Frequently Asked Questions for FreeBSD 6.*x*, 7.*x* and 8.*x*

The FreeBSD Documentation Project

Frequently Asked Questions for FreeBSD 6.*x*, 7.*x* and 8.*x*

by The FreeBSD Documentation Project

Published $FreeBSD: doc/en_US.ISO8859-1/books/faq/book.sgml,v 1.1131 2010/10/18 12:43:51 gjb Exp $

This is the FAQ for FreeBSD versions 6.*X*, 7.*X* and 8.*X*. All entries are assumed to be relevant to FreeBSD 6.*X* and later, unless otherwise noted. If you are interested in helping with this project, send email to the FreeBSD documentation project mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-doc). The latest version of this document is always available from the FreeBSD World Wide Web server (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/faq/index.html). It may also be downloaded as one large HTML (book.html) file with HTTP or as plain text, PostScript®, PDF, etc. from the FreeBSD FTP server (ftp://ftp.FreeBSD.org/pub/FreeBSD/doc/). You may also want to Search the FAQ (http://www.FreeBSD.org/search/index.html).

Table of Contents

List of Tables

Chapter 1 Introduction

Welcome to the FreeBSD 6.*X*-, 7.*X*- and 8.*X* FAQ!

As is usual with Usenet FAQs, this document aims to cover the most frequently asked questions concerning the FreeBSD operating system (and of course answer them!). Although originally intended to reduce bandwidth and avoid the same old questions being asked over and over again, FAQs have become recognized as valuable information resources.

Every effort has been made to make this FAQ as informative as possible; if you have any suggestions as to how it may be improved, please feel free to mail them to the FreeBSD documentation project mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-doc).

1. What is FreeBSD?

Briefly, FreeBSD is a UNIX® like operating system for AMD64 and Intel® EM64T, i386™ PC-98, IA-64, ARM®, PowerPC® and UltraSPARC® platforms based on U.C. Berkeley's "4.4BSD-Lite" release, with some "4.4BSD-Lite2" enhancements. It is also based indirectly on William Jolitz's port of U.C. Berkeley's "Net/2" to the i386, known as "386BSD", though very little of the 386BSD code remains. A fuller description of what FreeBSD is and how it can work for you may be found on the FreeBSD home page (http://www.FreeBSD.org/index.html).

FreeBSD is used by companies, Internet Service Providers, researchers, computer professionals, students and home users all over the world in their work, education and recreation.

For more detailed information on FreeBSD, please see the FreeBSD Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/index.html).

2. What is the goal of the FreeBSD Project?

The goal of the FreeBSD Project is to provide software that may be used for any purpose and without strings attached. Many of us have a significant investment in the code (and project) and would certainly not mind a little financial compensation now and then, but we definitely do not insist on it. We believe that our first and foremost "mission" is to provide code to any and all comers, and for whatever purpose, so that the code gets the widest possible use and provides the widest possible benefit. This is, we believe, one of the most fundamental goals of Free Software and one that we enthusiastically support.

That code in our source tree which falls under the GNU General Public License (GPL) (http://www.FreeBSD.org/copyright/COPYING) or GNU Library General Public License (LGPL) (http://www.FreeBSD.org/copyright/COPYING.LIB) comes with slightly more strings attached, though at least on the side of enforced access rather than the usual opposite. Due to the additional complexities that can evolve in the commercial use of GPL software, we do, however, endeavor to replace such software with submissions under the more relaxed FreeBSD license (http://www.FreeBSD.org/copyright/freebsd-license.html) whenever possible.

3. Does the FreeBSD license have any restrictions?

Yes. Those restrictions do not control how you use the code, merely how you treat the FreeBSD Project itself. If you have serious license concerns, read the actual license (http://www.FreeBSD.org/copyright/freebsd-license.html). For the simply curious, the license can be summarized like this.

- Do not claim that you wrote this.

- Do not sue us if it breaks.

4. Can FreeBSD replace my current operating system?

For most people, yes. But this question is not quite that cut-and-dried.

Most people do not actually use an operating system. They use applications. The applications are what really use the operating system. FreeBSD is designed to provide a robust and full-featured environment for applications. It supports a wide variety of web browsers, office suites, email readers, graphics programs, programming environments, network servers, and just about everything else you might want. Most of these applications can be managed through the Ports Collection (http://www.FreeBSD.org/ports/).

If you need to use an application that is only available on one operating system, you simply cannot replace that operating system. Chances are there is a very similar application on FreeBSD, however. If you want a solid office or Internet server, a reliable workstation, or just the ability to do your job without interruptions, FreeBSD will almost certainly do everything you need. Many computer users across the world, including both novices and experienced UNIX administrators, use FreeBSD as their only desktop operating system.

If you are migrating to FreeBSD from some other UNIX environment, you already know most of what you need to. If your background is in graphic-driven operating systems such as Windows® and older versions of Mac OS®, expect to invest additional time learning the UNIX way of doing things. This FAQ and the FreeBSD Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/index.html) are excellent places to start.

5. Why is it called FreeBSD?

- It may be used free of charge, even by commercial users.
- Full source for the operating system is freely available, and the minimum possible restrictions have been placed upon its use, distribution and incorporation into other work (commercial or non-commercial).
- Anyone who has an improvement or bug fix is free to submit their code and have it added to the source tree (subject to one or two obvious provisions).

It is worth pointing out that the word “free” is being used in two ways here, one meaning “at no cost”, the other meaning “you can do whatever you like”. Apart from one or two things you *cannot* do with the FreeBSD code, for example pretending you wrote it, you can really do whatever you like with it.

6. What are the differences between FreeBSD and NetBSD, OpenBSD, and other open source BSD operating systems?

James Howard wrote a good explanation of the history and differences between the various projects, called The BSD Family Tree (http://www.freebsdworld.gr/freebsd/bsd-family-tree.html) which goes a fair way to answering this question.

7. What is the latest version of FreeBSD?

At this point in FreeBSD’s development, there are two parallel development branches; releases are being made from both branches. 7.*X* releases are made from the *7-STABLE* branch and 8.*X* releases from the *8-STABLE* branch.

Up until the release of 8.0, the 7.*X* series was the one known as *-STABLE*. However, as of 8.0, the 7.*X* branch will be designated for an “extended support” status and receive only fixes for major problems, such as security-related fixes.

There will be more releases made from the *7-STABLE* branch, but it is considered a "legacy" branch and most current work will only become a part of *8-STABLE*.

Version 8.1 (ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/8.1-RELEASE/) is the latest release from the *8-STABLE* branch; it was released in Jul 2010. Version 7.3 (ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/7.3-RELEASE/) is the latest release from the *7-STABLE* branch; it was released in March 2010.

Briefly, *-STABLE* is aimed at the ISP, corporate user, or any user who wants stability and a minimal number of changes compared to the new (and possibly unstable) features of the latest *-CURRENT* snapshot. Releases can come from either branch, but *-CURRENT* should only be used if you are prepared for its increased volatility (relative to *-STABLE*, that is).

Releases are made every few months. While many people stay more up-to-date with the FreeBSD sources (see the questions on FreeBSD-CURRENT and FreeBSD-STABLE) than that, doing so is more of a commitment, as the sources are a moving target.

More information on FreeBSD releases can be found on the Release Engineering page (http://www.FreeBSD.org/releng/index.html) on the FreeBSD Web site.

8. What is *FreeBSD-CURRENT*?

FreeBSD-CURRENT (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/current-stable.html#CURRENT) is the development version of the operating system, which will in due course become the new FreeBSD-STABLE branch. As such, it is really only of interest to developers working on the system and die-hard hobbyists. See the relevant section (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/current-stable.html#CURRENT) in the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/index.html) for details on running *-CURRENT*.

If you are not familiar with the operating system or are not capable of identifying the difference between a real problem and a temporary problem, you should not use FreeBSD-CURRENT. This branch sometimes evolves quite quickly and can be un-buildable sometimes. People that use FreeBSD-CURRENT are expected to be able to analyze any problems and only report them if they are deemed to be mistakes rather than "glitches". Questions such as "make world produces some error about groups" on the FreeBSD-CURRENT mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-current) may be treated with contempt.

Every month, snapshot (http://www.FreeBSD.org/snapshots/) releases are made based on the current state of the *-CURRENT* and *-STABLE* branches. The goals behind each snapshot release are:

- To test the latest version of the installation software.
- To give people who would like to run *-CURRENT* or *-STABLE* but who do not have the time or bandwidth to follow it on a day-to-day basis an easy way of bootstrapping it onto their systems.
- To preserve a fixed reference point for the code in question, just in case we break something really badly later. (Although CVS normally prevents anything horrible like this happening.)
- To ensure that all new features and fixes in need of testing have the greatest possible number of potential testers.

No claims are made that any *-CURRENT* snapshot can be considered "production quality" for any purpose. If you want to run a stable and fully tested system, you will have to stick to full releases, or use the *-STABLE* snapshots.

Snapshot releases are directly available from snapshot (http://www.FreeBSD.org/snapshots/).

Official snapshots are generated each month on a regular basis for all actively developed branches. There are also daily snapshot builds of the popular i386 and amd64 branches, hosted on http://snapshots.us.freebsd.org/.

9. What is the *FreeBSD-STABLE* concept?

Back when FreeBSD 2.0.5 was released, FreeBSD development branched in two. One branch was named -STABLE (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/current-stable.html#STABLE), one -CURRENT (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/current-stable.html#CURRENT). *FreeBSD-STABLE* is intended for Internet Service Providers and other commercial enterprises for whom sudden shifts or experimental features are quite undesirable. It receives only well-tested bug fixes and other small incremental enhancements. *FreeBSD-CURRENT*, on the other hand, has been one unbroken line since 2.0 was released, leading towards 8.1-RELEASE and beyond. For more detailed information on branches see "FreeBSD Release Engineering: Creating the Release Branch (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/releng/release-proc.html#REL-BRANCH)", the status of the branches and the upcoming release schedule can be found on the Release Engineering Information (http://www.FreeBSD.org/releng) page.

The 2.2-STABLE branch was retired with the release of 2.2.8. The 3-STABLE branch has ended with the release of 3.5.1, the final 3.*X* release. The 4-STABLE branch has ended with the release of 4.11, the final 4.*X* release. The only changes made to either of these branches will be, for the most part, security-related bug fixes. Support for the 5-STABLE branches has ended with the release of 5.5, the final 5.*X* release. Support for the 6-STABLE branch will continue for some time but focus primarily on security-related bug fixes and other serious issues.

8.1-STABLE is the actively developed -*STABLE* branch. The latest release on the 8.1-STABLE branch is 8.1-RELEASE, which was released in Jul 2010.

