive in `upsd.conf`, `upsd` only listens on the loopback interface. See the upsd.conf(5) man page for details.

28 How do you make upsmon reload the config file?

Or a variation like…

29 How do you make upsd reload the config file?

Either find the pid of the background process and send it a SIGHUP, or just start it again with *-c reload* (requires that the previously started daemon saves a PID file).

If you send the signals yourself instead of using -c, be sure you hit the right process. There are usually two `upsmon` processes, and you should only send signals to one of them. To be safe, read the PID file.

If your daemons are managed as systemd units, it is more idiomatic to use the framework commands, e.g. `systemctl reload nut-server` (upsd) or `systemctl reload nut-monitor` (upsmon). Note that the implementation of `nut-server.servic` by default starts `upsd -F` and does not save a PID file; if your workflow requires to use plain `upsd -c reload`, you should customize the unit (with a drop-in file) to start `upsd -FF`.

NUT releases after 2.8.0 define aliases for these units, so if your Linux distribution uses NUT-provided unit definitions, `systemctl reload upsd` may also work.

30 I just bought a new WhizBang UPS that has a USB connector. How do I monitor it?

There are several driver to support USB models.

- usbhid-ups(8) supports various manufacturers complying to the HID Power Device Class (PDC) standard,
- tripplite_usb(8) supports various older Tripp-Lite units (with USB ProductID 0001)
- bcmxcp_usb(8) supports various Powerware units,
- blazer_usb(8) supports various manufacturers that use the Megatec / Q1 protocol.
- nutdrv_qx(8) supports various manufacturers that use the wider Megatec / Q* protocol family. This is the driver slated to receive all further development in this area, it was specially designed to support many more sub-drivers and has added a lot over time, so please do try it first nowadays.
- apc_modbus(8) supports some APC branded devices built after 2010 (requires to be built against a libmodbus version with RTU USB support)

Refer to the respective *driver-name* (8) man page for more information.

You can also consult the Hardware Compatibility List (HCL) and filter on USB: https://www.networkupstools.org/stable-hcl.html?connec

31 My USB UPS has a bogus Vendor ID 0x0001 and Product ID 0x0000, what driver supports it?

Unfortunately, many devices are made without registering as a Vendor with the corresponding standards body, and use generic USB chips for interfacing with a computer (roughly similar to using a network interface card with a random MAC address and PCI ID, and thus poorly identifiable device specifics needed to automatically load some certain driver). Often they also lack a unique serial number field, so monitoring several devices is problematic.

One frequent case is with devices identifying as "Fry's Electronics" and/or "MEC0003", if those metadata are served at all, or plain "0001/0000" in ID field. In some cases they are accompanied by "UPSmart" software with a "MEGA(USB)" connection option that works for Windows users.

Your best bet is to search for community discussions of issues on NUT GitHub at https://github.com/networkupstools/nut/-issues?q=is%3Aissue and try options there. Devices with these chips were known to connect with drivers for such unrelated protocols as Megatec Q* (different sub-drivers, often `fabula` or `hunnox`), ATCL, or USB-HID.

32 My USB UPS is supported but doesn't work!

On Linux, udev rules are provided to set the correct permissions on device file. This allows the NUT driver to communicate with the UPS, through this device file.

However, the driver may still fail to start and support the device, with a message like:

```
failed to claim USB device: could not claim interface 0: Operation not permitted
```

Operation not permitted is a message pointing to a privilege issue. The most frequent issue is that udev has not actually applied the rule:

- if NUT has been freshly installed,
- and if the device USB cord was already plugged when installing NUT.

In this case, just unplug and plug back the USB cord, then restart NUT. Instead of unplugging, you might also be able to run `udevadm trigger --subsystem-match=usb --action=change` to fix permissions.

