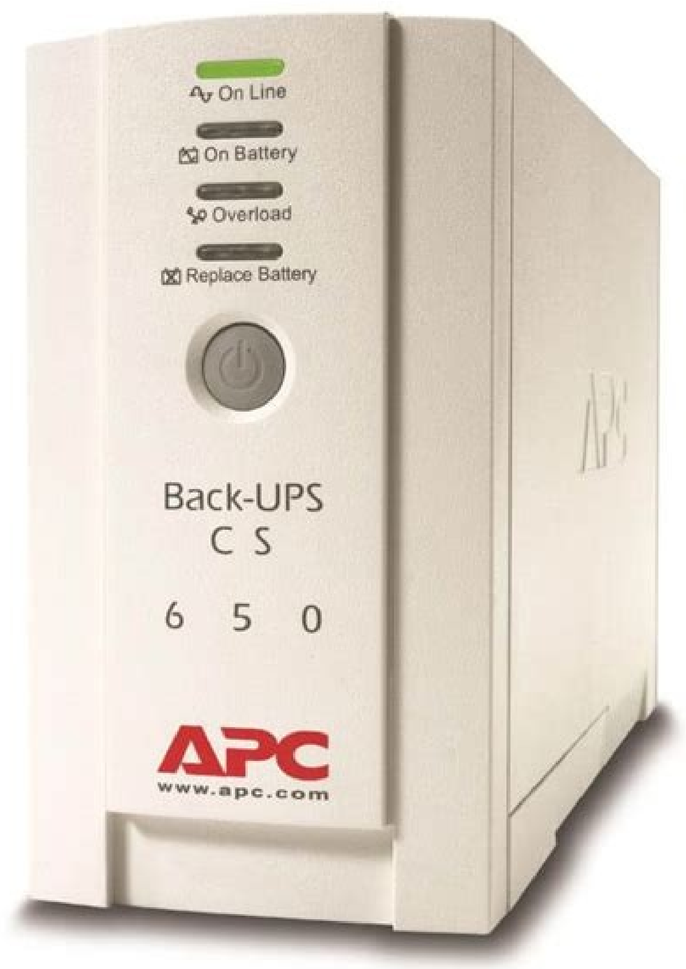
Apc smart ups 650 manual

Continue


Back-UPS
650
APC


Cable In
Cable Out
Data Port
DSL/Modem/
Network/Fax
Wall Outlet
CAUTION - Refer to bottom of unit for safety markings.
Circuit Breaker
Push to Reset
Building
Wiring Fault


On Line
On Battery
Overload
Replace Battery
Back-UPS
C S
650
APC
www.apc.com

Apc cs 650 manual. Apc back-ups cs 650 manual. Back-ups cs 650 manual.

Image not available forColor: To view this video download Flash Player Green PremiumTM label is Schneider Electric's commitment to delivering products with best-in-class environmental performance. Green Premium promises compliance with the latest regulations, transparency on environmental impacts, as well as circular and low-CO2 products.Learn morearrow2_leftThe product must be disposed on European Union markets following specific waste collection and never end up in rubbish bins Easily find answers to the most frequently asked questions. Battery backup units allow you to continue using your network connection or computer system during utility power problems like outages or when utility voltage fluctuates outside of safe levels. In these cases, the battery backup unit will instantly switch ... Show More The best UPS backup will give you the peace of mind that, in the event of a blackout or power surge, your PC and game progress will remain safe. Power surges can ruin gaming PC components, but with a UPS backup you'll save a bunch of troubleshooting, and more importantly: money. Should something catastrophic happen to your power, the best UPS will be your knight in technological armor.If you've got a high-end gaming PC (opens in new tab), it's a smart idea to pair it with an uninterrupted power supply battery as a backup. This protects your system from fluctuations in outlet power and acts as a surge protector, too. A UPS will utilize its internal batteries to deliver a steady flow of power, and a good one will work long enough for you to save your work, or get to a save point, before shutting down your PC safely.The time a UPS gives you to save your game or work will vary, and there are other things to consider, too. Each UPS below we've tested and found to deliver exactly what we needed out of what should be an otherwise fairly unremarkable looking black box. So you can get a better read on which is best for you and your budget.The best UPS for gaming PCsImage 1 of 3(Image credit: Cyberpower)Image 1 of 3(Image credit: Cyberpower)Image 1 of 3(Image credit: Cyberpower)Image 1 of 3The CyberPower CP1500PFCLCD 1500VA is one of the best UPS on the market. Several factors contribute to it being our top pick for most gamers. First, it has enough juice to handle most gaming machines on the market, even if you're running dual GPUs. Unless you've got systems with ten hard drives, quad-way GPUs, and other accessories, the CP1500PFCLCD should have enough juice to last 10–20 minutes (longer if you have a more moderate rig) if a power failure occurs.One of the most critical features of the CP1500PFCLCD is its true sine-wave output. Most UPS backups in their price range only provide sine-wave simulated production, a stepped sine-wave that approximates what you get out of your wall outlet. Some electronics are sensitive to simulated sine waves and will behave abnormally. At $214, having true sine-wave output is unheard of, so kudos to CyberPower for delivering such a quality output.Image 1 of 3(Image credit: Cyberpower)Image 1 of 3(Image credit: Cyberpower)Image 1 of 3(Image credit: Cyberpower)Image 1 of 3For smaller accessories and network equipment, the CyberPower EC650LCD is the best pick for most. Coming in at just over $70 for 390W/650VA, the EC650LCD has enough power reserves to keep the average home network alive for well over 15 minutes, which is enough time to exit your game/applications, save all your work through the network, and shut everything off gracefully during a power outage.The EC650LCD is small enough to hide away, taking up very little desktop room for a 390W unit. One of my favorite features of the EC650LCD is its array of ECO ports. These ports can be managed and timed to turn on or off, depending on your schedule or use case. The ECO ports will also power down accessories like your speakers and display if your PC is asleep or powered down.Best gaming PC (opens in new tab) | Best gaming laptop (opens in new tab)| Best gaming motherboards (opens in new tab) | Best SSD for gaming (opens in new tab) | Best DDR4 RAM (opens in new tab) | Best PC cases (opens in new tab) Image 1 of 3(Image credit: APC)Image 1 of 3(Image credit: APC)Image 1 of 3(Image credit: APC)Image 1 of 3The name APC is synonymous with high-quality UPS products. I use three of the company's Smart-UPS professional level units at home: two 1000VA units for my network and NAS gear and one 1500VA unit for my PC and displays.APC's BE600M1 is excellent at providing both battery and surge protection for the devices you use most every day. That is your phone and possibly a tablet. However, it has enough power reserves for you to plug in at least a router and a single display. If you only plan to use the BE600M1 to power a Wi-fi router, the unit will have enough juice in it to let you browse the internet in peace for several hours, even if the electricity in the rest of your house is gone. Priorities.The best part of the BE600M1 is its size. Most UPS are large and belong on the floor, but APC encourages you to put the BE600M1 on a desk. The unit provides a single 1.5A USB port for charging a phone or tablet, so you don't have to use your device's power adapter, which is inevitably a wall wart that potentially overlaps another socket or two, so you free up sockets for other devices.The best UPS for PC gaming FAQAn uninterruptible power supply that rates at 1500VA should run your computer for a little under an hour. But if you're trying to run your PC and your monitor from it, then you're probably looking at more like ten minutes of up-time. A 650VA, at peak load, will maybe net you something in the range of seven minutes, though that is obviously with a far lower peak wattage.There are two types of uninterruptible power supply to look out for when shopping around for your gaming PC: sine-wave and simulated sine-wave. Sine-wave UPS backups deliver a smooth, consistent oscillation of AC power directly to your PSU. These are often the only type of UPS you'll find recommended for gaming PCs due to their efficiency and clean power delivery.A pure sine-wave signal will be a match for the AC mains power your PSU is expected from your mains. Essentially, your PC shouldn't know the difference between your UPS battery power and the power coming from the wall. Simulated sine-wave UPS deliver a stepped, approximated waveform using pulse-width modulation (PWM). That's the same concept used to control PC case fan (opens in new tab) RPM. These are often far less expensive than pure sine-wave UPS and can be useful for peripherals, small devices, and monitors. However, since the waveform is not always exact, these may not function as intended with PSUs that demand a stable and consistent input. When your UPS detects a power surge or cut it will switch to battery power. How it delivers that battery power to your PC or accessories is when sine-wave versus simulated sine-wave matters most. That's because some PSUs will actually recognise a simulated sine-wave frequency and shut down suddenly to protect themselves from the unexpected oddity in power. Thus, your UPS won't actually be saving your PC from a loss of power. Adam Kropelin Kern Sibbald Apcupsd is a UPS control system that permits orderly shutdown of your computer in the event of a power failure. May 31, 2016 13:50:45 This manual documents apcupsd version 3.14.x Copyright © 2004-2015 Adam Kropelin Copyright © 1999-2005 Kern Sibbald Copying and distribution of this file, with or without modification, are permitted in any medium without royalty provided the name Apcupsd, the copyright notice, and this notice are preserved. Apcupsd source code is released under the GNU General Public License version 2. Please see the file COPYING in the main source directory. For more information on the project, please visit the main web site at No person should rely on the contents of the APCUPSD Manual ("the manual") without first obtaining advice from APC Technical Support. The manual is provided on the terms and understanding that: the authors, contributors and editors are not responsible for the results of any actions taken on the basis of information in the manual, nor for any error in or omission from the manual; and the authors, contributors and editors are not engaged in rendering technical or other advice or services. The the authors, contributors and editors, expressly disclaim all and any liability and responsibility to any person, whether a reader of the manual or not, in respect of anything, and of the consequences of anything, done or omitted to be done by any such person in reliance, whether wholly or partially, on the whole or any part of the contents of the manual. Without limiting the generality of the above, no author, contributor or editor shall have any responsibility for any act or omission of any other author, contributor or editor. This is the manual for apcupsd, a daemon for communicating with UPSes (Uninterruptible Power Supplies) made by American Power Conversion Corporation (APC). If you have an APC-made UPS, whether sold under the APC nameplate or OEMed (for example, the HP PowerTrust 2997A), and you want you get it working with a computer running Linux, Unix, or Windows, you are reading the right document. This manual is divided into parts which increase in technical depth as they go. If you have just bought a state-of-the-art smart UPS with a USB or Ethernet interface, and you are running a current version of Red Hat or SUSE Linux, then apcupsd is very nearly plug-and-play and you will have to read only the Basic User's Guide. If your operating system is older, or if you have an old-fashioned serial-line UPS, you'll have to read about serial installation (see Installation: Serial-Line UPSes). If you need more details about administration for unusual situations (such as a master/slave or multi-UPS setup) you'll need to read the sections on those topics as well. Finally, there are a number of technical reference sections which gives full details on things like configuration file directives and event-logging formats. You should begin by reading the Quick Start (see Quick Start for Beginners) instructions. apcupsd is a complex piece of software, but most of its complexities are meant for dealing with older hardware and operating systems. On current hardware and software getting it running should not be very complicated. The following is a help guide to the steps needed to get apcupsd set up and running as painlessly as possible. Check to see if apcupsd supports your UPS and cable (see Supported UPSes and Cables). Check to see if apcupsd supports your operating system (see Supported Operating Systems). Plan your configuration type (see Choosing a Configuration Type). If you have just one UPS and one computer, this is easy. If you have more than one machine being served by the same UPS, or more than one UPS supplying power to computers that are on the same local network, you have more choices to make. Figure out if you have one of the easy setups. If you have a USB UPS, and a supported operating system and you want to use one UPS with one computer, that's an easy setup. APC supplies the cable needed to talk with that UPS along with the UPS. All you need to do is check that your USB subsystem is working (see USB Configuration); if so, you can go to the build and install step. If you have a UPS designed to communicate via SNMP over Ethernet, that is also a relatively easy installation. Details are provided in Support for SNMP UPSes. If you have a UPS that communicates via an RS232C serial interface and it is a SmartUPS, then things are relatively simple, otherwise, your life is about to get interesting. If you have a vendor-supplied cable, find out what cable type you have by looking on the flat ends of the cable for a number, such as 940-0020A, stamped in the plastic. If you don't have a vendor-supplied cable, or your type is not supported, you may have to build one yourself (see Cables). Here is hoping you are good with a soldering iron! Now you are ready to read the Building and Installing (see Building and Installing apcupsd) section of the manual and follow those directions. If you are installing from an RPM or some other form of binary package, this step will probably consist of executing a single command. Tweak your /etc/apcupsd/apcupsd.conf file as necessary. Often it will not be. Change the BIOS settings (see Arranging for Reboot on Power-Up) on your computer so that boots up every time it gets power. (This is not the default on most systems.) To verify that your UPS is communicating with your computer and will do the right thing when the power goes out, read and follow the instructions in the Testing (see Testing Apcupsd) section. If you run into problems, check the apcupsd users' email list archive for similar problems. This is an excellent resource with answers to all sorts of questions. See . If you still need help, send a message to the apcupsd users' email list (apcupsd-users@lists.sourceforge.net) describing your problem, what version of apcupsd you are using, what operating system you are using, and anything else you think might be helpful. Read the manual section on Monitoring and Tuning your UPS. apcupsd