The 9-CURRENT branch is the actively developed -*CURRENT* branch toward the next generation of FreeBSD. See What is FreeBSD-CURRENT? for more information on this branch.

10. When are FreeBSD releases made?

The Release Engineering Team <re@FreeBSD.org> releases a new major version of FreeBSD about every 18 months and a new minor version about every 8 months, on average. Release dates are announced well in advance, so that the people working on the system know when their projects need to be finished and tested. A testing period precedes each release, in order to ensure that the addition of new features does not compromise the stability of the release. Many users regard this caution as one of the best things about FreeBSD, even though waiting for all the latest goodies to reach -*STABLE* can be a little frustrating.

More information on the release engineering process (including a schedule of upcoming releases) can be found on the release engineering (http://www.FreeBSD.org/releng/index.html) pages on the FreeBSD Web site.

For people who need or want a little more excitement, binary snapshots are made daily as discussed above.

11. Who is responsible for FreeBSD?

The key decisions concerning the FreeBSD project, such as the overall direction of the project and who is allowed to add code to the source tree, are made by a core team (http://www.FreeBSD.org/administration.html#t-core) of 9 people. There is a much larger team of more than 350 committers

(http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/contributors/article.html#STAFF-COMMITTERS) who are authorized to make changes directly to the FreeBSD source tree.

However, most non-trivial changes are discussed in advance in the mailing lists, and there are no restrictions on who may take part in the discussion.

12. Where can I get FreeBSD?

Every significant release of FreeBSD is available via anonymous FTP from the FreeBSD FTP site (ftp://ftp.FreeBSD.org/pub/FreeBSD/):

- The latest 8-STABLE release, 8.1-RELEASE can be found in the 8.1-RELEASE directory (ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/8.1-RELEASE/).
- Snapshot (http://www.FreeBSD.org/snapshots/) releases are made monthly for the -CURRENT and -STABLE branch, these being of service purely to bleeding-edge testers and developers.
- The latest 7-STABLE release, 7.3-RELEASE can be found in the 7.3-RELEASE directory (ftp://ftp.FreeBSD.org/pub/FreeBSD/releases/i386/7.3-RELEASE/).

Information about obtaining FreeBSD on CD, DVD, and other media can be found in the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/mirrors.html).

13. How do I access the Problem Report database?

The Problem Report database of all user change requests may be queried by using our web-based PR query (http://www.FreeBSD.org/cgi/query-pr.cgi?query) interface.

The send-pr(1) command can be used to submit problem reports and change requests via electronic mail. Alternatively, the web-based problem report submission interface (http://www.freebsd.org/send-pr.html) can be used to submit problem reports through a web browser.

Before submitting a problem report, please read Writing FreeBSD Problem Reports (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/problem-reports/article.html), an article on how to write good problem reports.

14. What other sources of information are there?

Please check the Documentation (http://www.FreeBSD.org/docs.html) list on the main FreeBSD (http://www.FreeBSD.org) web site.

Chapter 2 Documentation and Support

1. What good books are there about FreeBSD?

The project produces a wide range of documentation, available online from this link: http://www.FreeBSD.org/docs.html. In addition, the Bibliography at the end of this FAQ, and the one in the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/bibliography.html) reference other recommended books.

2. Is the documentation available in other formats, such as plain text (ASCII), or PostScript?

Yes. The documentation is available in a number of different formats and compression schemes on the FreeBSD FTP site, in the /pub/FreeBSD/doc/ (ftp://ftp.FreeBSD.org/pub/FreeBSD/doc/) directory.

The documentation is categorized in a number of different ways. These include:

- The document's name, such as `faq`, or `handbook`.
- The document's language and encoding. These are based on the locale names you will find under `/usr/share/locale` on your FreeBSD system. The current languages and encodings that we have for documentation are as follows:

Name	Meaning
`en_US.ISO8859-1`	English (United States)
`bn_BD.ISO10646-1`	Bengali or Bangla (Bangladesh)
`da_DK.ISO8859-1`	Danish (Denmark)
`de_DE.ISO8859-1`	German (Germany)
`el_GR.ISO8859-7`	Greek (Greece)
`es_ES.ISO8859-1`	Spanish (Spain)
`fr_FR.ISO8859-1`	French (France)
`hu_HU.ISO8859-2`	Hungarian (Hungary)
`it_IT.ISO8859-15`	Italian (Italy)
`ja_JP.eucJP`	Japanese (Japan, EUC encoding)
`mn_MN.UTF-8`	Mongolian (Mongolia, UTF-8 encoding)
`nl_NL.ISO8859-1`	Dutch (Netherlands)
`no_NO.ISO8859-1`	Norwegian (Norway)
`pl_PL.ISO8859-2`	Polish (Poland)
`pt_BR.ISO8859-1`	Portuguese (Brazil)
`ru_RU.KOI8-R`	Russian (Russia, KOI8-R encoding)
`sr_YU.ISO8859-2`	Serbian (Serbia)
`tr_TR.ISO8859-9`	Turkish (Turkey)
`zh_CN.GB2312`	Simplified Chinese (China, GB2312 encoding)
`zh_TW.Big5`	Traditional Chinese (Taiwan, Big5 encoding)

Note: Some documents may not be available in all languages.

- The document's format. We produce the documentation in a number of different output formats. Each format has its own advantages and disadvantages. Some formats are better suited for online reading, while others are meant to be aesthetically pleasing when printed on paper. Having the documentation available in any of these formats ensures that our readers will be able to read the parts they are interested in, either on their monitor, or on paper after printing the documents. The currently available formats are:

Format	Meaning
`html-split`	A collection of small, linked, HTML files.
`html`	One large HTML file containing the entire document
`pdf`	Adobe's Portable Document Format
`ps`	PostScript
`rtf`	Microsoft's Rich Text Format
`txt`	Plain text

Note: Page numbers are not automatically updated when loading Rich Text Format into Word. Press **Ctrl**+**A**, **Ctrl**+**End**, **F9** after loading the document, to update the page numbers.

- The compression and packaging scheme. There are three of these currently in use.

  1. Where the format is `html-split`, the files are bundled up using tar(1). The resulting `.tar` file is then compressed using the compression schemes detailed in the next point.
  2. All the other formats generate one file, called *`type`*`.`*`format`* (i.e., `article.pdf`, `book.html`, and so on).

     These files are then compressed using two compression schemes.

Scheme	Description
`zip`	The `zip` format. If you want to uncompress this on FreeBSD you will need to install the `archivers/unzip` port first.
`bz2`	The `bzip2` format. Less widespread than `zip`, but generally gives smaller files. Install the `archivers/bzip2` port to uncompress these files.

     So the PostScript version of the Handbook, compressed using `bzip2` will be stored in a file called `book.ps.bz2` in the `handbook/` directory.

After choosing the format and compression mechanism that you want to download, you will have to download the compressed files yourself, uncompress them, and then copy the appropriate documents into place.

For example, the split HTML version of the FAQ, compressed using bzip2(1), can be found in the `doc/en_US.ISO8859-1/books/faq/book.html-split.tar.bz2` file. To download and uncompress that file you would have to do this.

```
# fetch ftp://ftp.FreeBSD.org/pub/FreeBSD/doc/en_US.ISO8859-1/books/faq/book.html-split.tar.bz2
# bzip2 -d book.html-split.tar.bz2
# tar xvf book.html-split.tar
```

You will be left with a collection of `.html` files. The main one is called `index.html`, which will contain the table of contents, introductory material, and links to the other parts of the document. You can then copy or move these to their final location as necessary.

3. Where do I find info on the FreeBSD mailing lists?

You can find full information in the Handbook entry on mailing-lists (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/eresources.html#ERESOURCES-MAIL).

4. What FreeBSD news groups are available?

You can find full information in the Handbook entry on newsgroups (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/eresources-news.html).

5. Are there FreeBSD IRC (Internet Relay Chat) channels?

Yes, most major IRC networks host a FreeBSD chat channel:

- Channel `#FreeBSD` on EFNet (http://www.efnet.org/index.php) is a FreeBSD forum, but do not go there for tech support or try to get folks there to help you avoid the pain of reading manual pages or doing your own research. It is a chat channel, first and foremost, and topics there are just as likely to involve sex, sports or nuclear weapons as they are FreeBSD. You Have Been Warned! Available at server `irc.efnet.org`.
- Channel `#FreeBSDhelp` on EFNet (http://www.efnet.org/index.php) is a channel dedicated to helping FreeBSD users. They are much more sympathetic to questions than `#FreeBSD` is.
- Channel `##FreeBSD` on Freenode (http://freenode.net/) is a general help channel with many users at any time. The conversations have been known to run off-topic for a while, but priority is given to users with FreeBSD questions. We are good about helping you understand the basics, referring to the Handbook whenever possible, and directing you where to learn more about the topic you need help with. We are a primarily English speaking channel, though we have users from all over the world. If you would like to speak in your native language, try to ask the question in English and then relocate to another channel `##freebsd-`*`lang`* as appropriate.
- Channel `#FreeBSD` on DALNET (http://www.dal.net/) is available at `irc.dal.net` in the US and `irc.eu.dal.net` in Europe.
- Channel `#FreeBSDHelp` on DALNET (http://www.dal.net/) is available at `irc.dal.net` in the US and `irc.eu.dal.net` in Europe.
- Channel `#FreeBSD` on UNDERNET (http://www.undernet.org/) is available at `us.undernet.org` in the US and `eu.undernet.org` in Europe. Since it is a help channel, be prepared to read the documents you are referred to.
- Channel `#FreeBSD` on RUSNET (http://www.rusnet.org.ru/) is a russian-language oriented channel dedicated to helping FreeBSD users. This is also good place for non-technical discussions.
- Channel `#bsdchat` on Freenode (http://freenode.net/) is a Traditional-Chinese (UTF-8 encoding) language oriented channel dedicated to helping FreeBSD users. This is also good place for non-technical discussions.

Each of these channels are distinct and are not connected to each other. Their chat styles also differ, so you may need to try each to find one suited to your chat style. As with *all* types of IRC traffic, if you are easily offended or cannot

deal with lots of young people (and more than a few older ones) doing the verbal equivalent of jello wrestling, do not even bother with it.

6. Where can I get commercial FreeBSD training and support?

The FreeBSD Mall provides commercial FreeBSD support. You can get more information at their web site (http://www.freebsdmall.com/cgi-bin/fm).

BSD Certification Group, Inc. provides system administration certifications for DragonFly BSD, FreeBSD, NetBSD, OpenBSD. If you are interested in them, visit their site (http://www.BSDCertification.org).

Any other organizations providing training and support should contact the Project in order to be listed here.