There was a mistake in the naming of the NUT udev rules file which resulted in the rules being overridden by another udev configuration file. While this has been fixed in the Git master branch, your distribution may still be affected. Details are available in the following GitHub issue: https://github.com/networkupstools/nut/issues/140

There may also be a conflict with an already running instance of the driver, e.g. when a systemd unit instance `nut-driver@yourdevi` was automatically created and started by the `nut-driver-enumerator`, and then you try to follow older revisions of the NUT documentation or blogs, and start another copy with `upsdrvctl` (which should only be used on legacy systems nowadays).

33 Why do you not use the Linux kernel HID driver when communicating with USB UPSes?

When the `usbhid-ups` was first written, it replaced an older driver `hidups` which used the Linux kernel USB HID API. At the time, the kernel HID API could not distinguish between identical Usage IDs that were nested in different parent IDs, so many common measurements were not available from `hidups`. For this reason, the libusb approach was chosen, which has the added side effect of being more portable than the Linux HID API. The Linux hiddev device nodes have very similar permissions problems as the `/dev/bus/usb` nodes that the libusb approach uses.

Due to difficulties in running libusb on OS X and Windows, those platforms might actually benefit more from a native HID approach.

34 I get a message from the kernel that the driver "did not claim interface 0 before use"

On Linux, if two copies of a driver are competing for the UPS, these messages will appear in dmesg:

```
usbfs: process 29641 (usbhid-ups) did not claim interface 0 before use
```

This can be a symptom of a source install conflicting with a package install. There is a rudimentary locking mechanism in NUT, but there is a chance that the packages might not use the same directory as the NUT default, and the conflict will be reported by the kernel.

Also see above about conflicts between driver instances started by a service management framework like systemd and started manually (e.g. with `upsdrvctl`).

35 Why does my (Eaton 5E) USB UPS on Linux connect but quickly disconnects soon?

This issue was extensively investigated by NUT community members in https://github.com/networkupstools/nut/issues/630 and resulted in a chain of distribution bugs logged such as https://bugzilla.redhat.com/show_bug.cgi?id=1715504

The gist of it is that some versions of Linux kernel used an "USB HID quirk" for certain devices, see Linux kernel source `drivers/hid/hid-quirks.c` file, including MGE/Eaton vendor ID (0x0463) based on an older device a contributor had issues with. Firmware in newer devices no longer had the bug which needed the "quirk" and misbehaved when it was enabled. While newer (and much older) Linux kernels should not have the problem, with the quirk removed according to https://lkml.org/lkml/2018/11/26/580 having the issue in the field really depends on the combination of Linux kernel and device firmware that meet.

Either way, it seems that problematic combinations preclude Linux from seeing the device as a `hid-generic` first, to hand it over to a NUT driver.

This quirk can be tuned with a kernel boot parameter (via GRUB etc.):

```
usbhid.quirks=0x0463:0xffff:0x08
```

to re-enable the NOGET quirk.

For context, according to https://bugzilla.redhat.com/show_bug.cgi?id=1875532 the symptoms for the problem look like this:

```
# Plug in the UPS and observe the dmesg logs,
# the following continuously appears:
[   93.568082] usb 1-6: new full-speed USB device number 9 using xhci_hcd
[   94.311469] usb 1-6: New USB device found, idVendor=0463, idProduct=ffff,  ↩
   bcdDevice= 0.01
[   94.311475] usb 1-6: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[   94.311483] usb 1-6: Product: 5E
[   94.311486] usb 1-6: Manufacturer: EATON
[   96.269989] hid-generic 0003:0463:FFFF.000A: hiddev96,hidraw2: USB HID v1.10  ↩
   Device [EATON 5E] on usb-0000:00:14.0-6/input0
[  107.369425] hid-generic 0003:0463:FFFF.000A: usb_submit_urb(ctrl) failed: -1
[  107.369469] hid-generic 0003:0463:FFFF.000A: timeout initializing reports
[  112.828826] usb 1-6: USB disconnect, device number 9
[  113.284452] usb 1-6: new full-speed USB device number 10 using xhci_hcd
[  114.027693] usb 1-6: New USB device found, idVendor=0463, idProduct=ffff,  ↩
   bcdDevice= 0.01
[  114.027698] usb 1-6: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[  114.027701] usb 1-6: Product: 5E
[  114.027704] usb 1-6: Manufacturer: EATON
[  115.984222] hid-generic 0003:0463:FFFF.000B: hiddev96,hidraw2: USB HID v1.10  ↩
   Device [EATON 5E] on usb-0000:00:14.0-6/input0
[  126.825756] hid-generic 0003:0463:FFFF.000B: usb_submit_urb(ctrl) failed: -1
[  126.825775] hid-generic 0003:0463:FFFF.000B: timeout initializing reports
[  132.527809] usb 1-6: USB disconnect, device number 10
```