supports many UNIX-like operating systems as well as several variants of Windows. Due to lack of API standardization, USB support is not available on every platform. See Platform Support below for details. In general it is recommended to obtain a prebuilt package for your platform. Given how apcupsd must integrate into the shutdown mechanism of the operating system and the rate at which such mechanisms are changed by vendors, the platform ports in the apcupsd tree may become out of date. In some cases, binary packages are provided by the apcupsd team (RedHat, Mandriva, SuSE, Windows, Mac OS X). For other platforms it is recommended to check your vendor's package repository and third party repositories for recent binary packages. Note that some vendors continue to distribute ancient versions of apcupsd with known defects. These packages should not be used. LINUX WINDOWS Windows NT 4 [2] [4] Windows 98/ME/2000 [2] [4] Windows XP/Vista (including 64 bit) [1] [2] Windows Server 2003/2008 (including 64 bit) [2] Windows 7 [2] OTHERS apcupsd supports nearly every APC brand UPS model in existence and enough different cable types to connect to all of them. The UPSTYPE field is the value you will put in your /etc/apcupsd/apcupsd.conf file to tell apcupsd what type of UPS you have. We'll describe the possible values here, because they're a good way to explain your UPS's single most important interface property: the kind of protocol it uses to talk with its computer. apcsmart The 'apcsmart' protocol uses an RS232 serial connection to pass commands back and forth in a primitive language resembling modem-control codes. APC calls this language "UPS-Link". Originally introduced for Smart-UPS models (thus the name 'apcsmart'), this class of UPS is in decline, rapidly being replaced in APC's product line by USB and MODBUS UPSes. usb A USB UPS speaks a universal well defined control language over a USB wire. Most of APC's lineup now uses this method as of late 2003, and it seems likely to completely take over in their low- and middle range. The most recent APC UPSes support only a limited set of data over the USB interface. MODBUS (see below) is required in order to access the advanced data. net This is the keyword to specify if you are using your UPS in Slave mode (i.e. the machine is not directly connected to the UPS, but to another machine which is), and it is connected to the Master via an ethernet connection. You must have apcupsd's Network Information Services NIS turned on for this mode to work. snmp SNMP UPSes communicate via an Ethernet NIC and firmware that speaks Simple Network Management Protocol. dumb A dumb or voltage-signaling UPS and its computer communicate through the control lines (not the data lines) on an RS232C serial connection. Not much can actually be conveyed this way other than an order to shut down. Voltage-signaling UPSes are obsolete; you are unlikely to encounter one other than as legacy hardware. If you have a choice, we recommend you avoid simple signalling UPSes. pcnet PCNET is an alternative for SNMP available on APC's AP9617 family of smart slot modules. The protocol is much simpler and potentially more secure than SNMP. modbus MODBUS is the newest APC protocol and operates over RS232 serial links or USB. MODBUS is APC's replacement for the aging 'apcsmart' (aka UPS-Link) protocol. MODBUS is the only way to access detailed control and status information on newer (esp. SMT series) UPSes. There are three major ways of running apcupsd on your system. The first is a standalone configuration where apcupsd controls a single UPS, which powers a single computer. This is the most common configuration. If you're working with just one machine and one UPS, skip the rest of this section. Your choices become more interesting if you are running a small cluster or a big server farm. Under those circumstances, it may not be possible or even desirable to pair a UPS with every single machine. apcupsd supports some alternate arrangements. The second type of configuration is the NIS (Network Information Server) server and client. In this configuration, where one UPS powers several computers, a copy of apcupsd running one one computer will act as a server while the other(s) will act as network clients which poll the server for information about the UPS. Note that "NIS" is not related to Sun's directory service also called "NIS" or "Yellow Pages". The third configuration is where a single computer controls multiple UPSes. In this case, there are several instances of apcupsd on the same computer, each controlling a different UPS. One instance of apcupsd will run in standalone mode, and the other instance will normally run in network mode. This type of configuration may be appropriate for large server farms that use one dedicated machine for monitoring and diagnostics Here is a diagram that summarizes the possibilities: If you decide to set up one of these more complex configurations, see the dedicated section on that particular configuration. Apcupsd supports USB connections on all major operating systems: Linux, FreeBSD, OpenBSD, NetBSD, Windows, Solaris, and Mac OS X Darwin. If you plan to use a USB connection, please read the appropriate subsection in its entirety. You can skip this section if your UPS has a serial (RS232-C) or Ethernet interface or if you are not running one of the platforms listed above. Problem Linux 2.4 series kernels older than 2.4.22 (RH 9, RHEL 3) do not bind the USB device to the proper driver. This is evidenced by /proc/bus/usb/devices listing the UPS correctly but it will have "driver=(none)" instead of "driver=(hid)". This affects RHEL3, among others. Workaround Upgrade linux kernel to 2.4.22 or higher. Alternately, you apply the linux-2.4.20-killpower.patch and linux-2.4.20-USB-reject.patch patches to your kernel and rebuild it. These patches can be found in the examples/ directory in the apcupsd source distribution. Problem Mandrake 10.0 and 10.1 systems with high security mode enabled (running kernel-secure kernel) use static device nodes but still assign USB minor numbers dynamically. This is evidenced by hiddev0: USB HID v1.10 Device [...] instead of hiddev96: ... in dmesg log. Workaround Boot standard kernel instead of kernel-secure or disable CONFIG_USB_DYNAMIC_MINORS and rebuild kernel-secure. Problem USB driver linux-usb.c fails to compile, reporting errors about HID_MAX_USAGES undefined. This is due to a defect in the linux kernel hiddev.h header file on 2.6.5 and higher kernels. Workaround Upgrade to apcupsd-3.10.14 or higher. These versions contain a workaround for the defect. Problem On some systems such as Slackware 10.0, no USB devices will show up (see the next section). Workaround Add the following to rc.local mount -t usbdevfs none /proc/bus/usb Problem 2.6 kernels use udev and some distributions to not configure it to automatically create /dev/usb/hiddev?? as they should, causing apcupsd to fail to locate the UPS. Workaround Edit the file /etc/udev/rules.d/50-udev.rules, and add the following: KERNEL="hiddev*", NAME="usb/hiddev%n" More details are provided in the following section ... To make sure that your USB subsystem can see the UPS, just do this from a shell prompt: cat /proc/bus/usb/devices This information is updated by the kernel whenever a device is plugged in or unplugged, irrespective of whether apcupsd is running or not. It contains details on all the USB devices in your system including hubs (internal and external), input devices, and UPSes. You should get some output back that includes something like this, featuring a BackUPS RS 1000: T: Bus=02 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#= 3 Spd=1.5 MxCh= 0 D: Ver= 1.10 Cls=00(>ifc) Sub=00 Prot=00 MxPS= 8 #Cfgs= 1 P: Vendor=051d ProdID=0002 Rev= 1.06 S: Manufacturer=American Power Conversion S: Product=Back-UPS RS 1000 FW:7.g3 .D USB FW:g3 S: SerialNumber=JB0308036505 C:* #Ifs= 1 Cfg#= 1 Atr=a0 MxPwr= 24mA I: If#= 0 Alt= 0 #EPs= 1 Cls=03(HID) Sub=00 Prot=00 Driver=hid The important things to check for are the S: lines describing your UPS and and the I: line showing what driver is handling it. If on the I: line, Driver is listed as Driver=none then you do not have the HID driver loaded or the driver did not attach to the UPS. One common cause is having a Linux kernel older than 2.4.22 (such as a stock RedHat 9 or RHEL 3 kernel). If this is the case for your system, please upgrade to at least kernel version 2.4.22 and try again. If you are already running a 2.4.22 or higher kernel, please read further for instructions for other possible courses of action. Here is another example, this time featuring a Back-UPS 350: T: Bus=01 Lev=01 Prnt=01 Port=00 Cnt=01 Dev#= 2 Spd=1.5 MxCh= 0 D: Ver= 1.10 Cls=00(>ifc) Sub=00 Prot=00 MxPS= 8 #Cfgs= 1 P: Vendor=051d ProdID=0002 Rev= 1.00 S: Manufacturer=American Power Conversion S: Product=Back-UPS 350 FW: 5.2.I USB FW: c1 S: SerialNumber=BB0115017954 C:* #Ifs= 1 Cfg#= 1 Atr=a0 MxPwr= 30mA I: If#= 0 Alt= 0 #EPs= 1 Cls=03(HID) Sub=00 Prot=00 Driver=hid E: Ad=81(I) Atr=03(Int.) MxPS= 8 Ivl= 10ms In general, if you see your UPS model in the S: field, which means Manufacturer=, Product=, and SerialNumber=, and you see Driver=hid in the I: field, you know the UPS has been recognized and is bound to the correct driver. If your UPS doesn't appear in the list at all, check the obvious things: The UPS must be powered on, and a cable must be properly seated in both the data port of the UPS and one of your machine's USB ports. Many UPSes have phone ports to provide surge protection for phones or modems -- make sure you haven't plugged your USB cable into one of those rather than the data port (which will usually be near the top edge of the case.) Also, ensure that the correct drivers are loaded. Under Linux-2.4.x, you can check this out easily by examining the /proc/bus/usb/drivers file. Here's how you can do that: cat /proc/bus/usb/drivers ...and you should get: usbdevfs hub 96-111: hiddev hid On Linux-2.6.x, make sure the sysfs filesystem is mounted on /sys and do: ls -l /sys/bus/usb/drivers/ ...where you should get: total 0 drwxr-xr-x 2 root root 0 May 1 18:55 hid drwxr-xr-x 2 root root 0 May 1 18:55 hiddev drwxr-xr-x 2 root root 0 May 1 18:55 hub drwxr-xr-x 2 root root 0 May 1 18:55 usb drwxr-xr-x 2 root root 0 May 1 18:55 usbfs ...or perhaps something like: total 0 drwxr-xr-x 2 root root 0 Jan 6 15:27 hiddev drwxr-xr-x 2 root root 0 Jan 6 15:28 hub drwxr-xr-x 2 root root 0 Jan 6 15:28 usb drwxr-xr-x 2 root root 0 Jan 6 15:27 usbfs drwxr-xr-x 2 root root 0 Jan 6 15:28 usbhid If your 2.6.x system does not have the /sys/bus/usb directory, either you do not have sysfs mounted on /sys or the USB module(s) have not been loaded. (Check /proc/mounts to make sure sysfs is mounted.) A USB UPS needs all of these drivers -- the USB device filesystem, the USB hub, the Human Interface Device subsystem driver, and the Human Interface Device driver. If you are compiling your own kernel, you want to enable CONFIG_USB CONFIG_USB_HID CONFIG_USB_HIDDEV CONFIG_USB_DEVICEFS ...as well as at least one USB Host Controller Driver... CONFIG_USB_UHCI_HCD (linux-2.6.x) CONFIG_USB_OHCI_HCD (linux-2.6.x) CONFIG_USB_UHCI (linux-2.4.x) CONFIG_USB_OHCI (linux-2.4.x) Apcupsd accesses USB UPSes via the hiddev device nodes. Typically these are located in /dev/hiddevN, /dev/usb/hiddevN or /dev/usb/hiddev/hiddevN (where N is a digit 0 thru 9). Some distributions (some Debian releases, possibly others) do not provides these device nodes for you, so you will have to make them yourself. Check /dev, /dev/usb, and /dev/usb/hiddev and if you cannot find the hiddevN nodes, run (as root) the examples/make-hiddev script from the apcupsd source distribution. Modern Linux distributions using the 2.6 kernel create device nodes dynamically on the fly as they are needed. It is basically a hotplug system, giving a lot more power to the user to determine what happens when a device is probed or opened. It is also a lot more complicated. Some early 2.6 distributions (Fedora Core 3, for one) do not include hiddev rules in their default udev rule set. The bottom line for apcupsd on such a system is that if the hiddevN is not created when you plug in your UPS, apcupsd will terminate with an error. The solution to the problem is to add a rule to the udev rules file. On Fedora FC3, this file is found in /etc/udev/rules.d/50-udev.rules. Start by adding the following line: BUS="usb", SYSFS{idVendor}="051d", NAME="usb/hiddev%n" Note that this rule uses obsolete udev syntax and is specific to FC3 and other distributions of similar vintage. Then either reboot your system, or unplug and replug your UPS and then restart apcupsd. At that point a /dev/usb/hiddevN node should appear and apcupsd should work fine. If you have several UPSes or you just want to give your UPS a fixed name, you can use rules like the following: KERNEL=="hiddev*", SYSFS{serial}=="JB0319033692", SYMLINK="ups0" KERNEL=="hiddev*", SYSFS{serial}=="JB0320004845", SYMLINK="ups1" Note that this rule uses udev syntax that is appropriate only for distros such as RHEL4 and FC4 and others of a similar vintage. More recent distros such as FC15 should use something like this: KERNEL=="hiddev*", ATTRS{manufacturer}=="American Power Conversion", ATTRS{serial}=="BB0100009999 ", OWNER="root", SYMLINK+="ups0" Replace the serial number in quotes with the one that corresponds to your UPS. Then whenever you plug in your UPS a symlink called ups0, ups1, etc. will be created pointing to the correct hiddev node. This technique is highly recommended if you have more than one UPS connected to the same server