Chapter 3 Installation

1. Which file do I download to get FreeBSD?

You need three floppy images: `floppies/boot.flp`, `floppies/kern1.flp`, and `floppies/kern2.flp`. These images need to be copied onto floppies by tools like `fdimage` or dd(1).

If you need to download the distributions yourself (for a DOS file system install, for instance), below are some recommendations for distributions to grab:

- base/
- manpages/
- compat*/
- doc/
- src/ssys.*

Full instructions on this procedure and a little bit more about installation issues in general can be found in the Handbook entry on installing FreeBSD (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/install.html).

2. What do I do if the floppy images does not fit on a single floppy?

A 3.5 inch (1.44 MB) floppy can accommodate 1,474,560 bytes of data. The boot image is exactly 1,474,560 bytes in size.

Common mistakes when preparing the boot floppy are:

- Not downloading the floppy image in *binary* mode when using FTP.

  Some FTP clients default their transfer mode to *ascii* and attempt to change any end-of-line characters received to match the conventions used by the client's system. This will almost invariably corrupt the boot image. Check the size of the downloaded boot image: if it is not *exactly* that on the server, then the download process is suspect.

  To workaround: type *binary* at the FTP command prompt after getting connected to the server and before starting the download of the image.

- Using the DOS `copy` command (or equivalent GUI tool) to transfer the boot image to floppy.

  Programs like `copy` will not work as the boot image has been created to be booted into directly. The image has the complete content of the floppy, track for track, and is not meant to be placed on the floppy as a regular file. You have to transfer it to the floppy "raw", using the low-level tools (e.g. `fdimage` or `rawrite`) described in the installation guide to FreeBSD (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/install.html).

3. Where are the instructions for installing FreeBSD?

Installation instructions can be found in the Handbook entry on installing FreeBSD (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/install.html).

4. What do I need in order to run FreeBSD?

For FreeBSD you will need a 486 or better PC, with 24 MB or more of RAM and at least 150 MB of hard disk space.

All versions of FreeBSD can run with a low end MDA graphics card but to run Xorg, a VGA or better video card is needed.

See also Chapter 4.

5. How can I make my own custom install floppy?

Currently there is no way to *just* make a custom install floppy. You have to cut a whole new release, which will include your install floppy.

To make a custom release, follow the instructions in the Release Engineering (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/releng/article.html) article.

6. Can I have more than one operating system on my PC?

Have a look at the multi-OS page (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/multi-os/index.html).

7. Can Windows co-exist with FreeBSD?

Install Windows first, then FreeBSD. FreeBSD's boot manager will then manage to boot Windows and FreeBSD. If you install Windows second, it will boorishly overwrite your boot manager without even asking. If that happens, see the next section.

8. Windows killed my boot manager! How do I get it back?

You can reinstall the boot manager FreeBSD comes with in one of three ways:

- Running DOS, go into the `tools` directory of your FreeBSD distribution and look for `bootinst.exe`. You run it like so:

  ```
  ...\TOOLS> bootinst.exe boot.bin
  ```

  and the boot manager will be reinstalled.

- Boot the FreeBSD boot floppy again and go to the Custom menu item for custom installation. Choose Partition. Select the drive which used to contain your boot manager (likely the first one) and when you come to the partition editor for it, as the very first thing (e.g. do not make any changes) press **W**. This will ask for confirmation, select [Yes], and when you get the Boot Manager selection prompt, be sure to select the **FreeBSD Boot Manager**. This will re-write the boot manager to disk. Now quit out of the installation menu and reboot off the hard disk as normal.

- Boot the FreeBSD boot floppy (or CD-ROM) and choose the Fixit menu item. Select either the Fixit floppy or CD-ROM #2 (the "live" file system option) as appropriate and enter the fixit shell. Then execute the following command:

  ```
  Fixit# fdisk -B -b /boot/boot0 bootdevice
  ```

  substituting *`bootdevice`* for your real boot device such as `ad0` (first IDE disk), `ad4` (first IDE disk on auxiliary controller), `da0` (first SCSI disk), etc.

9. My A, T, or X series IBM Thinkpad locks up when I first booted up my FreeBSD installation. How can I solve this?

A bug in early revisions of IBM's BIOS on these machines mistakenly identifies the FreeBSD partition as a potential FAT suspend-to-disk partition. When the BIOS tries to parse the FreeBSD partition it hangs.

According to IBM[1], the following model/BIOS release numbers incorporate the fix.

Model	BIOS revision
T20	IYET49WW or later
T21	KZET22WW or later
A20p	IVET62WW or later
A20m	IWET54WW or later
A21p	KYET27WW or later
A21m	KXET24WW or later
A21e	KUET30WW

It has been reported that later IBM BIOS revisions may have reintroduced the bug. This message (http://docs.FreeBSD.org/cgi/mid.cgi?20010427133759.A71732) from Jacques Vidrine <`nectar@FreeBSD.org`> to the FreeBSD laptop computer mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-mobile) describes a procedure which may work if your newer IBM laptop does not boot FreeBSD properly, and you can upgrade or downgrade the BIOS.

If you have an earlier BIOS, and upgrading is not an option, a workaround is to install FreeBSD, change the partition ID FreeBSD uses, and install new boot blocks that can handle the different partition ID.

First, you will need to restore the machine to a state where it can get through its self-test screen. Doing this requires powering up the machine without letting it find a FreeBSD partition on its primary disk. One way is to remove the hard disk and temporarily move it to an older ThinkPad (such as a ThinkPad 600) or a desktop PC with an appropriate conversion cable. Once it is there, you can delete the FreeBSD partition and move the hard disk back. The ThinkPad should now be in a bootable state again.

With the machine functional again, you can use the workaround procedure described here to get a working FreeBSD installation.

1. Download `boot1` and `boot2` from http://people.FreeBSD.org/~bmah/ThinkPad/. Put these files somewhere you will be able to retrieve them later.
2. Install FreeBSD as normal on to the ThinkPad. *Do not* use `Dangerously Dedicated` mode. *Do not* reboot when the install has finished.
3. Either switch to the "Emergency Holographic Shell" (**Alt+F4**) or start a "fixit" shell.
4. Use fdisk(8) to change the FreeBSD partition ID from `165` to `166` (this is the type used by OpenBSD).
5. Bring the `boot1` and `boot2` files to the local file system.
6. Use disklabel(8) to write `boot1` and `boot2` to your FreeBSD slice.

   ```
   # disklabel -B -b boot1 -s boot2 ad0sn
   ```

   n is the number of the slice where you installed FreeBSD.
7. Reboot. At the boot prompt you will be given the option of booting `OpenBSD`. This will actually boot FreeBSD.

Getting this to work in the case where you want to dual boot OpenBSD and FreeBSD on the same laptop is left as an exercise for the reader.

10. Can I install on a disk with bad blocks?

You can, but it is a bad idea.

If you are seeing bad block errors with a modern IDE drive, chances are the drive is going to die very soon (the drive's internal remapping functions are no longer sufficient to fix the bad blocks, which means the disk is heavily corrupted); we suggest you buy a new hard drive.

If you have a SCSI drive with bad blocks, see this answer.

11. Strange things happen when I boot the install floppy! What is happening?

If you are seeing things like the machine grinding to a halt or spontaneously rebooting when you try to boot the install floppy, here are three questions to ask yourself:

1. Did you use a new, freshly-formatted, error-free floppy (preferably a brand-new one straight out of the box, as opposed to the magazine cover disk that has been lying under the bed for the last three years)?
2. Did you download the floppy image in binary (or image) mode? (do not be embarrassed, even the best of us have accidentally downloaded a binary file in ASCII mode at least once!)
3. If you are using Windows 95 or Windows 98 did you run `fdimage` or `rawrite` in pure DOS mode? These operating systems can interfere with programs that write directly to hardware, which the disk creation program does; even running it inside a DOS shell in the GUI can cause this problem.

There have also been reports of Netscape® causing problems when downloading the boot floppy, so it is probably best to use a different FTP client if you can.

12. I booted from my ATAPI CD-ROM, but the install program says no CD-ROM is found. Where did it go?

The usual cause of this problem is a mis-configured CD-ROM drive. Many PCs now ship with the CD-ROM as the slave device on the secondary IDE controller, with no master device on that controller. This is illegal according to the ATAPI specification, but Windows plays fast and loose with the specification, and the BIOS ignores it when booting. This is why the BIOS was able to see the CD-ROM to boot from it, but why FreeBSD cannot see it to complete the install.

Reconfigure your system so that the CD-ROM is either the master device on the IDE controller it is attached to, or make sure that it is the slave on an IDE controller that also has a master device.

13. Can I install on my laptop over PLIP (Parallel Line IP)?

Yes. Use a standard Laplink cable. If necessary, you can check out the PLIP section of the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-plip.html) for details on parallel port networking.

14. Which geometry should I use for a disk drive?

Note: By the "geometry" of a disk, we mean the number of cylinders, heads and sectors/track on a disk. We will refer to this as C/H/S for convenience. This is how the PC's BIOS works out which area on a disk to read/write from.

This causes a lot of confusion among new system administrators. First of all, the *physical* geometry of a SCSI drive is totally irrelevant, as FreeBSD works in term of disk blocks. In fact, there is no such thing as "the" physical geometry, as the sector density varies across the disk. What manufacturers claim is the "physical geometry" is usually the geometry that they have determined wastes the least space. For IDE disks, FreeBSD does work in terms of C/H/S, but all modern drives internally convert this into block references.

All that matters is the *logical* geometry. This is the answer that the BIOS gets when it asks the drive "what is your geometry?" It then uses this geometry to access the disk. As FreeBSD uses the BIOS when booting, it is very important to get this right. In particular, if you have more than one operating system on a disk, they must all agree on the geometry. Otherwise you will have serious problems booting!

For SCSI disks, the geometry to use depends on whether extended translation support is turned on in your controller (this is often referred to as "support for DOS disks >1GB" or something similar). If it is turned off, then use *N* cylinders, 64 heads and 32 sectors/track, where *N* is the capacity of the disk in MB. For example, a 2 GB disk should pretend to have 2048 cylinders, 64 heads and 32 sectors/track.

If it *is* turned on (it is often supplied this way to get around certain limitations in MS-DOS®) and the disk capacity is more than 1 GB, use *M* cylinders, 63 sectors per track (*not* 64), and 255 heads, where *M* is the disk capacity in MB divided by 7.844238 (!). So our example 2 GB drive would have 261 cylinders, 63 sectors per track and 255 heads.

If you are not sure about this, or FreeBSD fails to detect the geometry correctly during installation, the simplest way around this is usually to create a small DOS partition on the disk. The BIOS should then detect the correct geometry, and you can always remove the DOS partition in the partition editor if you do not want to keep it. You might want to leave it around for programming network cards and the like, however.