A similar report on the driver side may look like:

```
usbhid-ups[4554]: libusb_get_report: Input/output error
upsd[4591]: Data for UPS [eaton] is stale - check driver
usbhid-ups[4554]: Can't claim USB device [0463:ffff]: No such file or directory
upsd[4591]: Can't connect to UPS [eaton] (usbhid-ups-eaton): No such file or  ↩
   directory
upsmon[5148]: Poll UPS [eaton@localhost] failed - Driver not connected
upsmon[5148]: Communications with UPS eaton@localhost lost
```

Other similar looking issues may include improper setup of udev, upower and similar frameworks to hand over the device from the OS to a driver daemon; competition with other software probing USB devices (ModemManager was mentioned in this context), including running several copies of the NUT drivers trying to use same port (e.g. one started by services and another manually as you tried to debug the problems).

Software quirks aside, please do test with a different USB cable and/or port on the computer. These were known to cause grief beyond what can be fixed with a few key words ;)

Finally, sometimes the issue is on the OS side (and/or USB chipset), to the point that the USB driver can not be unloaded and re-attached until you power cycle the system.

36 Why doesn't my package work?

Or a variation like...

37 I can't run this because there's no package for it. Why isn't this in a package yet?

Sorry, can't help you there. All official releases are source code and are posted on https://www.networkupstools.org/ along with PGP signatures for verification.

This means all packages have been built by a third party. If you have an issue that's related to packaging, you will need to seek help with whoever built it for you.

38 My UPS is directly connected to an appliance with a limited version of NUT, how can I monitor the UPS from arbitrary clients?

You can set up a separate general-purpose system as the NUT server for your "arbitrary clients", using `dummy-ups` in "relay mode" as the driver. This instance of `dummy-ups` would in turn be the NUT client allowed to interact with the appliance and that way with the UPS connected there.

Note
The original question related to a NAS with NUT provided in its firmware OS, that only allowed one or few clients and not a whole rack's worth of client IP addresses.

39 My networked UPS can't handle being monitored by dozens of NUT clients

Network management cards on many UPSes are rather puny appliances, often known to either limit the amount of clients who may connect, for security or performance reasons, or otherwise to crash or respond very slowly when overwhelmed.

You may be better off reducing the amount of servers connected to the UPS with the `snmp-ups`, `netxml-ups` or similar type of driver, and set up other systems as clients of these NUT servers.

Developers who are working on NUT, its drivers, or further projects and appliances based on NUT, and who need to monitor their UPS from multiple systems using the complete NUT stack on each system (e.g. during testing), can benefit from dedicating a separate general-purpose system as the NUT server using the real (networked) driver for the UPS, while using the `dummy-ups` in "relay mode" as the driver connected to this dedicated server on each tested system.

40 How can I setup NUT as a proxy (setup a server to forward/relay client data)?

This can easily be achieved by using the `dummy-ups` driver. The `port` field acts as the reference to the "other" UPS served by another NUT server.

Example with `dummy-ups` driver:

```
[proxy]
        driver = dummy-ups
        port = upsname@ip-or-hostname[:port]
        desc = "UPS proxy for UPS upsname on server ip-or-hostname"
```

Also note that there is a `clone` driver with similar purpose, which allows users to group clients to a particular outlet of a device with a "real" driver running locally, and deal with this output as if it was a normal UPS.

Here the `port` field references the driver socket name that the "real" UPS driver is using. See its manual page for more details and caveats.