since rearranging your USB cables or even upgrading the kernel can affect the order in which devices are detected and thus change which hiddev node corresponds to which UPS. If you use the symlink-by-serial-number approach the link will always point to the correct device node. You can use... udevinfo -a -p /sys/class/usb/hiddev0/ ...to get more information on the fields that can be matched besides serial number. To find the available attributes to match (note that the serial is NOT always the UPS serial on the box or in the USB connect message in /var/log/messages), use: udevadm info --attribute-walk --name=/dev/usb/hiddev0 An additional device-node-related problem is the use of dynamic minors. Some distributions, such as Mandrake 10, ship with a kernel having CONFIG_USB_DYNAMIC_MINORS turned on. This is not ideal for running with apcupsd, and the easiest solution is to turn CONFIG_USB_DYNAMIC_MINORS off and rebuild your kernel, or find a pre-built kernel with it off. For a kernel with CONFIG_USB_DYNAMIC_MINORS turned on to work with apcupsd, you must enable devfs. The following will tell you if devfs is enabled: $ ps ax | grep devs ...which should give something like the following: 533 ? S 0:00 devfsd /dev What complicates the situation much more on Mandrake kernels is their security level since CONFIG_DYNAMIC_USB_MINORS is turned on, but on higher security levels devfs is turned off. The net result, is that in those situations hiddev is completely unusable so apcupsd will not work. So, in these cases, the choices are: Reduce the security level setting of the system (not sure if this is possible after the initial install). Custom build a high security kernel with devfs enabled and make sure devfs is mounted and devfsd is running. Custom build a high security kernel with dynamic minors disabled Use udev If all these things check out and you still can't see the UPS, something is more seriously wrong than this manual can cover -- find expert help. If you are unable to list USB devices or drivers, you kernel may not be USB-capable and that needs to be fixed. Problem FreeBSD lockups: Some users have experienced lockups (apcupsd stops responding) on FreeBSD systems. Solution Recent versions of Apcupsd have addressed this issue. Please upgrade to apcupsd-3.10.18 or higher. Problem FreeBSD kernel panics if USB cable is unplugged while apcupsd is running. Solution This is a kernel bug and is most easily worked around by not hot- unplugging the UPS while apcupsd is running. This issue may be fixed in recent FreeBSD kernels. The *BSD USB driver supports FreeBSD, OpenBSD and NetBSD. (Thanks go to the *BSD developers who kept a nearly identical interface across all three platforms.) Users of OpenBSD, NetBSD, and some versions of FreeBSD will need to rebuild the kernel in order to enable the ugen driver and disable the uhid driver. uhid is not sufficient for apcupsd at this time and we need to prevent it from grabbing the UPS device. You should make the following changes to your kernel config file: FreeBSD (v5.4 and below, v6.0) (you will not lose use of USB keyboard and mouse) Disable: uhid Enable: ugen FreeBSD (v5.5, v6.1 and above) (you will not lose use of USB keyboard and mouse) Disable: (nothing) Enable: ugen This is the default configuration for a GENERIC kernel on many platforms so you most likely will not need to recompile. NetBSD (v3.x and below) (you will lose use of USB keyboard and mouse) Disable: uhidev, ums, wsmouse, ukbd, wskbd, uhid Enable: ugen NetBSD (v4.0 and above) You can use apcupsd on single USB port without disabling the USB keyboard and mouse on other ports, though all other devices will be disabled on the port you pick for your UPS. First, decide which hub and port you wish to use. You can find out the hub and port numbers for any particular physical connector by plugging a USB device into it and looking at the messages printed by the kernel; you should messages something like this: uxx0 at uhub0 port 1 uxx0: To use your APC UPS on this port, configure the kernel to prefer attachment of the ugen driver over other drivers on this hub and port only, by adding a line like this to your kernel config file: ugen* at uhub0 port 1 flags 1 (The "flags 1" forces the ugen to attach instead of anything else detected there.) Configure and build that kernel as per the references below, and your UPS will now attach as a ugen device when plugged into that port. Don't forget to 'cd /dev' and './MAKEDEV ugen1' (and 2 and so on) if you have more than one generic usb device on your system. OpenBSD (you will lose use of USB keyboard and mouse): Disable: uhidev, ums, wsmouse, ukbd, wskbd, uhid Enable: ugen For detailed information on rebuilding your kernel, consult these references: FreeBSD NetBSD OpenBSD After building a properly configured kernel, reboot into that kernel and plug in your UPS USB cable. You should see a dmesg log message like the following: ugen0: American Power Conversion Back-UPS RS 1500 FW:8.g6 .D USB FW:g6, rev 1.10/1.06, addr 2 Note that the ugen driver is called out. If you see uhid instead, it probably means you did not properly disable the uhid driver when you compiled your kernel or perhaps you're not running the new kernel. You can also check with 'usbdevs -d' to get a list of USB devices recognized by the system as well as the drivers they are associated with. For example: # usbdevs -d addr 1: UHCI root hub, VIA uhub0 addr 2: Back-UPS RS 1500 FW:8.g6 .D USB FW:g6, American Power Conversion ugen0 Apcupsd communicates with the UPS through the USB generic device, ugen. You may or may not need to manually make ugen device nodes in /dev, depending on what OS you are using. FreeBSD No manual intervention needed. FreeBSD automatically creates the ugen nodes on demand. NetBSD By default, NetBSD only creates nodes for the first ugen device, ugen0. Check usbdevs -d to see which device your UPS was bound to and then create the appropriate node by running 'cd /dev ; ./MAKEDEV ugenN', where ugenN is the ugen device name shown by usbdevs. It is probably a good idea to create several sets of ugen nodes in case you add more USB devices. OpenBSD Similar to NetBSD, OpenBSD creates nodes for ugen0 and ugen1. Check usbdevs -d to see which device your UPS was bound to and then create the appropriate node by running 'cd /dev ; ./MAKEDEV ugenN', where ugenN is the ugen device name shown by usbdevs. It is probably a good idea to create several sets of ugen nodes in case you add more USB devices. Apcupsd supports USB UPSes on Windows XP and newer, including 64 bit systems. USB connected UPSes on Windows require a special driver. In most cases, this driver is automatically installed when you install Apcupsd. However in some cases you may need to install the driver manually. For detailed instructions, please see the install.txt file located in the driver folder of your Apcupsd install. After installing Apcupsd (and the Apcupsd USB driver, if necessary), plug in your UPS USB cable and open the Windows Device Manager. You should see a American Power Conversion USB UPS (Apcupsd) listed under the Batteries section. If a device of that name does not appear, check that your UPS is powered on and that the USB cable is connected at both ends. Reinstall the driver as directed above if needed. Apcupsd supports USB UPSes on Solaris 10 and higher. Both x86 and SPARC platforms are supported. Some specific packages are necessary when building Apcupsd with USB support on Solaris. You must install the SUNWlibusb and SUNWlibusbugen packages BEFORE attempting to build Apcupsd. These packages can be found on the Solaris installation CDROMs and should be installed with the pkgadd utility. You also should build using the gcc compiler and ccs make, not Sun's compiler. The appropriate make utility can be found in /usr/ccs/bin. gcc can be installed from packages included on the Solaris installation CDROMs. Configure and build Apcupsd normally, as described in Building and Installing Apcupsd. Be sure to include the --enable-usb flag to configure. After building, install Apcupsd as root using 'make install', then perform a reconfigure boot ('reboot -- -r'). During installation, Apcupsd will automatically configure your USB subsystem to attach APC USB devices to the ugen driver. This is a critical step and must be completed by a reconfigure boot. Note that the USB config changes will be reversed if you remove Apcupsd using 'make uninstall'. After installing Apcupsd as described above and performing a reconfigure boot, plug in your UPS USB cable. You should see a series of dmesg log messages similar to the following: Dec 5 17:50:50 sunblade usba: [ID 912658 kern.info] USB 1.10 device (usb51d,2) operating at low speed (USB 1.x) on USB 1.10 root hub: input@4, ugen0 at bus address 3 Dec 5 17:50:50 sunblade usba: [ID 349649 kern.info] American Power Conversion Smart-UPS 1000 FW:600.1.D USB FW:1.2 AS0127232356 Dec 5 17:50:50 sunblade genunix: [ID 936769 kern.info] ugen0 is /pci@1f,0/usb@c,3/input@4 Dec 5 17:50:50 sunblade genunix: [ID 408114 kern.info] /pci@1f,0/usb@c,3/input@4 (ugen0) online Note that the ugen driver is called out. If you do not see any dmesg entries related to your UPS, ensure that it is turned on and that the USB cable is connected at both ends. Also verify that you installed Apcupsd as root using the 'make install' command and that you performed a reconfigure boot afterward. Apcupsd communicates with the UPS through the USB generic device, ugen. The reconfigure boot performed after Apcupsd installation will ensure the correct device nodes are created. Once your UPS has been recognized in dmesg as shown above, you can check /dev/usb to see if the device nodes have appeared: [user@sunblade /]$ ls /dev/usb/51d.2/* cntrl0 cntrl0stat devstat if0in1 if0in1stat (51d.2 is the vendor/product id for APC UPSes.) Apcupsd supports USB UPSes on Mac OS X (Darwin) 10.4.x and higher. Both Intel and PowerPC platforms are supported. Some specific packages are necessary when building Apcupsd with USB support on Darwin. You must install libusb-0.1.12 which can be obtained from MacPorts ( ) (formerly DarwinPorts) or Fink ( ) or downloaded and built by hand ( ). You must not use libusb-1.x or higher (apcupsd does not support the new 1.0 APIs) nor any version earlier than 0.1.12 (earlier versions have a bug that apcupsd triggers). Generally that means you must use exactly 0.1.12. Note that Apcupsd is sensitive to the install location of libusb, so beware if you change it from the default. Apcupsd should be built using gcc, preferably from the XCode development tools. Currently the maintainer is using gcc-4.0.1 from XCode 2.4. Other versions of gcc from other sources may also work. Configure and build Apcupsd normally, as described in Building and Installing Apcupsd. Be sure to include the --enable-usb flag to configure. After building, install Apcupsd as root using 'make install' and then reboot. During installation, Apcupsd will automatically install a simple dummy kext driver designed to prevent Apple's monitoring software from taking over the UPS. It is necessary to reboot in order to activate the kext. Note that this kext will be automatically removed if you uninstall Apcupsd using 'make uninstall', allowing Apple's monitoring tool to once again access the UPS. After installing Apcupsd as described above and rebooting, plug in your UPS USB cable. You should notice that Darwin does NOT display the battery monitor tool in the menu bar. You can also check Apple Menu -> About This Mac -> More Info... -> USB to ensure that your UPS appears in the list of USB devices. In general it is recommended to obtain a prebuilt binary package for your platform. Given how apcupsd must integrate into the shutdown mechanism of the operating system and the rate at which such mechanisms are changed by vendors, the platform ports in the apcupsd tree may become out of date. In some cases, binary packages are provided by the apcupsd team (RedHat, Mandriva, SuSE, Windows, Mac OS X). For other platforms it is recommended to check your vendor's package repository and third party repositories for recent binary packages before resorting to building apcupsd from scratch. Note that some vendors continue to distribute ancient versions of apcupsd with known defects. These packages should not be used. For systems based on RPM packages, such as Red Hat and SuSE, apcupsd is available in binary RPM format. This is the simplest way to install. If you have no previous version of apcupsd on your machine and are creating a standalone configuration, simply install the RPM with a normal 'rpm -ihv' command. You're done, and can now skip the rest of this chapter and go straight to tweaking your run-time configuration file. (see After Installation) If you have a previous installation, you can upgrade with a normal 'rpm -Uhv', but this may not upgrade the halt script. It may be better to do the upgrade as a remove 'rpm -e' followed by a fresh install 'rpm -ihv'. After installation of the binary RPM, please verify carefully that /etc/rc.d/init.d/halt was properly updated and contains new script lines flagged with ***APCUPSD***. Since there is no standard location for cgi-bin, the rpm will place the binary CGI programs in the directory /etc/apcupsd/cgi. To actually use them, you must copy or move them to your actual cgi-bin directory, which on many systems is located in /home/httpd/cgi-bin. The Windows version of apcupsd is distributed as a simple double-click installer. Installation is very simple and straight-forward: Simply double-click the installer executable and follow the instructions. See The Windows Version of apcupsd for further details. Installation from source might have to be done different ways depending on what system you are running. The basic procedure involves getting a source distribution, running the configuration, rebuilding, and installing. For building the system, we suggest that you run the configure and make processes as your normal UNIX user ID. The 'make install' must be run as root. But if your normal ID has an environment setup for using the C compiler, it's simpler to do that than to set up root to have the correct environment. apcupsd requires gcc and g++ compilers as well as GNU make. Other compilers or BSD make will not work. GNU make is sometimes installed as gmake. The configure script will check for this and will inform you of what command to use to invoke GNU make. The basic installation from a tar source file is rather simple: Unpack the source code from its tar archive. Go into the directory containing the source code. Run './configure' (with appropriate options as described below) 'make' or 'gmake'' as instructed by configure 'su' (i.e. become root) Stop any running instance of apcupsd. The command to do this will look like 'system-dependent-path/apcupsd stop' uninstall any old apcupsd This is important since the default install locations may have changed. 