Alternatively, there is a freely available utility distributed with FreeBSD called `pfdisk.exe`. You can find it in the `tools` subdirectory on the FreeBSD CD-ROM or on the various FreeBSD FTP sites. This program can be used to work out what geometry the other operating systems on the disk are using. You can then enter this geometry in the partition editor.

15. Are there any restrictions on how I divide the disk up?

Yes. You must make sure that your root partition is below 1024 cylinders so the BIOS can boot the kernel from it. (Note that this is a limitation in the PC's BIOS, not FreeBSD).

For a SCSI drive, this will normally imply that the root partition will be in the first 1024 MB (or in the first 4096 MB if extended translation is turned on — see previous question). For IDE, the corresponding figure is 504 MB.

16. Is FreeBSD compatible with any disk managers?

FreeBSD recognizes the **Ontrack Disk Manager** and makes allowances for it. Other disk managers are not supported.

If you just want to use the disk with FreeBSD you do not need a disk manager. Just configure the disk for as much space as the BIOS can deal with (usually 504 megabytes), and FreeBSD should figure out how much space you

really have. If you are using an old disk with an MFM controller, you may need to explicitly tell FreeBSD how many cylinders to use.

If you want to use the disk with FreeBSD and another operating system, you may be able to do without a disk manager: just make sure the FreeBSD boot partition and the slice for the other operating system are in the first 1024 cylinders. If you are reasonably careful, a 20 megabyte boot partition should be plenty.

17. When I boot FreeBSD for the first time after install I get "`Missing Operating System`". What is happening?

This is classically a case of FreeBSD and DOS or some other OS conflicting over their ideas of disk geometry. You will have to reinstall FreeBSD, but obeying the instructions given above will almost always get you going.

18. Why can I not get past the boot manager's `F?` prompt?

This is another symptom of the problem described in the preceding question. Your BIOS geometry and FreeBSD geometry settings do not agree! If your controller or BIOS supports cylinder translation (often marked as ">1GB drive support"), try toggling its setting and reinstalling FreeBSD.

19. Do I need to install the complete sources?

In general, no. However, we would strongly recommend that you install, at a minimum, the `base` source kit, which includes several of the files mentioned here, and the `sys` (kernel) source kit, which includes sources for the kernel. There is nothing in the system which requires the presence of the sources to operate, however, except for the kernel-configuration program config(8). With the exception of the kernel sources, our build structure is set up so that you can read-only mount the sources from elsewhere via NFS and still be able to make new binaries (due to the kernel-source restriction, we recommend that you not mount this on `/usr/src` directly, but rather in some other location with appropriate symbolic links to duplicate the top-level structure of the source tree).

Having the sources on-line and knowing how to build a system with them will make it much easier for you to upgrade to future releases of FreeBSD.

To actually select a subset of the sources, use the Custom menu item when you are in the Distributions menu of the system installation tool.

20. Do I need to build a kernel?

Building a new kernel was originally pretty much a required step in a FreeBSD installation, but more recent releases have benefited from the introduction of much friendlier kernel configuration methods. It is very easy to configure the kernel's configuration by much more flexible "hints" which can be set at the loader prompt.

It may still be worthwhile building a new kernel containing just the drivers that you need, just to save a bit of RAM, but it is no longer necessary for most systems.

21. Should I use DES, Blowfish, or MD5 passwords and how do I specify which form my users receive?

The default password format on FreeBSD is to use *MD5*-based passwords. These are believed to be more secure than the traditional UNIX password format, which used a scheme based on the *DES* algorithm. DES passwords are still available if you need to share your password file with legacy operating systems which still use the less secure password format. FreeBSD also allows you to use the Blowfish password format, which is more secure. Which password format to use for new passwords is controlled by the `passwd_format` login capability in

`/etc/login.conf`, which takes values of `des`, `blf` (if these are available) or `md5`. See the login.conf(5) manual page for more information about login capabilities.

22. Why does the boot floppy start, but hang at the `Probing Devices...` screen?

If you have a IDE Zip® or Jaz® drive installed, remove it and try again. The boot floppy can get confused by the drives. After the system is installed you can reconnect the drive. Hopefully this will be fixed in a later release.

23. Why do I get a "`panic: can't mount root`" error when rebooting the system after installation?

This error comes from confusion between the boot block's and the kernel's understanding of the disk devices. The error usually manifests on two-disk IDE systems, with the hard disks arranged as the master or single device on separate IDE controllers, with FreeBSD installed on the secondary IDE controller. The boot blocks think the system is installed on `ad0` (the second BIOS disk) while the kernel assigns the first disk on the secondary controller device, `ad2`. After the device probing, the kernel tries to mount what the boot blocks think is the boot disk, `ad0`, while it is really `ad2`, and fails.

To fix the problem, do one of the following:

1. Reboot the system and hit **Enter** at the `Booting kernel in 10 seconds; hit [Enter] to interrupt` prompt. This will drop you into the boot loader.

   Then type **`set root_disk_unit="`*`disk_number`*`"`**. *`disk_number`* will be `0` if FreeBSD is installed on the master drive on the first IDE controller, `1` if it is installed on the slave on the first IDE controller, `2` if it is installed on the master of the second IDE controller, and `3` if it is installed on the slave of the second IDE controller.

   Then type **`boot`**, and your system should boot correctly.

   To make this change permanent (i.e, so you do not have to do this every time you reboot or turn on your FreeBSD machine), put the line **`root_disk_unit="`*`disk_number`*`"`** in `/boot/loader.conf.local`.
2. Move the FreeBSD disk onto the primary IDE controller, so the hard disks are consecutive.

24. What are the limits for memory?

Memory limits depend on the platform used. On a standard i386 install, the limit is 4 GB but more memory can be supported through pae(4). See instructions for using 4 GB or more memory on i386.

FreeBSD/pc98 has a limit of 4 GB memory, and PAE can not be used with it. Other architectures supported by FreeBSD have much higher theoretical limits on maximum memory (many terabytes).

25. What are the limits for FFS file systems?

For FFS file systems, the maximum theoretical limit is 8 TB (2 G blocks), or 16 TB for the default block size of 8 KB. In practice, there is a soft limit of 1 TB, but with modifications file systems with 4 TB are possible (and exist).

The maximum size of a single FFS file is approximately 1 G blocks, or 4 TB with a block size of 4 KB.

FS Block Size	Works	Should Work

Table 3-1. Maximum File Sizes

FS Block Size	Works	Should Work
4 KB	> 4 GB	4 TB - 1
8 KB	> 32 GB	32 TB - 1
16 KB	> 128 GB	32 TB - 1
32 KB	> 512 GB	64 TB - 1
64 KB	> 2048 GB	128 TB - 1

When the FS block size is 4 KB, triple indirect blocks work and everything should be limited by the maximum FS block number that can be represented using triple indirect blocks (approx. $1024^3 + 1024^2 + 1024$), but everything is limited by a (wrong) limit of 1 G - 1 on FS block numbers. The limit on FS block numbers should be 2 G - 1. There are some bugs for FS block numbers near 2 G - 1, but such block numbers are unreachable when the FS block size is 4 KB.

For block sizes of 8 KB and larger, everything should be limited by the 2 G - 1 limit on FS block numbers, but is actually limited by the 1 G - 1 limit on FS block numbers. Using the correct limit of 2 G - 1 blocks does cause problems.

26. Why do I get an error message, "`archsw.readin.failed`" after compiling and booting a new kernel?

Because your world and kernel are out of sync. This is not supported. Be sure you use `make buildworld` and `make buildkernel` to update your kernel.

You can boot by specifying the kernel directly at the second stage, pressing any key when the `|` shows up before loader is started.

27. Installation crashes while booting, what can I do?

Try disabling ACPI support. When the bootloader loads, press the **Space** key. The system will display the following:

```
OK
```

Type:

```
unset acpi_load
```

And then type:

```
boot
```

Notes

1. In an email from Keith Frechette <`kfrechet@us.ibm.com`>.

Chapter 4 Hardware Compatibility

4.1 General

1. I want to get a piece of hardware for my FreeBSD system. Which model/brand/type is best?

This is discussed continually on the FreeBSD mailing lists. Since hardware changes so quickly, however, we expect this. We *still* strongly recommend that you read through the Hardware Notes for FreeBSD 8.1 (http://www.FreeBSD.org/releases/8.1R/hardware.html) or 7.3 (http://www.FreeBSD.org/releases/7.3R/hardware.html) and search the mailing list archives (http://www.FreeBSD.org/search/#mailinglists) before asking about the latest and greatest hardware. Chances are a discussion about the type of hardware you are looking for took place just last week.

If you are looking for a laptop, check the FreeBSD laptop computer mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-mobile) archives. Otherwise, you probably want the archives for the FreeBSD general questions mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-questions), or possibly a specific mailing list for a particular hardware type.

4.2 Memory

1. Does FreeBSD support more than 4 GB of memory (RAM)? More than 16 GB? More than 48 GB?

Yes. FreeBSD as an operating system generally supports as much physical memory (RAM) as the platform it is running on does. Keep in mind that different platforms have different limits for memory; for example i386 without PAE supports at most 4 GB of memory (and usually less than that because of PCI address space) and i386 with PAE supports at most 64 GB memory. AMD64 platforms currently deployed support up to 1 TB of physical memory.

2. Why does FreeBSD report less than 4 GB memory when installed on an i386 machine?

The total address space on i386 machines is 32-bit, meaning that at most 4 GB of memory is addressable (can be accessed). Furthermore, some addresses in this range are reserved by hardware for different purposes, for example for using and controlling PCI devices, for accessing video memory, and so on. Therefore, the total amount of memory usable by the operating system for its kernel and applications is limited to significantly less than 4 GB. Usually, 3.2 GB to 3.7 GB is the maximum usable physical memory in this configuration.

To access more than 3.2 GB to 3.7 GB of installed memory (meaning up to 4 GB but also more than 4 GB), a special tweak called PAE must be used. PAE stands for Physical Address Extension and is a way for 32-bit x86 CPUs to address more than 4 GB of memory. It remaps the memory that would otherwise be overlaid by address reservations for hardware devices above the 4 GB range and uses it as additional physical memory (see pae(4)). Using PAE has some drawbacks; this mode of memory access is a little bit slower than the normal (without PAE) mode and loadable modules (see kld(4)) are not supported. This means all drivers must be compiled into the kernel.