Example with `clone` driver:

```
[realups]
        driver = usbhid-ups
        port = auto

[clone-outlet-1]
        driver = clone
        port = usbhid-ups-realups
        load.on = outlet.1.load.on
        load.off = outlet.1.load.off
        load.status = outlet.1.status
        [...]
```

This allows to group load attached to a separately manageable outlet (or group of outlets) on larger UPS and ePDUs, in order to power those devices on/off together. This may be also useful to delegate management of feeds to devices for purposes like hosting or supporting hardware for smaller teams sharing a rack in a larger company.

41 Why are there two copies of upsmon running?

It's not really two complete copies if your OS forks efficiently.

By default, `upsmon` runs most of the grunt work as an unprivileged user and keeps a stub process around with *root* powers that can only shut down the system when necessary. This should make it much harder to gain *root* in the event a hole is ever discovered in `upsmon`.

If this really bothers you and you like running lots of code as `root`, start `upsmon` with `-p` option, and it will go back to being one big process. This is not recommended, so don't blame us if something bad happens in this mode.

42 I get the following error while building: `make[4]: don't know how to make HP-UX/nut-drvctl.sh. Stop`

Answer 1

Current NUT codebase (since v2.8.0) is regularly tested with GNU, BSD and Sun implementations of `make`, so seeing failures in release snapshots (or iterations that made it to the master branch) is very surprising. Please raise an issue on GitHub.

Answer 2

Older NUT codebase (release tarballs) has some hidden dependencies on GNU Make which show up while running `make distcheck`. If you are running `make distcheck` or its variants, you will need to install GNU Make (`devel/gmake` in the ports tree), which is incidentally what the official FreeBSD port of NUT does for all builds.

43 I have *some problem* with *some old version* . . .

Get the latest stable release, and see if it still happens. If it goes away, it means someone else reported it and got it fixed a long time ago.

You may want to search the mailing lists to see if someone else has experienced the same problem. If so, there is a good chance that someone else has worked through the process necessary to shoehorn the latest NUT version into your distribution (potentially with unofficial packages).

Some OS distributions contain old versions of NUT. If your hardware is newer than the NUT release, there is a good chance that support has not been added yet. Please do not tell us you have the "latest version for Distro XYZ" — even if the developers are familiar with that distribution, it helps others if you quote the exact package version.

Note
Check the release date on the version you have. If it's more than about 6-12 months old, there's probably a newer stable tree version out there. As development happens actively, be sure to also check if a custom build from Git (usually using the `master` branch of NUT https://github.com/networkupstools/nut/ repository) has your issue fixed by some kind soul already.

---

44 I can't connect with SSL using old NUT on an appliance

Unfortunately, some vendors do not issue new firmwares often, and even more rarely do they significantly update any programs inside. It is not uncommon to see NUT versions over a decade old delivered with small NAS boxes, for example.

This may impact not only NUT protocol compatibility, but also transport protocols such as SSL — as cipher algorithms get outdated over time, and ones deemed insecure are no longer handled at all (by default). This is not a problem limited to NUT: old SSH Key Exchange (kex) protocols or old HTTPS mechanisms also become hard or impossible to use eventually.

On one hand, you can look into NUT configuration of `DISABLE_WEAK_SSL`.

On another, you can modify configuration of the newer system to allow older algorithms as required by the older other system.

For NUT built against OpenSSL, the change would be in `/etc/ssl/openssl.cnf` and similar to the diff block below:

```
--- a/etc/ssl/openssl.cnf
+++ b/etc/ssl/openssl.cnf
@@ -52,13 +52,6 @@ tsa_policy3 = 1.2.3.4.5.7

 [openssl_init]
 providers = provider_sect
+ssl_conf = ssl_sect
+
+[ssl_sect]
+system_default = system_default_sect
+
+[system_default_sect]
+CipherString = DEFAULT@SECLEVEL=0

 # List of providers to load
 [provider_sect]
```

Of course, keep in mind that by doing this you degrade your security level. If better solutions are possible in your situation, prefer to follow them!