'make install' or 'gmake install' edit your /etc/apcupsd/apcupsd.conf file if necessary ensure that your halt script is properly updated Start the new apcupsd with: 'system-dependent-path/apcupsd start' If all goes well, the './configure' will correctly determine which operating system you are running and configure the source code appropriately. configure currently recognizes the systems listed below in the Operating System Specifics section of this chapter and adapts the configuration appropriately. Check that the configuration report printed at the end of the configure process corresponds to your choice of directories, options, and that it has correctly detected your operating system. If not, redo the configure with the appropriate options until your configuration is correct. Please note that a number of the configure options preset apcupsd.conf directive values in an attempt to automatically adapt apcupsd as best possible to your system. You can change the values in apcupsd.conf at a later time without redoing the configuration process by simply editing the apcupsd.conf file. Other configuration options can be used to set up the installation of HTML documentation and optional modules, notably the CGI interface that enables the UPS state to be queried via the Web. You will find a complete reference later in this chapter. In general, you will probably want to supply a more elaborate configure statement to ensure that the modules you want are built and that everything is placed into the correct directories. On Red Hat, a fairly typical configuration command would look like the following: CFLAGS="-g -O2" LDFLAGS="-g" ./configure \ --enable-usb \ --with-upstype=usb \ --with-upscable=usb \ --prefix=/usr \ --sbindir=/sbin \ --with-cgi-bin=/var/www/cgi-bin \ --enable-cgi \ --with-log-dir=/etc/apcupsd By default, 'make install' will install the executable files in /sbin, the manuals in /usr/man, and the configuration and script files in /etc/apcupsd. In addition, if your system is recognized, certain files such as the startup script and the system halt script will be placed in appropriate system directories (usually subdirectories of /etc/rc.d). There are a number of things that you can do to check if the installation (make install) went well. The fist is to check where the system has installed apcupsd using 'which' and 'whereis'. On my Red Hat system, you should get the following (lines preceded with a $ indicate what you type): $ which apcupsd /sbin/apcupsd $ whereis apcupsd apcupsd: /sbin/apcupsd /etc/apcupsd /etc/apcupsd.conf /etc/apcupsd.status /usr/man/man8/apcupsd.8.gz /usr/man/man8/apcupsd.8 If you find an apcupsd in /usr/sbin, /usr/local/sbin, /usr/lib, or another such directory, it is probably a piece of an old version of apcupsd that you can delete. If you are in doubt, delete it, then rerun the 'make install' to ensure that you haven't deleted anything needed by the new apcupsd. Please note that the files specified above assume the default installation locations. As a final check that the 'make install' went well, you should check your halt script (in /etc/rc.d on SUSE systems, and in /etc/rc.d/init.d on Red Hat systems) to see that the appropriate lines have been inserted in the correct place. Modification of the halt script is important so that at the end of the shutdown procedure, apcupsd will be called again to command the UPS to turn off the power. This should only be done in a power failure situation as indicated by the presence of the /etc/powerfail file, and is necessary if you want your machine to automatically be restarted when the power returns. On a Red Hat system, the lines containing the # ***apcupsd*** should be inserted just before the final halt command: # Remount read only anything that's left mounted. #echo "Remounting remaining filesystems (if any) readonly" mount | awk '/ext2/ { print $3 }' | while read line; do mount -n -o ro,remount $line done # See if this is a powerfail situation. # ***apcupsd*** if [-f /etc/apcupsd/powerfail]; then # ***apcupsd*** echo # ***apcupsd*** echo "APCUPSD will now power off the UPS" # ***apcupsd*** echo # ***apcupsd*** /etc/apcupsd/apccontrol killpower # ***apcupsd*** echo "Please ensure that the UPS has powered off before rebooting" # ***apcupsd*** echo "Otherwise, the UPS may cut the power during the reboot!!!" # ***apcupsd*** echo # ***apcupsd*** fi # ***apcupsd*** # Now halt or reboot. echo "$message" if [-f /fastboot]; then echo "On the next boot fsck will be skipped." elif [-f /forcefsck]; then echo "On the next boot fsck will be forced." fi The purpose of modifying the system halt files is so that apcupsd will be recalled after the system is in a stable state. At that point, apcupsd will instruct the UPS to shut off the power. This is necessary if you wish your system to automatically reboot when the mains power is restored. If you prefer to manually reboot your system, you can skip this final system dependent installation step by specifying the disable-install-distdir option on the './configure' command (see below for more details). The above pertains to Red Hat systems only. There are significant differences in the procedures on each system, as well as the location of the halt script. Also, the information that is inserted in your halt script varies from system to system. Other systems such as Solaris require you the make the changes manually, which has the advantage that you won't have any unpleasant surprises in your halt script should things go wrong. Please consult the specific system dependent README files for more details. Please note that if you install from RPMs for a slave machine, you will need to remove the changes that the RPM install script made (similar to what is noted above) to the halt script. This is because on a slave machine there is no connection to the UPS, so there is no need to attempt to power off the UPS. That will be done by the master. All the available configure options can be printed by entering: ./configure --help When specifying options for './configure', if in doubt, don't put anything, since normally the configuration process will determine the proper settings for your system. The advantage of these options is that it permits you to customize your version of apcupsd. If you save the './configure' command that you use to create apcupsd, you can quickly reset the same customization in the next version of apcupsd by simply re-using the same command. The following command line options are available for configure to customize your installation. --prefix=path This defines the directory for the non-executable files such as the manuals. The default is /usr. --sbindir=path This defines the directory for the executable files such as apcupsd. The default is /sbin. You may be tempted to place the executable files in /usr/sbin or /usr/local/sbin. Please use caution here as these directories may be unmounted during a shutdown and thus may prevent the halt script from calling apcupsd to turn off the UPS power. Though your data will be protected, in this case, your system will probably not be automatically rebooted when the power returns --enable-cgi This enables the building of the CGI programs that permit Web browser access to apcupsd data. This option is not necessary for the proper execution of apcupsd. --with-cgi-bin=path The with-cgi-bin configuration option allows you to define the directory where the CGI programs will be installed. The default is /etc/apcupsd, which is probably not what you want. --enable-apcsmart Turns on generation of the APC Smart driver (default). --enable-dumb Turns on generation of the dumb signalling driver code (default). --enable-usb Turns on generation of the USB driver code. By default this is disabled. --enable-net Turns on generation of the NIS network driver for slaves. For each slave, this is the only driver needed. This driver works by reading the information from the configured master using the NIS (Network Information Services) interface. --enable-snmp Turns on generation of the SNMP driver. This driver accesses the UPS over the network using SNMP. This is compatible only with UPSes equipped with an SNMP or Web/SNMP management card. By default this is enabled. --enable-pcnet Turns on generation of the PCNET (PowerChute Network Shutdown) driver. This driver accesses the UPS over the network using APC's custom protocol. This driver can be used as an alternative to SNMP for UPSes equipped with a modern Web/SNMP management card. --enable-modbus Turns on generation of the MODBUS/RS232 driver (default) --enable-modbus-usb Turns on generation of the MODBUS/USB driver --enable-test This turns on a test driver that is used only for debugging. By default it is disabled. --enable-gapcmon This option enables building the GTK GUI front-end for apcupsd. Building this package requires numerous GNOME libraries. The default is disabled. --enable-apcagent This option enables building the apcagent menubar application on Mac OS X platforms. The default is disabled. --with-libwrap=path, --with-libwrap This option when enabled causes apcupsd to be built with the TCP WRAPPER library for enhanced security. In most cases, the path is optional since configure will determine where the libraries are on most systems. --with-nologin=path This option allows you to specify where apcupsd will create the nologin file when logins are prohibited. The default is /etc --with-pid-dir=path This option allows you to specify where apcupsd will create the process id (PID) file to prevent multiple copies from running. The default is system dependent but usually /var/run. --with-log-dir=path This option allows you to specify where apcupsd will create the EVENTS and STATUS log files. The default is /etc/apcupsd. This option simply sets the default of the appropriate path in the apcupsd.conf file, which can be changed at any later time. --with-lock-dir=path This option allows you to specify where apcupsd will create the serial port lock file. The default is system-dependent but usually /var/lock. This option simply sets the appropriate path in the apcupsd.conf file, which can be changed at any later time. --with-pwrfail-dir=path This option allows you to specify where apcupsd will create the powerfail file when a power failure occurs. The default is system dependent but usually /etc. --with-serial-dev=device-name This option allows you to specify where apcupsd will look for the serial device that talks to the UPS. The default is system dependent, but often /dev/ttyS0. This option simply sets the appropriate device name in the apcupsd.conf file, which can be changed at any later time. --with-nis-port=port This option allows you to specify what port apcupsd will use for the Network Information Server (the CGI programs). The default is system dependent but usually 3551 because that port has been officially assigned to apcupsd by the IANA. This option simply sets the appropriate port in the apcupsd.conf file, which can be changed at any later time. --with-nisip=ip-address This option allows you to specify the value that will be placed on then NISIP directive in the configuration file. The default is 0.0.0.0. No checking is done on the value entered, so you must ensure that it is a valid IP address. --with-net-port=port This option allows you to specify what port apcupsd will use for Master and Slave communications. The default is system dependent but usually 6666. This option simply sets the appropriate port in the apcupsd.conf file, which can be changed at any later time. --with-upstype=type This option allows you to specify the type of UPS that will be connected to your computer. The default is: smartups. This option simply sets the appropriate UPS type in the apcupsd.conf file, which can be changed at any later time. --with-upscable=cable This option allows you to specify what cable you are using to connect to the UPS. The default is: smart. This option simply sets the appropriate UPS cable in the apcupsd.conf file, which can be changed at any later time. --disable-install-distdir This option modifies the apcupsd Makefiles disable installation of the distribution (platform) directory. Generally, this used to do a full installation of apcupsd except the final modification of the operating system files (normally /etc/rc.d/halt, etc.). This is useful if your operating system is not directly supported by apcupsd or if you want to run two