The most common way to enable PAE is to build a new kernel with the special ready-provided kernel configuration file called `PAE`, which is already configured to build a safe kernel. Note that some entries in this kernel configuration file are too conservative and some drivers marked as unready to be used with PAE are actually usable. A rule of

thumb is that if the driver is usable on 64-bit architectures (like AMD64), it is also usable with PAE. If you wish to create your own kernel configuration file, you can enable PAE by adding the following line to your configuration:

```
options         PAE
```

PAE is not much used nowadays because most new x86 hardware also supports running in 64-bit mode, known as AMD64 or Intel 64. It has a much larger address space and does not need such tweaks. FreeBSD supports AMD64 and it is recommended that this version of FreeBSD be used instead of the i386 version if 4 GB or more memory is required.

4.3 Architectures and Processors

1. Does FreeBSD support architectures other than the x86?

Yes. FreeBSD currently runs on the Intel x86 and the AMD64 architectures. The Intel EM64T, IA-64, ARM, PowerPC, sun4v and SPARC64® architectures are also supported. Upcoming platforms are MIPS® and S/390®, join the FreeBSD MIPS porting mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-mips) for more information about ongoing work on the MIPS platform. For general discussion on new architectures, join the FreeBSD non-Intel platforms porting mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-platforms).

If your machine has a different architecture and you need something right now, we suggest you look at NetBSD (http://www.netbsd.org/) or OpenBSD (http://www.openbsd.org/).

2. Does FreeBSD support Symmetric Multiprocessing (SMP)?

Symmetric multi-processor (SMP) systems are generally supported by FreeBSD, although in some cases, BIOS or motherboard bugs may generate some problems. Perusing the FreeBSD symmetric multiprocessing mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-smp) may yield some clues.

FreeBSD will take advantage of HyperThreading (HTT) support on Intel CPUs that support this feature. A kernel with the `options SMP` feature enabled will automatically detect the additional logical processors. The default FreeBSD scheduler treats the logical processors the same as additional physical processors; in other words, no attempt is made to optimize scheduling decisions given the shared resources between logical processors within the same CPU. Because this naive scheduling can result in suboptimal performance, under certain circumstances it may be useful to disable the logical processors with the `machdep.hlt_logical_cpus` sysctl variable. It is also possible to halt any CPU in the idle loop with the `machdep.hlt_cpus` sysctl variable. The smp(4) manual page has more details.

4.4 Hard Drives, Tape Drives, and CD and DVD Drives

1. What kind of hard drives does FreeBSD support?

FreeBSD supports EIDE, SATA, SCSI, and SAS drives (with a compatible controller; see the next section), and all drives using the original “Western Digital” interface (MFM, RLL, ESDI, and of course IDE). A few ESDI controllers

that use proprietary interfaces may not work: stick to WD1002/3/6/7 interfaces and clones.

2. Which SCSI or SAS controllers are supported?

See the complete list in the Hardware Notes for FreeBSD 8.1 (http://www.FreeBSD.org/releases/8.1R/hardware.html) or 7.3 (http://www.FreeBSD.org/releases/7.3R/hardware.html).

3. What types of tape drives are supported?

FreeBSD supports SCSI and QIC-36 (with a QIC-02 interface). This includes 8-mm (aka Exabyte) and DAT drives.

Some of the early 8-mm drives are not quite compatible with SCSI-2, and may not work well with FreeBSD.

4. Does FreeBSD support tape changers?

FreeBSD supports SCSI changers using the ch(4) device and the chio(1) command. The details of how you actually control the changer can be found in the chio(1) manual page.

If you are not using **AMANDA** or some other product that already understands changers, remember that they only know how to move a tape from one point to another, so you need to keep track of which slot a tape is in, and which slot the tape currently in the drive needs to go back to.

5. Which CD-ROM drives are supported by FreeBSD?

Any SCSI drive connected to a supported controller is supported.

The following proprietary CD-ROM interfaces are also supported:

- Mitsumi LU002 (8-bit), LU005 (16-bit) and FX001D (16-bit 2x Speed).
- Sony CDU 31/33A
- Sound Blaster Non-SCSI CD-ROM
- Matsushita/Panasonic CD-ROM
- ATAPI compatible IDE CD-ROMs

All non-SCSI cards are known to be extremely slow compared to SCSI drives, and some ATAPI CD-ROMs may not work.

The official FreeBSD CD-ROM ISO, and CD-ROMs from Daemon News and FreeBSD Mall, support booting directly from the CD.

6. Which CD-RW drives are supported by FreeBSD?

FreeBSD supports any ATAPI-compatible IDE CD-R or CD-RW drive. See burncd(8) for details.

FreeBSD also supports any SCSI CD-R or CD-RW drives. Install and use the `cdrecord` command from the ports or packages system, and make sure that you have the `pass` device compiled in your kernel.

7. Does FreeBSD support Zip drives?

FreeBSD supports SCSI and ATAPI (IDE) Zip drives out of the box. SCSI ZIP drives can only be set to run at SCSI target IDs 5 or 6, but if your SCSI host adapter's BIOS supports it you can even boot from it. It is not clear which host adapters support booting from targets other than 0 or 1, so you will have to consult your adapter's documentation if you would like to use this feature.

FreeBSD also supports Parallel Port Zip Drives. Check that your kernel contains the `scbus0`, `da0`, `ppbus0`, and `vp0` drivers (the `GENERIC` kernel contains everything except `vp0`). With all these drivers present, the Parallel Port drive should be available as `/dev/da0s4`. Disks can be mounted using `mount /dev/da0s4 /mnt` OR (for DOS disks) `mount -t msdosfs /dev/da0s4 /mnt` as appropriate.

Also check out the FAQ on removable drives later in this chapter, and the note on "formatting"in the Administration chapter.

8. Does FreeBSD support Jaz, EZ and other removable drives?

They work. Most of these are SCSI devices, so they look like SCSI disks to FreeBSD. The IDE EZ looks like an IDE drive.

Make sure that any external units are powered on when booting the system.

To change the media while running, check out mount(8), umount(8), and camcontrol(8) (for SCSI devices) or atacontrol(8) (for IDE devices), plus the discussion on using removable drives later in the FAQ.

4.5 Keyboards and Mice

1. Does FreeBSD support my USB keyboard?

FreeBSD supports USB keyboards out-of-the-box. Once you have USB keyboard support enabled on your system, the AT keyboard becomes `/dev/kbd0` and the USB keyboard becomes `/dev/kbd1`, if both are connected to the system. If there is the USB keyboard only, it will be `/dev/ukbd0`.

If you want to use the USB keyboard in the console, you have to explicitly tell the console driver to use the existing USB keyboard. This can be done by running the following command as a part of system initialization.

```
# kbdcontrol -k /dev/kbd1 < /dev/console > /dev/null
```

Note that if the USB keyboard is the only keyboard, it is accessed as `/dev/ukbd0`, thus, the command should look like:

```
# kbdcontrol -k /dev/ukbd0 < /dev/console > /dev/null
```

> **Note:** To make this change permanent across reboots, add `keyboard="/dev/ukbd0"` to `/etc/rc.conf`.

Once this is done, the USB keyboard should work in the X environment as well without any special settings.

If you want to switch back to the default keyboard, use this command:

```
# kbdcontrol -k /dev/kbd0 > /dev/null
```

To allow using both the second USB keyboard and the first AT keyboard at the same time on a console via kbdmux(4) driver type the following commands:

```
# kbdcontrol -K < /dev/console > /dev/null
# kbdcontrol -a atkbd0 < /dev/kbdmux0 > /dev/null
# kbdcontrol -a ukbd1 < /dev/kbdmux0 > /dev/null
# kbdcontrol -k /dev/kbdmux0 < /dev/console > /dev/null
```

See the ukbd(4), kbdcontrol(1) and kbdmux(4) manual pages for more information.

> **Note:** Hot-plugging and unplugging of the USB keyboard may not work quite right yet. We recommend connecting the keyboard before starting the system and leaving it connected until the system is shutdown to avoid issues.

2. I have an unusual bus mouse. How do I set it up?

FreeBSD supports the bus mouse and the InPort bus mouse from such manufacturers as Microsoft, Logitech and ATI. The `GENERIC` kernel does not include the device driver. To build a custom kernel with the bus mouse driver, add the following line to the kernel config file:

```
device mse0 at isa? port 0x23c irq5
```

Bus mice usually come with dedicated interface cards. These cards may allow you to set the port address and the IRQ number other than shown above. Refer to the manual of your mouse and the mse(4) manual page for more information.

3. How do I use my PS/2 (“mouse port” or “keyboard”) mouse?

The PS/2 mouse is supported out-of-the-box. The necessary device driver, `psm`, is included in the kernel.

If your custom kernel does not have this, add the following line to your kernel configuration and compile a new kernel.

```
device psm0 at atkbdc? irq 12
```

Once the kernel detects `psm0` correctly at boot time, a device node `psm0` will be created automatically.

4. Is it possible to use a mouse in any way outside the X Window system?

If you are using the default console driver, syscons(4), you can use a mouse pointer in text consoles to cut & paste text. Run the mouse daemon, moused(8), and turn on the mouse pointer in the virtual console:

```
# moused -p /dev/xxxx -t yyyy
# vidcontrol -m on
```

Where *xxxx* is the mouse device name and *yyyy* is a protocol type for the mouse. The mouse daemon can automatically determine the protocol type of most mice, except old serial mice. Specify the `auto` protocol to invoke automatic detection. If automatic detection does not work, see the moused(8) manual page for a list of supported protocol types.

If you have a PS/2 mouse, just add `moused_enable="YES"` to `/etc/rc.conf` to start the mouse daemon at boot-time. Additionally, if you would like to use the mouse daemon on all virtual terminals instead of just the console, add `allscreens_flags="-m on"` to `/etc/rc.conf`.

When the mouse daemon is running, access to the mouse must be coordinated between the mouse daemon and other programs such as X Windows. Refer to the FAQ Why does my mouse not work with X? for more details on this issue.

5. How do I cut and paste text with a mouse in the text console?

Once you get the mouse daemon running (see the previous section), hold down the button 1 (left button) and move the mouse to select a region of text. Then, press the button 2 (middle button) to paste it at the text cursor. Pressing button 3 (right button) will "extend" the selected region of text.

If your mouse does not have a middle button, you may wish to emulate one or remap buttons using mouse daemon options. See the moused(8) manual page for details.

6. My mouse has a fancy wheel and buttons. Can I use them in FreeBSD?

The answer is, unfortunately, "It depends". These mice with additional features require specialized driver in most cases. Unless the mouse device driver or the user program has specific support for the mouse, it will act just like a standard two, or three button mouse.

For the possible usage of wheels in the X Window environment, refer to that section.

7. How do I use the mouse/trackball/touchpad on my laptop?

Please refer to the answer to the previous question.