Thanks to Kajetan Rzepecki for doing the research and posting the findings in https://github.com/networkupstools/nut/issues/-1899

45 I built NUT from Git, and it complains about lots of missing files. What happened?

If you are not actively developing a driver, can you use a snapshot instead? The NUT instance of Buildbot generates tar files of the latest NUT source after each successful build, and these snapshots include a pre-built version of the `./configure` script.

Warning
There is an outstanding bug that after the shutdown of BuildBot, no regular distribution tarballs are in fact published.

To build from Git, you will need recent versions of autoconf, automake, libtool, asciidoc, a2x and its dependencies for DocBook/dblatex. Rather than publish a list of the exact versions needed (which will quickly become out of date), we recommend you consult your distribution's dependency list for building a NUT package, and use that as a starting point.

That said, for the numerous systems running build agents of the NUT CI farm, their lists of dependency packages are available in `docs/config-prereqs.txt` in NUT sources.

46 Do I have to use a serial connection to monitor the UPS? What about direct network connections (SNMP or otherwise)?

NUT currently supports USB communication through several drivers, and also SNMP and XML/HTTP (Eaton and MGE) communications.

Since NUT is very extensible, support for a new communication bus can be added easily.

Any time there is a gap in features, it's usually because the group of people who own that hardware and the group of people who write code don't overlap. The fix is to make them overlap — turn an owner into a developer or vice-versa.

47 What happened to the patch I sent?

We try to prioritize emails with patches, but you should understand that a simple fix for your bug might be complicated to integrate with the rest of NUT. Changing the way a fundamental component works, such as USB support, means a lot of testing to ensure that your fix does not break other drivers.

Sometimes patches are put on hold due to a feature freeze. If it doesn't show up once the new version opens up, please send it again.

It may also be much more productive to submit changes as pull requests via https://github.com/networkupstools/nut/pulls so they are automatically processed by the NUT CI farm across numerous target platforms, and various inconsistencies can be diagnosed and fixed early.

48 I'm not much of a programmer. How can I help?

There's always work to be done outside of the realm of code bashing. Documentation can always be improved. A new user's perspective is sometimes needed to appreciate this, since developers and long-time users consider everything obvious. Bug reports and pull requests on any project's documentation are just as valuable as those for the actual programs' sources.

Fielding questions on the mailing lists is also helpful. This lets other people to focus on coding issues while allowing the original poster to get some information at the same time. It's quite a relief to open that mailbox and find that someone else has already handled it successfully.

49 I replaced the battery in my APC Smart-UPS and now it thinks the battery is low all the time. How do you fix this?

Or a variation like…

50 My APC UPS keeps reporting *OL LB*, even after it's been charging for many hours. What can I do about this?

This happened to me, and some other people too. The combination of our experiences should prove useful to you.

First, you need to realize that the UPS apparently stores data about the battery, load, and runtime. After replacing the battery, it needs to be clued in to the new situation. If the traditional runtime calibration doesn't work, you have to try something a little more drastic.

You need to **completely** drain the UPS while it has a good ground. This means you can't just pull the plug. You also have to disconnect it from the computer so this software won't shut it down.

The easiest way to do this is to first unplug your computer(s) from it, and plug in a token load like a lamp. Also, move the UPS to a power strip that doesn't switch the ground line or an outlet that you can switch off at your panel.

Once the UPS is up at 100% charge (this is important), disconnect the power. It *must* remain connected to the ground, or the results may not be accurate. Ignore the sounds it makes, and go away until it's done. Don't do anything to the front panel while this is happening.

After all of this, put things back the way they should be and let it charge up. You should find that it again gives reasonable values and behavior, as it was when it was new.

Thanks to Matthew Dharm for helping us nail down this procedure.

51 upsstats returns temperatures in Celsius. I like Fahrenheit. Where's the config file to switch it back?

Temperature scales are handled by the template files, so edit your upsstats.html and change it from TEMPC to TEMPF.