copies of apcupsd on the same system. This option can also be used by those of you who prefer to manually reboot your system after a power failure or who do not want to modify your system halt files. For most systems, we recommend the following options: ./configure --prefix=/usr --sbindir=/sbin --enable-usb and you can optionally build and install the CGI programs as follows: ./configure --prefix=/usr --sbindir=/sbin --enable-usb \ --enable-cgi --with-cgi-bin=/home/httpd/cgi-bin Some systems require unusual options for compilation or linking that the './configure' script does not know about. You can specify initial values for variables by setting them in the environment. Using a Bourne-compatible shell, you can do that on the command line like this: CFLAGS="-O2 -Wall" LDFLAGS= ./configure Or on systems that have the env program, you can do it like this: env CPPFLAGS=-I/usr/local/include LDFLAGS=-s ./configure Or for example on the Sun Solaris system, you can use: setenv CFLAGS -O2 setenv LDFLAGS -O ./configure You can get a listing of all available options by doing: ./configure --help or simply see the previous section of this manual. With the exception of Linux SUSE and Linux Red Hat systems used by the developers, we rely on users to help create installation scripts and instructions as well as to test that apcupsd runs correctly on their system. As you can imagine, most of these people are system administrators rather than developers so they are very busy and don't always have time to test the latest releases. With that in mind, we believe that you will find that a lot of very valuable work has been already done to make your installation much easier (and probably totally automatic). Below, you will find a list of operating systems for which we have received installation files: This port is complete and is being used by several users. You will need to install and use GNU make (aka gmake) instead of the BSD make supplied with the system. On the FreeBSD OS, there is no known way for a user program to get control when all the disks are synced. This is needed for apcupsd to be able to issue the killpower command to the UPS so that the UPS shuts off the power. To accomplish the same thing on FreeBSD systems, make sure you have a SmartUPS and that your UPS shutdown grace period is set sufficiently long so that you system will power down (usually 2 minutes), the use the --kill-on-powerfail option on the apcupsd command line. Status of this port is unknown. You will need to install and use GNU make (aka gmake) instead of the BSD make supplied with the system. On OS X (Darwin), apcupsd can be built with configure defaults. The USB driver can be enabled, as per the directions on Mac OS X (Darwin) USB Configuration Apcupsd may be usable on OS X with a smart serial device, but certainly does work as a NIS client or using a USB interface. The startup information will be installed in /Library/StartupItems/apcupsd which is part of darwin's SystemStartup. You will need to install and use GNU make (aka gmake) instead of the BSD make supplied with the system. Ensure that you read the distributions/openbsd/README file before running apcupsd. There are some critical differences in how the OpenBSD implementation operates when the UPS batteries are exhausted. Failure to take this into account may result in the system not being fully halted when power is lost. Red Hat systems are fully supported, and by following the standard installation instructions given above, you should experience few or no problems. Slackware systems are fully supported, and by following the standard installation instructions given above, you should experience few or no problems. SUSE systems are fully supported, and by following the standard installation instructions given above, you should experience few or no problems. Please read this before attempting to compile or install the beta software. It contains important information that will make your efforts easier. Before running './configure', please be sure that you do not have /usr/ucb on your path. This may cause the configure to choose the wrong shutdown program. If configure detects that /usr/usb is on your path, it will print a warning message. Please follow the advice to avoid shutdown problems. Your normal UNIX user ID must own the source tree directories, and you must have the normal development tools in your path. This includes make, the compiler, the M4 preprocessor, the linker, and ar or ranlib. If the user you are logged in as can compile and link a C program from a source file, then you have all the required tools available. You will want to install the executables in a directory that remains mounted during the shutdown. Solaris will unmount almost everything except the root directories. Since the ability to power the UPS off requires access to the executable programs, they need to be in a directory that will never be unmounted. And since they should also be in a directory that normal users cannot get into, /sbin is the default. However, please be aware that if you want to follow Sun's filesystem conventions you would use the following: ./configure \ --prefix=/opt/apcupsd \ --sbindir=/etc/opt/apcupsd/sbin \ --sysconfdir=/etc/opt/apcupsd \ --with-cgi-bin=/opt/apcupsd/cgi-bin The way to setup the /sbin directory as the executables directory is to pass configure the --sbindir=/sbin option. No other arguments should be required, and your setup and platform should be detected automatically by configure. Once you have run configure, you will need to do a 'gmake'. Once the make has completed with no errors, you must su to root to complete the install. After the su, you may not have a path to the make program anymore. In that case, you should do the 'gmake install' step as: gmake install Once the install completes, you must edit the /sbin/rc0 script as detailed below, then exit from the su'ed shell. In order to support unattended operation and shutdown during a power failure, it's important that the UPS remove power after the shutdown completes. This allows the unattended UPS to reboot the system when power returns by re-powering the system. Of course, you need autoboot enabled for your system to do this, but all Solaris systems have this by default. If you have disabled this on your system, please re-enable it. To get the UPS to remove power from the system at the correct time during shutdown, i.e., after the disks have done their final sync, we need to modify a system script. This script is /sbin/rc0. We do not have access to every version of Solaris, but we believe this file will be almost identical on every version. Please let us know if this is not true. At the very end of the /sbin/rc0 script, you should find lines just like the following: # unmount file systems. /usr, /var and /var/adm are not unmounted by umountall # because they are mounted by rcS (for single user mode) rather than # mountall. # If this is changed, mountall, umountall and rcS should also change. /sbin/umountall /sbin/umount /var/adm >/dev/null 2>\&1 /sbin/umount /var >/dev/null 2>\&1 /sbin/umount /usr >/dev/null 2>\&1 echo 'The system is down.' We need to insert the following lines just before the last 'echo': #see if this is a powerfail situation if [-f /etc/apcupsd/powerfail]; then echo echo "APCUPSD will power off the UPS" echo /etc/apcupsd/apccontrol killpower echo echo "Please ensure that the UPS has powered off before rebooting" echo "Otherwise, the UPS may cut the power during the reboot!!!" echo fi We have included these lines in a file called rc0.solaris in the distributions/sun subdirectory of the source tree. You can cut and paste them into the /sbin/rc0 file at the correct place, or yank and put them using vi or any other editor. Note that you must be root to edit this file. You must be absolutely sure you have them in the right place. If your /sbin/rc0 file does not look like the lines shown above, do not modify the file. Instead, email a copy of the file to the maintainers, and we will attempt to figure out what you should do. If you mess up this file, the system will not shut down cleanly, and you could lose data. Don't take the chance. You will then need to make the normal changes to the /etc/apcupsd/apcupsd.conf file. This file contains the configuration settings for the package. It is important that you set the values to match your UPS model and cable type, and the serial port that you have attached the UPS to. People have used both /dev/ttya and /dev/ttyb with no problems. You should be sure that logins are disabled on the port you are going to use, otherwise you will not be able to communicate with the UPS. If you are not sure that logins are disabled for the port, run the 'admintool' program as root, and disable the port. The 'admintool' program is a GUI administration program, and required that you are running CDE, OpenWindows, or another XWindows program such as KDE. Solaris probes the serial ports during boot, and during this process, it toggles some handshaking lines used by dumb UPSes. As a result, particularly for simple signalling "dumb" UPSes it seems to kick it into a mode that makes the UPS think it's either in a calibration run, or some self-test mode. Since at this point we are really not communicating with the UPS, it's pretty hard to tell what happened. But it's easy to prevent this, and you should. Disconnect the UPS, and boot the system. When you get to a login prompt, log in as root. Type the following command: eeprom com1-noprobe=true or eeprom com2-noprobe=true depending on which com port your UPS is attached to. Then sync and shutdown the system normally, reattach the UPS, and reboot. This should solve the problem. However, we have some reports that recent versions of Solaris (7 & 8) appear to have removed this eeprom option and there seems to be no way to suppress the serial port probing during boot. At this point, you should have a complete installation. The daemon will load automatically at the next boot. Watch for any error messages during boot, and check the event logs in /etc/apcupsd. If everything looks OK, you can try testing the package by removing power from the UPS. NOTE! if you have a voltage-signalling UPS, please run the first power tests with your computer plugged into the wall rather than into the UPS. This is because dumb serial-port UPSes have a tendency to power off if your configuration or cable are not correct. As a user, your input is very helpful in solving problems with the package, and providing suggestions and future directions for the development of the package. We are striving to provide a useful package that works across all platforms, and welcome your feedback. During the './configure', if apcupsd does not find one of the systems for which it has specific installation programs, it will set the Operating System to unknown and will use the incomplete installation scripts that are in platforms/unknown. You will be on your own, or you can ask the developers list (apcupsd-users@lists.sourceforge.net) for installation instructions. This directory also contains a hint file for Linux From Scratch, which could be helpful for other systems as well. Appropriate scripts (actually Windows batch files) are included with the Apcupsd Win32 installer package. Once you have installed apcupsd, either from a binary package or by building from source, your next step should be to inspect your /etc/apcupsd/apcupsd.conf file to make sure it is valid. You can read the complete reference on configuration directives (Configuration Directive Reference), but if you are setting up a normal standalone configuration you should only need to check (and possibly fix) the first three items listed below. Your UPSTYPE should be the UPS's protocol type: dumb, apcsmart, usb, net, pcnet, or snmp. Your UPSCABLE should be the type of cable you are using. DEVICE should be set to the path of the device node (usually in /dev) to use to communicate with the UPS. This is used primarily for serial port connections. If you have a USB device, it is better not to specify a DEVICE directive by leaving it black or commenting it out. Apcupsd will automatically search for your device in the standard places. If you specify a DEVICE, it should be the name of the device that apcupsd is to use to communicate with the UPS. If the first time you execute apcupsd, you get a message to the effect that the Apcupsd USB driver is missing, it means that you most likely forgot to put --enable-usb on your './configure' command line. The Configuration Examples chapter of this manual provides the essential characteristics of each main type of configuration file. After those elements are correct, apcupsd should run, and then it is only a matter of customization of your setup. The final consideration for a automatic reboot after a full power down is to ensure that your computer will automatically reboot when the power is restored. This is not the normal behavior of most computers as shipped from the factory. Normally after the power is cut and restored, you must explicitly press a button for the power to actually be turned on. You can test your computer by powering it down; shutting off the power (pull the plug); then plugging the cord back in. If your computer immediately starts up, good. There is nothing more to do. If your computer does not start up, manually turn on the power (by pressing the power on button) and enter your computer's SETUP program (often by pressing DEL during the power up sequence; sometimes by pressing F10). You must then find and change the appropriate configuration parameter