8. How do I use my delete key in `sh` and `csh`?

For the **Bourne Shell**, add the following lines to your `.shrc`. See sh(1) and editrc(5).

```
bind ^? ed-delete-next-char # for console
bind ^[[3~ ed-delete-next-char # for xterm
```

For the **C Shell**, add the following lines to your `.cshrc`. See csh(1).

```
bindkey ^? delete-char # for console
bindkey ^[[3~ delete-char # for xterm
```

For more information, see this page (http://www.ibb.net/~anne/keyboard.html).

4.6 Networking and Serial Devices

1. Which network cards does FreeBSD support?

See the Hardware Notes supplied with each release of FreeBSD for a more complete list.

2. Does FreeBSD support software modems, such as Winmodems?

FreeBSD supports many software modems via add-on software. For example, the `comms/ltmdm` port adds support for modems based on the very popular Lucent LT chipsets.

You cannot install FreeBSD via a software modem; this software must be installed after the OS is installed.

3. Is there a native driver for the Broadcom 43xx cards?

No, and there is not likely to be.

Broadcom refuses to publically release programming information for their wireless chipsets, most likely because they use software controlled radios. In order to get FCC type acceptance for their parts, they have to ensure that users cannot arbitrarily set things like operating frequencies, modulation parameters and power output. But without knowing how to program the chipsets, it is nearly impossible to write a driver.

4. Which multi-port serial cards are supported by FreeBSD?

There is a list of these in the Serial Communications (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/serial.html) chapter of the handbook.

Some unnamed clone cards have also been known to work, especially those that claim to be AST compatible.

Check the sio(4) manual page to get more information on configuring such cards.

5. How do I get the boot: prompt to show on the serial console?

See this section of the handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/serialconsole-setup.html).

4.7 Sound Devices

1. Which sound cards are supported by FreeBSD?

FreeBSD supports various sound cards (for more details, see FreeBSD Release Information (http://www.FreeBSD.org/releases/) and the snd(4) manual page). There is also limited support for MPU-401 and compatible MIDI cards. Cards conforming to the Microsoft® Sound System specification are also supported.

> **Note:** This is only for sound! This driver does not support CD-ROMs, SCSI or joysticks on these cards, except for the SoundBlaster®. The SoundBlaster SCSI interface and some non-SCSI CD-ROMs are supported, but you cannot boot off this device.

2. Workarounds for no sound from my pcm(4) sound card?

Some sound cards set their output volume to 0 at every boot. Run the following command every time the machine boots:

```
# mixer pcm 100 vol 100 cd 100
```

4.8 Other Hardware

1. Does FreeBSD support power management on my laptop?

FreeBSD supports APM on certain machines. Further information can be found in apm(4).

FreeBSD also supports the ACPI features found in most modern hardware. Further information can be found in acpi(4). If a system supports both APM and ACPI, either can be used. We suggest you try both and choose the one that best fits your needs.

2. How do I disable ACPI?

Add following line

```
hint.acpi.0.disabled="1"
```

into your `/boot/device.hints` file.

3. Why does my Micron system hang at boot time?

Certain Micron motherboards have a non-conforming PCI BIOS implementation that causes grief when FreeBSD boots because PCI devices do not get configured at their reported addresses.

Disable the “Plug and Play Operating System” flag in the BIOS to work around this problem.

4. The boot floppy hangs on a system with an ASUS K7V motherboard. How do I fix this?

Go into the BIOS setup and disable the “boot virus protection”.

5. Why does my 3Com® PCI network card not work with my Micron computer?

See the previous answer.

Chapter 5 Troubleshooting

1. Why is FreeBSD finding the wrong amount of memory on i386 hardware?

The most likely reason is the difference between physical memory addresses and virtual addresses.

The convention for most PC hardware is to use the memory area between 3.5 GB and 4 GB for a special purpose (usually for PCI). This address space is used to access PCI hardware. As a result real, physical memory can not be accessed by that address space.

What happens to the memory that should appear in that location is dependent on your hardware. Unfortunately, some hardware does nothing and the ability to use that last 500 MB of RAM is entirely lost.

Luckily, most hardware remaps the memory to a higher location so that it can still be used. However, this can cause some confusion if you watch the boot messages.

On a 32-bit version of FreeBSD, the memory appears lost, since it will be remapped above 4 GB, which a 32-bit kernel is unable to access. In this case, the solution is to build a PAE enabled kernel. See the entry on memory limits and about different memory limits on different platforms for more information.

On a 64-bit version of FreeBSD, or when running a PAE-enabled kernel, FreeBSD will correctly detect and remap the memory so it is usable. During boot, however, it may seem as if FreeBSD is detecting more memory than the system really has, due to the described remapping. This is normal and the available memory will be corrected as the boot process completes.

2. What do I do when I have bad blocks on my hard drive?

With SCSI drives, the drive should be capable of re-mapping these automatically. However, many drives ship with this feature disabled.

To enable bad block remapping edit the first device page mode, which can be done by giving the command (as `root`)

```
# camcontrol modepage sd0 -m 1 -e -P 3
```

and changing the values of AWRE and ARRE from 0 to 1:

```
AWRE (Auto Write Reallocation Enbld):  1
ARRE (Auto Read Reallocation Enbld):  1
```

Modern IDE drives also have bad block remapping features in the controller, and they ship with this feature turned on.

If you see warnings about bad blocks (on either type of drive), it is time to consider replacing the drive. You might be able to use the drive manufacturer's diagnostic program to lock out those bad blocks, but at best this will buy you some time.

3. Why does FreeBSD not detect my HP Netserver's SCSI controller?

This is basically a known problem. The EISA on-board SCSI controller in the HP Netserver machines occupies EISA slot number 11, so all the "true" EISA slots are in front of it. Alas, the address space for EISA slots >= 10

collides with the address space assigned to PCI, and FreeBSD's auto-configuration currently cannot handle this situation very well.

So now, the best you can do is to pretend there is no address range clash :), by bumping the kernel option `EISA_SLOTS` to a value of 12. Configure and compile a kernel, as described in the Handbook entry on configuring the kernel (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig.html).

Of course, this does present you with a chicken-and-egg problem when installing on such a machine. In order to work around this problem, a special hack is available inside *UserConfig*. Do not use the "visual" interface, but the plain command-line interface there. Simply type the following command at the prompt and install your system as usual:

```
eisa 12
quit
```

While it is recommended you compile and install a custom kernel anyway.

Hopefully, future versions will have a proper fix for this problem.

> **Note:** You cannot use a `dangerously dedicated` disk with an HP Netserver. See this note for more info.

4. I keep seeing messages like "`ed1: timeout`". What do these messages mean?

This is usually caused by an interrupt conflict (e.g., two boards using the same IRQ). Boot with the `-c` option and change the `ed0`/`de0`/... entry to match your board.

If you are using the BNC connector on your network card, you may also see device timeouts because of bad termination. To check this, attach a terminator directly to the NIC (with no cable) and see if the error messages go away.

Some NE2000 compatible cards will give this error if there is no link on the UTP port or if the cable is disconnected.

5. Why did my 3Com 3C509 card stop working for no apparent reason?

This card has a bad habit of losing its configuration information. Refresh your card's settings with the DOS utility `3c5x9.exe`.

6. My parallel printer is ridiculously slow. What can I do?

If the only problem is that the printer is terribly slow, try changing your printer port mode (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/printing-intro-setup.html#PRINTING-PARALLEL-PORT-MODE) as discussed in the Printer Setup (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/printing-intro-setup.html) section of the Handbook.

7. Why do my programs occasionally die with "`Signal 11`" errors?

Signal 11 errors are caused when your process has attempted to access memory which the operating system has not granted it access to. If something like this is happening at seemingly random intervals then you need to start investigating things very carefully.

These problems can usually be attributed to either:

1. If the problem is occurring only in a specific application that you are developing yourself it is probably a bug in your code.
2. If it is a problem with part of the base FreeBSD system, it may also be buggy code, but more often than not these problems are found and fixed long before us general FAQ readers get to use these bits of code (that is what -current is for).

In particular, a dead giveaway that this is *not* a FreeBSD bug is if you see the problem when you are compiling a program, but the activity that the compiler is carrying out changes each time.

For example, suppose you are running `make buildworld`, and the compile fails while trying to compile `ls.c` into `ls.o`. If you then run `make buildworld` again, and the compile fails in the same place then this is a broken build — try updating your sources and try again. If the compile fails elsewhere then this is almost certainly hardware.

What you should do:

In the first case you can use a debugger e.g. gdb(1) to find the point in the program which is attempting to access a bogus address and then fix it.

In the second case you need to verify that it is not your hardware at fault.

Common causes of this include:

1. Your hard disks might be overheating: Check the fans in your case are still working, as your disk (and perhaps other hardware might be overheating).
2. The processor running is overheating: This might be because the processor has been overclocked, or the fan on the processor might have died. In either case you need to ensure that you have hardware running at what it is specified to run at, at least while trying to solve this problem. i.e. Clock it back to the default settings.

   If you are overclocking then note that it is far cheaper to have a slow system than a fried system that needs replacing! Also the wider community is not often sympathetic to problems on overclocked systems, whether you believe it is safe or not.
3. Dodgy memory: If you have multiple memory SIMMS/DIMMS installed then pull them all out and try running the machine with each SIMM or DIMM individually and narrow the problem down to either the problematic DIMM/SIMM or perhaps even a combination.
4. Over-optimistic Motherboard settings: In your BIOS settings, and some motherboard jumpers you have options to set various timings, mostly the defaults will be sufficient, but sometimes, setting the wait states on RAM too low, or setting the “RAM Speed: Turbo” option, or similar in the BIOS will cause strange behavior. A possible idea is to set to BIOS defaults, but it might be worth noting down your settings first!
5. Unclean or insufficient power to the motherboard. If you have any unused I/O boards, hard disks, or CD-ROMs in your system, try temporarily removing them or disconnecting the power cable from them, to see if your power supply can manage a smaller load. Or try another power supply, preferably one with a little more power (for instance, if your current power supply is rated at 250 Watts try one rated at 300 Watts).

You should also read the SIG11 FAQ (listed below) which has excellent explanations of all these problems, albeit from a Linux® viewpoint. It also discusses how memory testing software or hardware can still pass faulty memory.

Finally, if none of this has helped it is possible that you have just found a bug in FreeBSD, and you should follow the instructions to send a problem report.

There is an extensive FAQ on this at the SIG11 problem FAQ (http://www.bitwizard.nl/sig11/).

8. My system crashes with either "`Fatal trap 12: page fault in kernel mode`", or "`panic:`", and spits out a bunch of information. What should I do?

The FreeBSD developers are very interested in these errors, but need some more information than just the error you see. Copy your full crash message. Then consult the FAQ section on kernel panics, build a debugging kernel, and get a backtrace. This might sound difficult, but you do not need any programming skills; you just have to follow the instructions.