52 Why is the mailing list ignoring me?

You probably asked a question that's answered in this FAQ, or somewhere else in the documentation, and nobody wants to quote it for you.

There is a small chance that the mailing list spam filter ate your message. Check the list archives to see if your message appears there. Also double-check that you have subscribed to the lists and completed all the confirmation rituals of its engine.

Convincing the other subscribers that you've actually read down this far might be useful. You might mention "queequeg" for better results.

This URL may also be helpful:

http://www.catb.org/~esr/faqs/smart-questions.html

53 Why are you so insistent about sending emails to public mailing lists instead of to individuals?

By and large, NUT is a volunteer effort. By emailing one person, you are asking them to take care of your question. If you email the list instead, you give others the opportunity to answer.

In addition, the mailing lists are publicly archived, and therefore easily searchable. Chances are, you aren't the only person who will ever have that question.

There are similar benefits to using the discussions on issue tracker at https://github.com/networkupstools/nut/issues and if suitable, in the currently open pull requests.

54 If you want mailing list replies to go to the list, why don't you add a Reply-To: header?

We are not going to rehash all of the arguments for and against this in a simple FAQ entry. If you intend for your reply to go to more than just the last person who posted, it is not too much trouble to hit "Reply All".

55 I found some information about another kind of UPS protocol you don't support yet, but I don't know what to do with it. Can you help?

If you're not a programmer, you can still help others by making that protocol available. You might host the document somewhere and send the URL to one of the mailing lists.

Posting an issue with attachments on https://github.com/networkupstools/nut/issues can also be helpful.

56 How can you answer questions to situations that nobody's encountered yet? Isn't this a frequently asked questions file?

Answer 1

It's a kind of Magic.

Answer 2

It's both that and a frequently **anticipated** questions file, too.

The idea is to write it up in here so that nobody asks the mailing list when it finally does get released.

57 My UPS powers up immediately after a power failure instead of waiting for the batteries to recharge!

Or a variation like…

58 My UPS (an APC as it happens) lacks the field "battery.charge.restart" — so how will it know when to restart?

You can rig up a little hack to handle this issue in software.

Essentially, you need to test for the POWERDOWNFLAG in your **startup** scripts while the filesystems are still read-only (before `upsmon` daemon has started and removed it). You can also query `upsmon -K` for presence of the file, to avoid hard-coding the path or parsing it from your `upsmon.conf` file. If the flag is there, you know your last shutdown was caused by a power failure and the UPS battery is probably still quite weak.

In this situation, your best bet is to sleep it off. Pausing in your startup script to let the batteries recharge with the filesystems in a safe state is recommended. This way, if the power goes out again, you won't face a situation where there's not enough battery capacity left for upsmon to do its thing.

Exactly how long to wait is a function of your UPS hardware, and will require careful testing.

If this is too evil for you, buy another kind of UPS that will either wait for a minimum amount of charge, a minimum amount of time, or both.

59 I'm facing a power race

Or a variation like…

60 The power came back during the shutdown, but before the UPS power off. Now the UPS does not reboot, and my computer stays off. How can I fix that?

There is a situation where the power may return during the shutdown process. This is known as a race. Here's how we handle it.

"Smart" UPSes typically handle this by using a command that forces the UPS to power the load off and back on. This way, you are assured that the systems will restart even if the power returns at the worst possible moment.

Contact closure units (ala genericups), on the other hand, have the potential for a race when feeding multiple systems. This is due to the design of most contact closure UPSes. Typically, the "kill power" line only functions when running on battery. As a result, if the line power returns during the shutdown process, there is no way to power down the load.

The workaround is to force your systems to reboot after some interval. This way, they won't be stuck in the halted state with the UPS running on line power.

Implement this by modifying your shutdown script like this:

```
        # `upsmon -K` if available on still mounted filesystems
        # at this point is more portable than the `test` below

        if (test -f /etc/killpower || /sbin/upsmon -K)
        then
                /sbin/upsdrvctl shutdown

                sleep 120

                # uh oh, we never got shut down! (power race?)
                reboot
        fi
```