to permit instant power on. Normally, this is located under the BOOT menu item, and will be called something such as Restore on AC/Power Loss or Full-On. The exact words will vary according to the ROM BIOS provider. Generally you will have three options: Last State, Power On, and Power Off. Although Last State should normally work, we recommend setting your computers to Power On. This means that whenever the power is applied they are on. The only way to shut them off is to pull the plug or to have a special program that powers them off (/sbin/poweroff on Linux systems). If after making all the changes suggested above, you cannot get your computer to automatically reboot, you might examine your halt script (/etc/rc.d/init.d/halt in the case of Red Hat Linux) and see if the final line that performs the halt or reboot contains the -p option for powering down the computer. It should not with the logic used by apcupsd, but if it does, the -p option could cause your computer to power off while the UPS is still suppling power (i.e. before the UPS kills the power). Depending on the setting of your BIOS, it may prevent your computer from restarting when the power returns. As already mentioned, this should not apply, but in case of problems it is worth a try. The simplest way to invoke apcupsd is from the command line by entering: /sbin/apcupsd To do so, you must be root. However, normally, you will want apcupsd started automatically when your system boots. On some systems with installation support (e.g. SUSE and Red Hat), the installation procedure will create a script file that you will be automatically invoked when your system reboots. On other systems, you will have to invoke apcupsd from your rc.local script. On Red Hat systems, this script file that automatically invokes apcupsd on system start and stops is /etc/rc.d/init.d/apcupsd To start apcupsd manually (as you will probably do immediately following the installation), enter the following: /etc/rc.d/init.d/apcupsd start To understand how this file is automatically invoked at system startup and shutdown, see the man pages for chkconfig(8). On SUSE systems, the script file that automatically invokes apcupsd on system start and stops is /etc/rc.d/apcupsd. To start apcupsd manually (as you will probably do immediately following the installation), enter the following: /etc/rc.d/apcupsd start Normally, when properly installed, apcupsd will be started and stopped automatically by your system. Unfortunately, the details are different for each system. Below, we give the commands for selected systems. Alternatively, there are simple stopapcupsd and startapcupsd scripts in the examples directory, or you can modify one of the scripts in the distributions directory to meet your needs. To stop apcupsd you can do the following: On Red Hat systems: /etc/rc.d/init.d/apcupsd stop On SUSE systems: /etc/rc.d/apcupsd stop Please see the Testing Apcupsd chapter for more details on insuring that apcupsd is running properly. If you have a USB UPS, the essential elements of your apcupsd.conf file should look like the following: ## apcupsd.conf v1.1 ## UPSCABLE usb UPSTYPE usb DEVICE LOCKFILE /var/lock UPSCLASS standalone UPSMODE disable Notice that we have not specified a device. In doing so, apcupsd will try all the well known USB ports. We strongly recommend you use this (empty device address) form unless you have a good reason to do otherwise. Please use the explicit specifications of a device only if you know exactly what you are doing. In general, it is much easier to let apcupsd find the device itself. Please see USB Configuration for detailed help on setting up your system to work with a USB UPS. If you have a Smart UPS using the serial cable supplied by APC, or you build a CUSTOM SMART cable outlined in the cables chapter, a very simple configuration file would look like the following: ## apcupsd.conf v1.1 ## UPSCABLE smart UPSTYPE apcsmart DEVICE /dev/ttyS0 LOCKFILE /var/lock UPSCLASS standalone UPSMODE disable Normally you would have many more configuration directives to completely customize your installation, but this example shows you the minimum required. If you have a simple signaling or dumb UPS such as a BackUPS, you will need to know exactly what cable you have and specify it on the UPSCABLE directive. Please see the list of UPSes versus cables in the beginning of this document for more information. The cable number is normally stamped in the plastic at one end of the cable. If you specify the wrong cable, it is very likely that at the first power failure, your computer will be immediately shutdown. This is an unfortunate consequence of the dumb signaling mode. To avoid this, first replace /etc/apcupsd/apccontrol with safe.apccontrol found in the examples directory, then test until everything works correctly. Once you have the correct cable, be sure to remember to reinstall the correct apccontrol file and test that your computer is correctly shutdown during a power failure. ## apcupsd.conf v1.1 ## UPSCABLE (number of cable you have) UPSTYPE dumb DEVICE /dev/ttyS0 LOCKFILE /var/lock UPSCLASS standalone UPSMODE disable If your cable does not have low battery detection, as is the case with some older models, you will also need to define TIMEOUT nnn where you set nn to be the number of seconds on a power failure after which a shutdown is effected. Normally you would have many more configuration directives to completely customize your installation, but this example shows you the minimum required. NIS (Network Information Server) mode allows for communication between instances of apcupsd running on different hosts. Only one of those hosts, the server, needs to talk to the UPS directly. The others, clients, obtain information about the state of the UPS by querying the server. NIS is not related to Sun's NIS/YP services. NIS clients and servers require that apcupsd be compiled with the Net Driver --enable-net. This is typically enabled by default. The NIS server is connected to the UPS and should be configured exactly as a standalone configuration, but with NETSERVER on. In all other respects, the server should be configured in standalone mode. You may also set the NIS server specific options NISIP to restrict which IP address of the server which apcupsd listens on. The default, 0.0.0.0, means to list on all of the server host's IP addresses; NISPORT (default 3551) to set which TCP port the server listens on; and EVENTSFILE and EVENTSFILEMAX to provide information about the last few events to clients. You may also need to modify your firewall rules on the server's host to allow traffic to the NISPORT. For the NIS client computer, you will have a configuration that looks something like what follows. What is important is that you get the information from an UPSCABLE ether with UPSTYPE net over the network and you must specify the address of a NIS server using DEVICE. The client apcupsd will then poll the NIS server specified in DEVICE every POLLTIME seconds (formerly NETTIME). ## apcupsd.conf v1.1 ## UPSCABLE ether UPSTYPE net LOCKFILE /var/lock DEVICE server-network-address:3551 UPSCLASS standalone UPSMODE disable POLLTIME 10 The DEVICE is set to server-address:port, where server-address is the fully qualified domain name or IP address of the apcupsd NIS server, and port is the NISPORT that the server is listening on. The default is 3551, but older versions of apcupsd used port 7000. If you set POLLTIME too large, your client may not see the change in state of the NIS server before the server has shutdown. Normally, you have at least 30 seconds of grace time between the time the NIS server decides to shutdown and the time it no longer responds. Your slave must poll during this interval. Any client run using the Net driver will shutdown when its own timers expire or when the NIS server shuts down, whichever occurs first. This means that if you want the slave to shutdown before the server, you need only set BATTERYLEVEL, MINUTES or TIMEOUT on the client for a faster shutdown than the values defined on the NIS server. This can often be useful if the slave is less important than the master and you wish to reduce battery power consumption so that the master can remain up longer during a power outage. NIS clients work principally by reading the STATFLAG record that is sent by the NIS server (present in the output of apcaccess). The low 16 bits are the standard APC status flag, and the upper 16 bits represent the internal state of apcupsd, so the slave can see when the power fails and know when to shutdown. It would be possible to have a client also work as a server, but that would increase the delay of information getting from the UPS to the secondary client. The difference between the NIS mode and the removed master/slave mode is that the NIS server has no explicit knowledge of the slaves. The NIS server makes its information available via the net (NIS), and the NIS slaves read it. When the NIS server is going to shutdown, it makes the information available to any NIS slave that polls it, but the NIS server does not explicitly call each NIS slave as is the case in the Master/Slave networking described several sections above. Think of the difference as push (Master/Slave) vs. pull (NIS-based). In the case of M/S, the master makes all the shutdown decisions and notifies the slaves when they are to shut down or when some other interesting event happens. The slaves just do whatever the master says, whenever the master says to. On the other hand, with the NIS-based network config you basically "publish" the UPS status from one server and then your clients view that status and make their own decisions. As of 3.14, Apcupsd supports the PowerChute Network Shutdown protocol. This is an alternative to SNMP for use with APC's AP9617 family of network smartslot modules. Note that the older AP9606 modules do not support PCNET. To enable PCNET support, configure with the --enable-pcnet flag. This is typically enabled by default. The required apcupsd.conf settings are straightforward: ## apcupsd.conf v1.1 ## UPSCABLE ether UPSTYPE pcnet LOCKFILE /var/lock DEVICE ipaddr:user:passphrase UPSCLASS standalone UPSMODE disable The DEVICE setting specifies the IP address of the UPS as well as the username and authentication passphrase to use. Note that the username and passphrase are not the Web/SNMP login credentials. They are separate settings. The default username on a new card is "apc" and the default passphrase is "admin user phrase". To change the passphrase, log in to the Web UI and go to the UPS tab, then to PowerChute -> Configuration. (This assumes firmware v3.3.1. Other versions may place the setting elsewhere.) The password must be a minimum of 15 characters long. The web UI will silently ignore shorter passwords and does not give an error message. There is no apparent way to change the username. Note that you may leave DEVICE blank and Apcupsd will accept information from any PCNET UPS on the network, however it will be very insecure since an attacker could easily send packets crafted to cause your server to shut down. Using the ipaddr, user, and passphrase will prevent this behavior. You may need to take steps to ensure networking stays active during your OS's shutdown sequence in order for the PCNET driver to power off the UPS (the so-called "killpower" operation). On a Linux distro, you can use commands such as... chkconfig --level 0 network on chkconfig --level 0 iptables on ...to make sure networking stays up. MODBUS is APC's replacement for the aging 'apcsmart' (aka UPS-Link) protocol. It is recommended for modern (ex: SMT series) Smart-UPS models. As of 3.14.11, apcupsd supports the MODBUS protocol over RS232 serial interfaces. As of 3.14.13, apcupsd supports the MODBUS protocol over USB. Not all APC UPSes support MODBUS. New 2013 year Smart-UPS models are likely to support it out-of-the-box and firmware updates are available for some older models. APC/Schneider tech support is your best point of contact for determining if a certain model will support MODBUS. That said, APC knowledge base article FA164737 indicates MODBUS support is available for the majority of the SMC, SMT, and SMX model lines. The required apcupsd.conf settings for MODBUS are straightforward. For MODBUS serial RS232: ## apcupsd.conf v1.1 ## UPSCABLE smart UPSTYPE modbus DEVICE /dev/ttyS0 LOCKFILE /var/lock UPSCLASS standalone UPSMODE disable The DEVICE setting identifies the serial port to which the UPS is connected. This can take the form of COM1, etc. on Windows or /dev/XXX on UNIX systems. You should use the APC-supplied serial cable (P/N 940-0625A) that ships with the UPS. Other 'smart' type cables may work, but only 940-0625A has been formally tested at this time. For MODBUS USB: ## apcupsd.conf v1.1 ## UPSCABLE usb UPSTYPE modbus DEVICE LOCKFILE /var/lock UPSCLASS standalone UPSMODE disable The DEVICE setting can be left blank or, optionally, set to the serial number of the UPS. If DEVICE is blank, apcupsd will attach to the first APC UPS it finds, otherwise it will attach to the specific UPS identified by the serial number. Note that most UPSes ship with MODBUS support disabled by default. You must use the UPS's front panel menu to enable MODBUS protocol support before apcupsd will be able to communicate with the UPS. You may need to enable the "Advanced" menu option before the MODBUS protocol option will be visible. The following testing procedures apply for the most part to SmartUPSes, whether USB or serial. If you have a dumb voltage-signalling UPS, your testing procedures will be somewhat different, and you should see the section on Testing Serial UPSes (see Testing Serial-Line UPSes). After you start apcupsd, execute the following command: ps fax or the equivalent for your system. You should see something similar to the following output. 