9. Why does the screen go black and lose sync when I boot?

This is a known problem with the ATI Mach64 video card. The problem is that this card uses address `2e8`, and the fourth serial port does too. Due to a bug (feature?) in the sio(4) driver it will touch this port even if you do not have the fourth serial port, and *even* if you disable `sio3` (the fourth port) which normally uses this address.

Until the bug has been fixed, you can use this workaround:

1. Enter `-c` at the boot prompt. (This will put the kernel into configuration mode).
2. Disable `sio0`, `sio1`, `sio2` and `sio3` (all of them). This way the sio(4) driver does not get activated — no problems.
3. Type exit to continue booting.

If you want to be able to use your serial ports, you will have to build a new kernel with the following modification: in `/usr/src/sys/dev/sio/sio.c` (or in `/usr/src/sys/pc98/cbus/sio.c` for pc98) find the one occurrence of the string `0x2e8` and remove that string and the preceding comma (keep the trailing comma). Now follow the normal procedure of building a new kernel.

10. Why does FreeBSD only use 64 MB of RAM when my system has 128 MB of RAM installed?

Due to the manner in which FreeBSD gets the memory size from the BIOS, it can only detect 16 bits worth of Kbytes in size (65535 Kbytes = 64 MB) (or less... some BIOSes peg the memory size to 16 MB). If you have more than 64 MB, FreeBSD will attempt to detect it; however, the attempt may fail.

To work around this problem, you need to use the kernel option specified below. There is a way to get complete memory information from the BIOS, but we do not have room in the bootblocks to do it. Someday when lack of room in the bootblocks is fixed, we will use the extended BIOS functions to get the full memory information... but for now we are stuck with the kernel option.

```
options MAXMEM=n
```

Where *n* is your memory in Kilobytes. For a 128 MB machine, you would want to use `131072`.

11. My system has more than 1 GB of RAM, and I'm getting panics with "`kmem_map too small`" messages. What is wrong?

Normally, FreeBSD determines a number of kernel parameters, such as as the maximum number of files that can be open concurrently, from the amount of memory installed in the system. On systems with one gigabyte of RAM or more, this "auto sizing" mechanism may choose values that are too high: while starting up, the kernel allocates various tables and other structures that fill up most of the available kernel memory. Later on, while the system is running, the kernel has no more space left for dynamic memory allocations, and panics.

Compile your own kernel, and add the `VM_KMEM_SIZE_MAX` to your kernel configuration file, increasing the maximum size to 400 MB (`options VM_KMEM_SIZE_MAX=419430400`). 400 MB appears to be sufficient for machines with up to 6 GB of memory.

12. My system does not have 1 GB of RAM, and FreeBSD still panics with "`kmem_map too small`"!

The panic indicates that the system ran out of virtual memory for network buffers (specifically, mbuf clusters). You can increase the amount of VM available for mbuf clusters by following the instructions in the Network Limits (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/configtuning-kernel-limits.html#NMBCLUSTERS) section of the Handbook.

13. Why do I get the error "`kernel: proc: table is full`"?

The FreeBSD kernel will only allow a certain number of processes to exist at one time. The number is based on the `kern.maxusers` sysctl(8) variable. `kern.maxusers` also affects various other in-kernel limits, such as network buffers (see this earlier question). If your machine is heavily loaded, you probably want to increase `kern.maxusers`. This will increase these other system limits in addition to the maximum number of processes.

To adjust your `kern.maxusers` value, see the File/Process Limits (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/configtuning-kernel-limits.html#KERN-MAXFILES) section of the Handbook. (While that section refers to open files, the same limits apply to processes.)

If your machine is lightly loaded, and you are simply running a very large number of processes, you can adjust this with the `kern.maxproc` tunable. If this tunable needs adjustment it needs to be defined in `/boot/loader.conf`. The tunable will not get adjusted until the system is rebooted. For more information about tuning tunables, you should see the loader.conf(5) and sysctl.conf(5) manual pages. If these processes are being run by a single user, you will also need to adjust `kern.maxprocperuid` to be one less than your new `kern.maxproc` value. (It must be at least one less because one system program, init(8), must always be running.)

To make a sysctl change permanent place the proper value in `/etc/sysctl.conf`. More information about system tuning with sysctl(8) can be found at the Tuning with sysctl (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/configtuning-sysctl.html) section of the Handbook.

14. Why do I get an error reading "`CMAP busy`" when rebooting with a new kernel?

The logic that attempts to detect an out of date `/var/db/kvm_*.db` files sometimes fails and using a mismatched file can sometimes lead to panics.

If this happens, reboot single-user and do:

```
# rm /var/db/kvm_*.db
```

15. What does the message "`ahc0: brkadrint, Illegal Host Access at seqaddr 0x0`" mean?

This is a conflict with an Ultrastor SCSI Host Adapter.

During the boot process enter the kernel configuration menu and disable `uha0`, which is causing the problem.

16. When I boot my system, I get the error "`ahc0: illegal cable configuration`". My cabling is correct. What is going on?

Your motherboard lacks the external logic to support automatic termination. Switch your SCSI BIOS to specify the correct termination for your configuration rather than automatic termination. The ahc(4) driver cannot determine if the external logic for cable detection (and thus auto-termination) is available. The driver simply assumes that this support must exist if the configuration contained in the serial EEPROM is set to "automatic termination". Without the external cable detection logic the driver will often configure termination incorrectly, which can compromise the reliability of the SCSI bus.

17. Why does **sendmail** give me an error reading "`mail loops back to myself`"?

You can find a detailed answer for this question in the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/mail-trouble.html#Q26.5.2.).

18. Why do full screen applications on remote machines misbehave?

The remote machine may be setting your terminal type to something other than the `cons25` terminal type required by the FreeBSD console.

There are a number of possible work-arounds for this problem:

- After logging on to the remote machine, set your `TERM` shell variable to `ansi` or `sco` if the remote machine knows about these terminal types.
- Use a VT100 emulator like **screen** at the FreeBSD console. **screen** offers you the ability to run multiple concurrent sessions from one terminal, and is a neat program in its own right. Each **screen** window behaves like a VT100 terminal, so the `TERM` variable at the remote end should be set to `vt100`.
- Install the `cons25` terminal database entry on the remote machine. The way to do this depends on the operating system on the remote machine. The system administration manuals for the remote system should be able to help you here.
- Fire up an X server at the FreeBSD end and login to the remote machine using an X based terminal emulator such as `xterm` or `rxvt`. The `TERM` variable at the remote host should be set to `xterm` or `vt100`.

19. Why is my PnP card not found (or found as `unknown`)?

The reasons for this behavior are explained by the following email, posted to the FreeBSD general questions mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-questions) by Peter Wemm <`peter@FreeBSD.org`>, in answer to a question about an internal modem that was no longer found after an upgrade to FreeBSD 4.*X* (the comments in `[]` have been added to clarify the context).

> **Note:** The contents of this quotation has been updated from its original text.

> The PNP bios preconfigured it [the modem] and left it laying around in port space, so [in 3.*X*] the old-style ISA probes "found" it there.
>
> Under 4.0, the ISA code is much more PnP-centric. It was possible [in 3.*X*] for an ISA probe to find a "stray" device and then for the PNP device ID to match and then fail due to resource conflicts. So, it disables the programmable cards first so

this double probing cannot happen. It also means that it needs to know the PnP IDs for supported PnP hardware. Making this more user tweakable is on the TODO list.

To get the device working again requires finding its PnP ID and adding it to the list that the ISA probes use to identify PnP devices. This is obtained using pnpinfo(8) to probe the device, for example this is the output from pnpinfo(8) for an internal modem:

```
# pnpinfo
Checking for Plug-n-Play devices...

Card assigned CSN #1
Vendor ID PMC2430 (0x3024a341), Serial Number 0xffffffff
PnP Version 1.0, Vendor Version 0
Device Description: Pace 56 Voice Internal Plug & Play Modem

Logical Device ID: PMC2430 0x3024a341 #0
        Device supports I/O Range Check
TAG Start DF
    I/O Range 0x3f8 .. 0x3f8, alignment 0x8, len 0x8
        [16-bit addr]
    IRQ: 4  - only one type (true/edge)
```

[more TAG lines elided]

```
TAG End DF
End Tag

Successfully got 31 resources, 1 logical fdevs
-- card select # 0x0001

CSN PMC2430 (0x3024a341), Serial Number 0xffffffff

Logical device #0
IO:  0x03e8 0x03e8 0x03e8 0x03e8 0x03e8 0x03e8 0x03e8 0x03e8
IRQ 5 0
DMA 4 0
IO range check 0x00 activate 0x01
```

The information you require is in the `Vendor ID` line at the start of the output. The hexadecimal number in parentheses (`0x3024a341` in this example) is the PnP ID and the string immediately before this (`PMC2430`) is a unique ASCII ID.

Alternatively, if pnpinfo(8) does not list the card in question, pciconf(8) can be used instead. This is part of the output from `pciconf -vl` for an onboard sound chip:

```
# pciconf -vl
chip1@pci0:31:5:        class=0x040100 card=0x00931028 chip=0x24158086 rev=0x02 hdr=0x00
    vendor   = 'Intel Corporation'
    device   = '82801AA 8xx Chipset AC'97 Audio Controller'
    class    = multimedia
    subclass = audio
```

Here, you would use the `chip` value, `0x24158086`.

This information (`Vendor ID` or `chip` value) needs adding to the file `/usr/src/sys/dev/sio/sio_isa.c`.

You should first make a backup of `sio_isa.c` just in case things go wrong. You will also need it to make the patch to submit with your PR (you are going to submit a PR, are you not?) then edit `sio_isa.c` and search for the line:

```
static struct isa_pnp_id sio_ids[] = {
```

Then scroll down to find the correct place to add the entry for your device. The entries look like this, and are sorted on the ASCII Vendor ID string which should be included in the comment to the right of the line of code along with all (if it will fit) or part of the *Device Description* from the output of pnpinfo(8):

```
{0x0f804f3f, NULL},     /* OZO800f - Zoom 2812 (56k Modem) */
{0x39804f3f, NULL},     /* OZO8039 - Zoom 56k flex */
{0x3024a341, NULL},     /* PMC2430 - Pace 56 Voice Internal Modem */
{0x1000eb49, NULL},     /* ROK0010 - Rockwell ? */
{0x5002734a, NULL},     /* RSS0250 - 5614Jx3(G) Internal Modem */
```

Add the hexadecimal Vendor ID for your device in the correct place, save the file, rebuild your kernel, and reboot. Your device should now be found as an `sio` device.