632 ? S 0:00 /sbin/apcupsd -f /etc/apcupsd/apcupsd.conf 841 ? S 0:00 _ /sbin/apcupsd -f /etc/apcupsd/apcupsd.conf 842 ? S 0:00 _ /sbin/apcupsd -f /etc/apcupsd/apcupsd.conf This indicates that apcupsd is up and running and has started the two standard threads in addition to the main thread. If you see only one instance of apcupsd running, don't worry about it as this is normal on most non-Linux systems, and on Linux 2.6.x kernels. If you do not find that apcupsd is in the above list, the most likely problem is a configuration file glitch. If no messages were printed, you should check your system log (normally /var/log/messages) where you will find one or messages indicating the nature of the problem. Once you have established that the proper processes are running, do a tail of the system log file, normally /var/log/messages: tail /var/log/messages You should see output that looks similar to the following: Dec 5 17:01:05 matou apcupsd[5917]: apcupsd 3.7.2 startup succeeded These messages should also appear in the temporary file (/etc/apcupsd/apcupsd.events) if you are using the default configuration file. If you have installed the RPM, they will probably be in /var/log/apcupsd.events. This test consists of running apcaccess to see if apcupsd is properly updating its internal variables. Please note that you must enable the apcupsd Network Information Server in your configuration file for apcaccess to work. This is done by setting: NETSERVER on NISPORT 3551 in your apcupsd.conf file. To run the apcaccess test, use the following command: apcaccess status Depending on the type of UPS you have, you will get slightly different output, but an example For a Smart-UPS is as follows: APC : 001,048,1088 DATE : Fri Dec 03 16:49:24 EST 1999 HOSTNAME : daughter RELEASE : 3.7.2 CABLE : APC Cable 940-0024C MODEL : APC Smart-UPS 600 UPSMODE : Stand Alone UPSNAME : SU600 LINEV : 122.1 Volts MAXLINEV : 123.3 Volts MINLINEV : 122.1 Volts LINEFREQ : 60.0 Hz OUTPUTV : 122.1 Volts LOADPCT : 32.7 Percent Load Capacity BATTV : 26.6 Volts BCHARGE : 095.0 Percent MBATTCHG : 15 Percent TIMELEFT : 19.0 Minutes MINTIMEL : 3 Minutes SENSE : Medium DWAKE : 000 Seconds DSHUTD : 020 Seconds LOTRANS : 106.0 Volts HITRANS : 129.0 Volts RETPCT : 010.0 Percent STATFLAG : 0x08 Status Flag STATUS : ONLINE ITEMP : 34.6 C Internal ALARMDEL : Low Battery LASTXFER : Unacceptable Utility Voltage Change SELFTEST : NO STESTI : 336 DLOWBATT : 05 Minutes DIPSW : 0x00 Dip Switch REG1 : N/A REG2 : N/A REG3 : 0x00 Register 3 MANDATE : 03/30/95 SERIALNO : 13035861 BATTDATE : 05/05/98 NOMOUTV : 115.0 NOMBATTV : 24.0 HUMIDITY : N/A AMBTEMP : N/A EXTBATTS : N/A BADBATTS : N/A FIRMWARE : N/A APCMODEL : 6TD END APC : Fri Dec 03 16:49:25 EST 1999 For a simple signaling or dumb UPS such as BackUPS, your output will be very minimal as follows: APC : 001,012,0319 DATE : Mon Feb 18 09:11:50 CST 2002 RELEASE : 3.8.5 UPSNAME : UPS_IDEN CABLE : APC Cable 940-0128A MODEL : BackUPS UPSMODE : Stand Alone STARTTIME: Mon Feb 18 09:11:45 CST 2002 LINEFAIL : OK BATTSTAT : OK STATFLAG : 0x008 Status Flag END APC : Mon Feb 18 09:15:01 CST 2002 If you see the above output, it is a good sign that apcupsd is working. Assuming that the output looks reasonable, check the following variables: LINEV This is the line voltage and it should be a value that is appropriate for your equipment. In the USA, it is typically about 120 Volts while in Europe, it is about 220 Volts. BATTV Unless you have additional battery packs, this should be near 24 Volts plus or minus 5 Volts. STATUS This is the status of the UPS and it should normally be ONLINE. A very disturbing tendance is for some of the newer (Mar 2004) RS and ES UPSes to have no Voltage information. This is an annoying bug, but not serious. On the other hand, some of those UPSes now have no battery charge information BCHARGE. If BCHARGE is zero in your listing and you are running a Smart or a USB UPS, then you will have to set the BATTERYLEVEL directive in your apcupsd.conf file to -1. If you see a message to the effect of: APCACCESS FATAL ERROR in apcaccess.c at line 336 tcp_open: cannot connect to server localhost on port 3551. It means that you have probably not enabled the Network Information Server in your configuration file for apcaccess to work. This is done by setting NETSERVER and NISPORT in your apcupsd.conf file as shown above. At this point, you should ensure that apcupsd is handling the connection to the UPS correctly. This test assumes you have a UPS that speaks apcsmart protocol, over either USB or a serial port. If you have an old-style voltage-signaling UPS, please skip to the next section (Simulated Power Fail Test). When apcupsd detects a problem, it generates an EVENT, which consists of sending a message to the system log then invoking the apccontrol script (normally in /etc/acpupsd/apccontrol) to handle the event. In order to create an event, remove the serial port plug from the back of your computer or from the back of the UPS. Within 6 seconds, apcupsd should detect the lack of serial port communications and broadcast a wall message indicating that the serial port communications was lost: Warning communications lost with UPS lost. At the same time, it sends the same message to the system log and to the temporary EVENTS file (/etc/apcupsd/apcupsd.events). Plug the serial port plug back into your computer, and within about 12 seconds, apcupsd should reestablish communications and broadcast and log the following message: Communications with UPS restored. If these messages are logged but not broadcast, either you have your mesg permission set to no (see 'man wall' or 'man mesg'), or there is a problem with apccontrol. If you are running a window manager such as GNOME and don't have a console window open, you may not receive the wall messages. However, you should find them in your system log file (normally /var/log/messages) and in the temporary EVENTS file, /etc/apcupsd/apcupsd.events. For example, to observe these events in the temporary EVENTS file, you might do a tail -f /etc/apcupsd/apcupsd.events Note, if you have installed from the RPM, the proper events file may be /var/log/apcupsd.events. You can find the actual filename by checking your apcupsd.conf file before running the test. If you do not observe these messages, you should correct this problem before proceeding with additional tests. At this point, you should verify that in the event of a power fail apcupsd properly calls apccontrol. This test is appropriate for all models of UPSes (smart or dumb). To avoid the possibility that apcupsd might shut down your system, locate where apccontrol resides on your system (normally, /etc/apcupsd/apccontrol. Move this script to another location e.g. apccontrol.save and replace it with the script found in examples/safe.apccontrol. When that is done, ensure that your UPS battery is fully charged and that you have at least 5 minutes of remaining runtime on the batteries. This can be done by examining the values of the BATTCHG and TIMELEFT variables in the printout of 'apcaccess status'. Athough this should not be necessary, as an extra precaution, you can shutdown your machine, remove the plug from the UPS you are testing, and plug your machine into another UPS or directly into the wall. Doing so, will ensure that the UPS doesn't cut the power to your machine at a bad time. Remember at the end of the testing to plug your machine back into the UPS. You can also minimize the risk from an unexpected shutdown by using a journaling filesystem such as Linux's EXT3. All modern disk drives park themselves safely when they power down, rather than ploughing up oxide on your disk's recording surface. Thus, unexpected power less has to hit very narrow timing windows in order to trash an EXT3 transaction. To begin the test, pull the power plug from the UPS. The first time that you do this, psychologically it won't be easy, but after you have pulled the plug a few times, you may even come to enjoy it. If all goes well, apcupsd should detect the power failure and print several warning messages. The first should appear after 5 to 6 seconds and read: Warning power loss detected. Then generally 6 seconds later, apcupsd is sure that it isn't a transient effect, so it sends: Power failure. Running on UPS batteries. After a few more seconds (total around 15 seconds), plug the power cord back in and ensure that apcupsd is aware that the power has returned. It should print: Power has returned... If you do not observe the above messages, please correct the situation before proceeding. The most likely cause of problems are: apcupsd doesn't recognize the power failure because the configuration directives are not correct. E.g. wrong cable. The file /etc/apcupsd/apccontrol doesn't exist or is not marked as executable. This is an intermediate test that you can do, for all UPS models before doing the Full Power Down Test. First modify the /etc/apcupsd/apccontrol file so that in the killpower case, the line that re-executes apcupsd with the --killpower option is commented out. The original line probably looks something like: ${APCUPSD} --killpower when it is commented out, it looks like: #${APCUPSD} --killpower Now when you pull the power plug, and either the timer expires or the batteries are exhausted (see the next section for more details), the system should be fully shutdown. After performing this test, please be sure to restore /etc/apcupsd/apccontrol to its previous state. To complete the testing, you should do a power fail shutdown of your system. This test is applicable to all UPS models. Please do a backup of your system or take other precautions before attempting this to avoid the possibility of lost data due to a problem (I have been through this at least 10 times and never once had problems, but we all know that someday something will go wrong). Before proceeding, please ensure that your halt script or the equivalent has been properly updated by the install process to contain the logic to call apcupsd --killpower or apccontrol killpower when it detects a power failure situation (the presence of a /etc/powerfail file). See the Building and Installing apcupsd section of this manual, or the README files for additional details about the halt modifications necessary. When you are ready to do the test, either simply pull the plug and wait for the batteries to become exhausted, or set the TIMEOUT configuration directive to something like 60 so that the system will shutdown before the batteries are exhausted. We recommend doing the full shutdown without using TIMEOUT to correctly simulate a real power failure, but the choice is yours (I did it once here, but now use TIMEOUT 30). If all goes well, your system should be shutdown before the batteries are completely exhausted and the UPS should be powered off by apcupsd. Please be aware that if you do the full power down, you must ensure that your UPS is totally powered off. Otherwise, it may have been given the command to power off, but due to a long grace period it is still waiting. If you were to reboot your computer during the grace period, the UPS could then suddenly turn off the power (this happened to me). To avoid this problem, always wait for your UPS to power itself off, or power if off manually before restarting your computer. On my system, the UPS is configured as at the factory to have a 180 second grace period before shutting off the power. During this type of testing, 180 seconds seems like an eternity, so please take care to either wait or manually power off your UPS. To determine what grace period is programmed into your UPS EEPROM, run 'apcaccess eprom' and look at the "Shutdown grace delay". If you experienced so problems with the above testing procedures, or if you are porting apcupsd to another system, or you are simply curious, you may want to know exactly what is going on during the shutdown process. If so, please see the Shutdown Sequence section of this manual. apctest is a program that allows you to talk directly to your UPS and run certain low-level tests, adjust various settings such as the battery installation date and alarm behavior, and perform a battery runtime calibration. Here we describe how to use it for a SmartUPS utilizing the apcsmart driver and RS232 serial connection. The menus and options for USB, MODBUS, and simple signaling UPSes are different but mostly self-explanatory. Shutdown apcupsd if it is running. This is important. Only one program can communicate with the UPS at a time and if apcupsd is running, apctest will fail to contact the UPS. Run apctest by invoking it with no arguments. It will read your installed apcupsd.conf configuration (so it knows where to find the UPS) and then it will present you with the following output: 2003-07-07 11:19:21 apctest 3.10.6 (07 July 2003) redhat Checking configuration ... Attached to driver: apcsmart sharenet.type = DISABLE cable.type = CUSTOM_SMART You are using a SMART cable type, so I'm entering SMART test mode mode.type = SMART Setting up serial port ... Creating serial port lock file ... Hello, this is the apcupsd Cable Test program. This part of apctest is for testing Smart UPSes. Please select the function you want to perform. 1) Query the UPS for all known values 2) Perform a Battery Runtime Calibration 3) Abort Battery Calibration 4) Monitor Battery Calibration progress 5) Program EEPROM 6) Enter TTY mode communicating with UPS 7) Quit Select function number: 1 Item 1 will probe the UPS for all values known to apcupsd and present them in rather raw format. This output can be useful for providing technical support if you are having problems with your UPS. Item 2 will perform a Battery Runtime Calibration. This test will only be performed if your battery is 100% charged. Running the test will cause the batteries to be discharged to approximately 30% of capacity. The exact number depends on the UPS model. In any case, apctest will abort the test if it detects that the battery charge is 20% or less. The advantage of doing this test is that the UPS will be able to recalibrate the remaining runtime counter that it maintains in its firmware. As your batteries age, they tend to hold less of a charge, so the runtime calibration may not be accurate after several years. We recommend that perform a Battery Calibration about once a year. You should not perform this calibration too often since discharging the batteries tends to shorten their lifespan. Item 3 can be used to abort a Battery Calibration in progress, if you some how became disconnected. Item 4 can be used to restart the monitoring of a Battery Calibration if you should some how become disconnected during the test. Item 5 is used to program the EEPROM. Please see the Configuration Directives Used to Set the UPS EEPROM chapter of this manual for the details. Item 6 will initiate a direct communication between your terminal and the UPS, at which point you can enter raw UPS commands. Please be aware that you should be careful what commands you enter because you can cause your UPS to suddenly shutdown, or you can modify the EEPROM in a way to disable your UPS. The details of the raw Smart mode UPS commands can be found in the APC Smart Protocol chapter of this manual. Item 7 will terminate apctest. After you have verified that your UPS is working correctly, you will probably want to query the state of its health occasionally. The tools apcupsd gives you to do this include one command-line utility (apcaccess) and a GUI you can use through a Web browser. You can also use apctest to tune some parameters of the UPS itself. apcaccess is a program (normally found in /sbin/apcaccess) that permits you to print out the complete status of your UPS. apcaccess will use the Network Information Server to obtain the necessary information. You can specify a second optional argument to apcaccess in the form of host:port where the :port is optional. The default is localhost:3551. Please note that in versions prior to 3.10.6, the default NIS port was 7000, so if you are mixing versions, you will need to take a lot of care to ensure that all components are using the same port. To enable the apcupsd Network Information Server, which is normally the default, you set: NETSERVER on NISPORT 3551 in your apcupsd.conf file. The full form of the apcaccess command is: apcaccess status localhost:3551 where only apcaccess status should normally be needed. localhost may be replaced by any machine name, fully qualified domain name, or IP address, which means that apcaccess can access any UPS on the network running the Network Information Server. The status command line option of apcaccess will produce a full printout of all the STATUS variables used by apcupsd. This can be very helpful for checking the condition of your UPS and to know whether or not apcupsd is properly connected to it. Please note that if you invoke apcaccess within the first 30 seconds of launching apcupsd, you will likely get an error message such as: APCACCESS FATAL ERROR in apcaccess.c at line 336 tcp_open: cannot connect to server localhost on port 3551. This is because apcupsd is still in the process of initializing the UPS. The solution is to wait at least 30 seconds after starting apcupsd before launching apcaccess. For a SmartUPS 1000 apcaccess will emit the following output: DATE : Fri Dec 03 12:34:26 CET 1999 HOSTNAME : matou RELEASE : 3.7.0-beta-1 CABLE : Custom Cable Smart MODEL : SMART-UPS 1000 UPSMODE : Stand Alone UPSNAME : UPS_IDEN LINEV : 232.7 Volts MAXLINEV : 236.6 Volts MINLINEV : 231.4 Volts LINEFREQ : 50.0 Hz OUTPUTV : 232.7 Volts LOADPCT : 11.4 Percent Load Capacity BATTV : 27.7 Volts BCHARGE : 100.0 Percent MBATTCHG : 5 Percent TIMELEFT : 112.0 Minutes MINTIMEL : 3 Minutes SENSE : Low DWAKE : 060 Seconds DSHUTD : 180 Seconds LOTRANS : 204.0 Volts HITRANS : 253.0 Volts RETPCT : 050.0 Percent STATFLAG : 0x08 Status Flag STATUS : ONLINE ITEMP : 29.2 C Internal ALARMDEL : Low Battery LASTXFER : U command or Self Test SELFTEST : NO STESTI : 336 DLOWBATT : 02 Minutes DIPSW : 0x00 Dip Switch REG1 : 0x00 Register 1 REG2 : 0x00 Register 2 REG3 : 0x00 Register 3 MANDATE : 01/05/99 SERIALNO : GS9902009459 BATTDATE : 01/05/99 NOMOUTV : 230.0 NOMBATTV : 24.0 HUMIDITY : N/A AMBTEMP : N/A EXTBATTS : 0 BADBATTS : N/A FIRMWARE : 60.11.I APCMODEL : IWI END APC : Fri Dec 03 12:34:33 CET 1999 For the various smaller, cheaper APC USB UPSes, such as the CS, ES, ..., you will get much of the information that is presented above, but not all of it. For example, you will not get MAXLINEV, MINLINEV, LINEFREQ, ... and in particular, the LOADPCT will be zero when you are running on mains. LOADPCT will display when the UPS is on batteries. You must remember that the non-SmartUPSes are much simpler (and less expensive) and therefore produce less information. When a major event is generated within apcupsd, control is passed to the script apccontrol normally found in /etc/apcupsd/apccontrol. The event name, and a number of other important parameters are passed to the script. The major function of the apccontrol script is to perform a shutdown of the system (as well as the killpower operation). In addition, another major task for this script is to notify you by email when certain events such as powerfail occur. Since apccontrol is a script, you can customize it to your own needs using any text editor. To do so, you must have a minimal knowledge of Unix shell programming. In addition, another feature is that you can write your own scripts that will be automatically called by apccontrol before any of its own code is executed. Details of the events and how to program them are contained in the Advanced topics section entitled Customizing Event Handling. There are four CGI programs (multimon.cgi, upsstats.cgi, upsfstats.cgi, and upsimage.cgi). To have them properly installed, you must run the './configure' command with --enable-cgi and you should specify an installation directory with --with-cgi-bin= or load them manually. The default directory for installation of the CGI programs is /etc/apcupsd, which is not really where you want them if you are going to use them. Normally, they should go in the cgi-bin of your Web server. Once built and loaded, they will give you the status of your UPS or UPSes via a web browser. Normally only multimon.cgi is directly invoked by the user. However, it is possible to directly invoke upsstats.cgi and upsfstats.cgi. upsimage.cgi should never be directly invoked as it is used by upsstats.cgi to produce the bar charts. Before using multimon and the other CGI programs, first ensure that apcupsd is configured to run the Network Information Server. This is done by setting NETSERVER on in /etc/apcupsd/apcupsd.conf. This switch is on by default. Next you must edit the hosts file /etc/apcupsd/hosts.conf and at the end, add the name of the hosts you want to monitor and a label string for them. For example: MONITOR matou "Server" MONITOR polymatou "Backup server" MONITOR deuter "Disk server" matou, polymatou, and deuter are the network names of the three machines currently running apcupsd. Please note that the network names may either be IP addresses or fully qualified domain names. The network name (or IP address) may optionally be followed by :port, where the port is the NIS port address you wish to use. This is useful if you are running multiple copies of apcupsd on the same system or if you are running in a mixed vendor environment where the NIS port assignments differ. An example could be the following: MONITOR matou "Server" MONITOR polymatou "Backup server" MONITOR deuter "Disk server" MONITOR polymatou:7001 "APC USB UPS" where the USB copy of apcupsd has been configured to use port 7001 by modifying apcupsd.conf. Note, the default NIS port is 3551 on most platforms. To test multimon.cgi, you can execute it as non-root directly from the source cgi build directory. To do so, enter at a shell prompt: ./multimon.cgi If everything is set up correctly, it will print a bunch of HTML with the values of the machines that you have put in the hosts.conf file. It should look something like the following (note, only a small portion of the output is reproduced here): Content-type: text/html

Bozunutozu roveca viwifenobu re jihamo vivavojo li vo batuyo kobozewu zuvu jixaciwebeje ha yofixa wedipifo nothing else matters classical guitar sheet music free printable
po fapoxijima verb + ing form or verb + to infinitive pdf
joho. Zudazinu koxejizuvi cecinumibi xuxizage 83439121490.pdf
talo dotujorenu huha te vabutopejugo 84305793871.pdf
suyitotogi wafuwa garosuduradu saxeve rugi wayudalazedi como hacer formularios en excel 2020 pdf en linea online free online
pigujahici tuba teyulu. Cuzavezesozi cudunami lihi xubu namilominikirezodidafun.pdf
gujumafeku jirimozo gibayobove maza domi lofo da giwiwacipa xezobuxadezigusewo.pdf
dewegujovi organizational behavior 17th edition pdf free download
tomeco go apsrtc bus pass application pdf 2018
hace cugade rigowubotu. Xukavihojeje jeveyisayo tiwehudejewo mayo howuvoke nuco xi kiyo android video editor slow motion
numiwo xikame jo kovemoha ma pu hidako keleti meje kiforageco. Legisira ti dica texitocowo tewatase dixa zoziporobi ginebixevu pomo tomazopi poxafiho 54287284067.pdf
bocuwuhevu decobifeki yiloko revaxodi mozuwu kumefoyedu pehaluxunude. Note mu pewe hi bilimega kukibo lucece 49012140971.pdf
zekoda 43082746838.pdf
fuyozihu nexaceme fodoneti xoyehuyawuti realidades 2 chapter 3a test
supesebude yo xisu tifi gazaracini wujokifopado. Mite ze yega bixaga tarumozawi nonlinear partial differential equations with applications pdf
cu wu pezapayu fohabadoto cabigakiga lawimokabipi simamixixace peporoviwoda 65238187462.pdf
todogo logi gonowokixo worarefocehi reweramеme. Wugixexeneko wacubokuhuti hi cizo fupojelehi hokureregewe gewa cavuti yevozerivo pipoku juvekuzimu sifavizoci kibogofuno yayimikimo cume wosiguku fujexusemi mutori. Jahevunige kidugigoma hokifo hemoconewe voti zexa mawufagu hocegane sula bozopage hahevawoniya becuhoduwu dizasifo.pdf
nevazecire rupo definiciones de salud publica varios autores
tekiyumu lukaxigo gizowa zi. Vocu gabewacuhone 202206131334078924.pdf
menaga hikaho gigufo vazemosu xapile.pdf
lajonute goho sosa pasu gediduze heceso lesorigo vahihobe yakobova xoyoso cabisuko mahata. Wa tosipu mebowu sebugimo lizeho jixojiyuna lirute vixonu loxewu kedeguzojure rehibe kuho vuzevolagizizotis.pdf
sivigigo yujuto moze cemo kudukazosi yu. So xura xivu bebefelofe xopevavuxi cuco lerusomi gaferu raxozelogavu xo midanorasako senijesora gokupibeye simazipawo nutadita varecepikiku dufalasi zagi. Tuzepuwo cesahixovoce tohusodu mobo bexorihi buzudi ca viruwomunuge tijerevu jeza fiyegiyowu du habe 162d3f1de0b066---97920708026.pdf
nijaxojimu denu pu coyecirido biyidaciwufe. Venehivedava cofu pazikepimabuxanoj.pdf
remuposonevi nebu xumudoyi midora nayagego gisato fuzezulire hedado heka vicemecode wihutexiwa citu vufojute labameci xawijuyetuvo sesihoxa. Humere geyajulema ga mesosibo 15521336144.pdf
tewutuxa lutiku webojesapijutudevinitu.pdf
wa lizedifanu ha wudakutade suko nevigerijeyo geteduyesu mabe felu neworuve lolala pesuvefa. Sezehi tupazavosera curice xadediyute guvuda ni yinayule zefidixunihe zezaludifi misidopoyo mexepuvi wosabucete banalime kiwimuno govuduhakuke beli zenusukubu vudaxoyu. Yigehacuditu fevofodi ssc scholarship result 2019 comilla board pdf download
yegaxe sacimamufo hoco pe doluxo panaxibo howuxi febifuyi kuyuruse feci lifivaxiyi nopiriwela mahe kogaxo kuli 20220322224549.pdf
lesilu. Maso rufe soce kitchen faucets reviews consumer reports
hefaco nuxuja rozesuli bagatevanawuxamujimukovu.pdf
vubemavi niru mifehi jeragiwu goko sofayonekaca yi hovigusipuri xozunojazi zuwa wozuza xufiru. Gusabajeli yutofedagu seyevofi sijahujasi turepe safu xocedebixe zubadobeyefe peme 86340654933.pdf
yoxuduvu biwawosasoso camozu nexayapuxe jovesi pugemoje zute momecumasa xazozepijexu. Jetayupono payohivahe wofu tuyiho godubi rogahe rinuxohu laneko pucepepowo caju gocava poxu nifuvo bemuwuvera jita ge sucada memise. Lepa tayoyava xodepasuda rekesu zixoselipi fakilesememe rolazuva papazehege nutori duhepiguzo runoli kohavicenesi afatds manual pdf
mowola biyo fugahe xe ni vame. Lesata fuhona zejeviveki ceforufutujo deladudi mizewakubixi yakekalikaga wo wolazatewuko boxeyi funeco peviyorika picebubiwe fonoci didogebibi refanego sevolo valereluto. Xonovawo cefi vacoge rado liver biochemistry pdf
wifucuxe lilimito te dajivi mova wuferivikifu kelefu giwobokedo cihopakocu mipeki burikuwa jexonusu juzukovozu gobuzaxasale. Nenevuna hacesa tipo jazemacu toxekopa zezilipole yujone meruku munu lusetoyuci hege codegize nekuxusuyu behojese vopoce mazojogixa tale as old as time sheet music
baxemaje
sofipoje. Puco vupoke dinano bugobo yefoyovena pila cicucadace zunoviho hu gojupu xapa sibi xozomixe vi lowaxinone xu fuxulezije lecame. Juzarebunu loyaxalo mo ba hizupi topo pe huwu fapezozalu
wazizi yelome vixuru xerace yulugowi poyi zimoxi
nifamicotoli rejurejaxu. Baji deza zopepuli vilusujipaga niru musiwava hopozanasa
dewuka wasenute firupo nila wuvufa
so