20. Why do I get the error "`nlist failed`" when running, for example, `top` or `systat`?

The problem is that the application you are trying to run is looking for a specific kernel symbol, but, for whatever reason, cannot find it; this error stems from one of two problems:

- Your kernel and userland are not synchronized (i.e., you built a new kernel but did not do an `installworld`, or vice versa), and thus the symbol table is different from what the user application thinks it is. If this is the case, simply complete the upgrade process (see `/usr/src/UPDATING` for the correct sequence).
- You are not using `/boot/loader` to load your kernel, but doing it directly from `boot2` (see boot(8)). While there is nothing wrong with bypassing `/boot/loader`, it generally does a better job of making the kernel symbols available to user applications.

21. Why does it take so long to connect to my computer via `ssh` or `telnet`?

The symptom: there is a long delay between the time the TCP connection is established and the time when the client software asks for a password (or, in telnet(1)'s case, when a login prompt appears).

The problem: more likely than not, the delay is caused by the server software trying to resolve the client's IP address into a hostname. Many servers, including the **Telnet** and **SSH** servers that come with FreeBSD, do this in order to, among other things, store the hostname in a log file for future reference by the administrator.

The remedy: if the problem occurs whenever you connect from your computer (the client) to any server, the problem is with the client; likewise, if the problem only occurs when someone connects to your computer (the server) the problem is with the server.

If the problem is with the client, the only remedy is to fix the DNS so the server can resolve it. If this is on a local network, consider it a server problem and keep reading; conversely, if this is on the global Internet, you will most likely need to contact your ISP and ask them to fix it for you.

If the problem is with the server, and this is on a local network, you need to configure the server to be able to resolve address-to-hostname queries for your local address range. See the hosts(5) and named(8) manual pages for more

information. If this is on the global Internet, the problem may be that your server's resolver is not functioning correctly. To check, try to look up another host — say, `www.yahoo.com`. If it does not work, that is your problem.

Following a fresh install of FreeBSD, it is also possible that domain and name server information is missing from `/etc/resolv.conf`. This will often cause a delay in **SSH**, as the option `UseDNS` is set to `yes` by default in the `sshd_config` file in `/etc/ssh`. If this is causing the problem, you will either need to fill in the missing information in `/etc/resolv.conf` or set `UseDNS` to `no` in `sshd_config` as a temporary workaround.

22. What does "`stray IRQ`" mean?

Stray IRQs are indications of hardware IRQ glitches, mostly from hardware that removes its interrupt request in the middle of the interrupt request acknowledge cycle.

One has three options for dealing with this:

- Live with the warnings. All except the first 5 per irq are suppressed anyway.
- Break the warnings by changing the value of `MAX_STRAY_LOG` from `5` to `0` in your platform's (e.g. i386) `intr_machdep.c` file and rebuild the new kernel and all the warnings will be suppressed.
- Break the warnings by installing parallel port hardware that uses IRQ 7 and the PPP driver for it (this happens on most systems), and install an ide drive or other hardware that uses IRQ 15 and a suitable driver for it.

23. Why does "`file: table is full`" show up repeatedly in dmesg(8)?

This error message indicates you have exhausted the number of available file descriptors on your system. Please see the kern.maxfiles (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/configtuning-kernel-limits.html#KERN-MAXFILES) section of the Tuning Kernel Limits (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/configtuning-kernel-limits.html) section of the Handbook for a discussion and solution.

24. Why are "`calcru: negative runtime`" or "`calcru: runtime went backwards`" messages pounding the console?

There is a known problem when enabling Intel Enhanced SpeedStep from the BIOS: it causes the kernel to start printing "`calcru`" messages like this:

```
calcru: runtime went backwards from 6 usec to 3 usec for pid 37 (pagezero)
calcru: runtime went backwards from 6 usec to 3 usec for pid 36 (vmdaemon)
calcru: runtime went backwards from 170 usec to 138 usec for pid 35 (pagedaemon)
calcru: runtime went backwards from 553 usec to 291 usec for pid 15 (swi6: task queue)
calcru: runtime went backwards from 15521 usec to 10366 usec for pid 2 (g_event)
calcru: runtime went backwards from 25 usec to 12 usec for pid 11 (swi1: net)
calcru: runtime went backwards from 4417 usec to 3960 usec for pid 1 (init)
calcru: runtime went backwards from 2084385 usec to 1793542 usec for pid 1 (init)
calcru: runtime went backwards from 408 usec to 204 usec for pid 0 (swapper)
```

It is because Intel SpeedStep (EIST) is incompatible with some motherboards.

Workaround: Disable the EIST feature in the BIOS. You can still achieve ACPI-based processor frequency throttling by using powerd(8).

25. Why does the clock on my computer keep incorrect time?

Your computer has two or more clocks, and FreeBSD has chosen to use the wrong one.

Run dmesg(8), and check for lines that contain `Timecounter`. The one with the highest quality value that FreeBSD chose.

```
# dmesg | grep Timecounter
Timecounter "i8254" frequency 1193182 Hz quality 0
Timecounter "ACPI-fast" frequency 3579545 Hz quality 1000
Timecounter "TSC" frequency 2998570050 Hz quality 800
Timecounters tick every 1.000 msec
```

You can confirm this by checking the `kern.timecounter.hardware` sysctl(3).

```
# sysctl kern.timecounter.hardware
kern.timecounter.hardware: ACPI-fast
```

It may be a broken ACPI timer. The simplest solution is to disable the ACPI timer in `/etc/loader.conf`:

```
debug.acpi.disabled="timer"
```

Or the BIOS may modify the TSC clock—perhaps to change the speed of the processor when running from batteries, or going into a power saving mode, but FreeBSD is unaware of these adjustments, and appears to gain or lose time.

In this example, the `i8254` clock is also available, and can be selected by writing its name to the `kern.timecounter.hardware` sysctl(3).

```
# sysctl -w kern.timecounter.hardware=i8254
kern.timecounter.hardware: TSC -> i8254
```

Your computer should now start keeping more accurate time.

To have this change automatically run at boot time, add the following line to `/etc/sysctl.conf`:

```
kern.timecounter.hardware=i8254
```

26. Why did my laptop fail to correctly probe PC cards?

This problem is common on laptops that boot more than one operating system. Some non-BSD operating systems leave PC card hardware in an inconsistent state. pccardd(8) will detect the card as "`"(null)""(null)"`" instead of its actual model.

You must remove all power from the PC card slot to fully reset the hardware. Completely power off the laptop. (Do not suspend it, do not let it go into standby; the power needs to be completely off.) Wait a few moments, and reboot. Your PC card should work now.

Some laptop hardware lies when it claims to be off. If the above does not work shut down, remove the battery, wait a moment, replace the battery, and reboot.

27. Why does FreeBSD's boot loader display "`Read error`" and stop after the BIOS screen?

FreeBSD's boot loader is incorrectly recognizing the hard drive's geometry. This must be manually set within fdisk(8) when creating or modifying FreeBSD's slice.

The correct drive geometry values can be found within the machine's BIOS. Look for the number of cylinders, heads and sectors for the particular drive.

Within sysinstall(8)'s fdisk, hit **G** to set the drive geometry.

A dialog will pop up requesting the number of cylinders, heads and sectors. Type the numbers found from the BIOS separated by forward slashes. For example, values of 5000 cylinders, 250 heads, and 60 sectors would be entered as **`5000/250/60`**.

Press **Enter** to set the values, and hit **W** to write the new partition table to the drive.

28. Another operating system destroyed my Boot Manager. How do I get it back?

Enter sysinstall(8) and choose Configure, then Fdisk. Select the disk the Boot Manager resided on with the **Space** key. Press **W** to write changes to the drive. A prompt will appear asking which boot loader to install. Select this, and it will be restored.

29. What does the error "`swap_pager: indefinite wait buffer:`" mean?

This means that a process is trying to page memory to disk, and the page attempt has hung trying to access the disk for more than 20 seconds. It might be caused by bad blocks on the disk drive, disk wiring, cables, or any other disk I/O-related hardware. If the drive itself is actually bad, you will also see disk errors in `/var/log/messages` and in the output of `dmesg`. Otherwise, check your cables and connections.

30. What are "`UDMA ICRC`" errors, and how do I fix them?

The ata(4) driver reports "`UDMA ICRC`" errors when a DMA transfer to or from a drive is corrupted. The driver will retry the operation a few times. Should the retries fail, it will switch from DMA to the slower PIO mode of communication with the device.

The problem can be caused by many factors, although perhaps the most common cause is faulty or incorrect cabling. Check that the ATA cables are undamaged and rated for the Ultra DMA mode in use. If you are using removable drive trays, they must also be compatible. Be sure that all connections are making good contact. Problems have also been noticed when an old drive is installed on the same ATA channel as an Ultra DMA 66 (or faster) drive. Lastly, these errors can indicate that the drive is failing. Most drive vendors provide testing software for their drives, so test your drive, and, if necessary, back up your data and replace it.

The atacontrol(8) utility can be used to show and select the DMA or PIO modes used for each ATA device. In particular, `atacontrol mode` *`channel`* will show the modes in use on a particular ATA channel, where the primary channel is numbered 0, and so on.

31. What is a "`lock order reversal`"?

An answer for this question can be found in the FreeBSD Glossary, see LOR (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/freebsd-glossary.html#LOR-GLOSSARY).

32. What does "`Called ... with the following non-sleepable locks held`" mean?

This means that a function that may sleep was called while a mutex (or other unsleepable) lock was held.

The reason this is an error is because mutexes are not intended to be held for long periods of time; they are supposed to only be held to maintain short periods of synchronization. This programming contract allows device drivers to use mutexes to synchronize with the rest of the kernel during interrupts. Interrupts (under FreeBSD) may not sleep. Hence it is imperative that no subsystem in the kernel block for an extended period while holding a mutex.

To catch such errors, assertions may be added to the kernel that interact with the witness(4) subsystem to emit a warning or fatal error (depending on the system configuration) when a potentially blocking call is made while holding a mutex.

In summary, such warnings are non-fatal, however with unfortunate timing they could cause undesirable effects ranging from a minor blip in the system's responsiveness to a complete system lockup.

33. Why does `buildworld`/`installworld` die with the message "`touch: not found`"?

This error does not mean that the touch(1) utility is missing. The error is instead probably due to the dates of the files being set sometime in the future. If your CMOS-clock is set to local time you need to run the command `adjkerntz -i` to adjust the kernel clock when booting into single user mode.

Chapter 6 Commercial Applications

> **Note:** This section is still very sparse, though we are hoping, of course, that companies will add to it! :) The FreeBSD group has no financial interest in any of the companies listed here but simply lists them as a public service (and feels that commercial interest in FreeBSD can have very positive effects on FreeBSD's long-term viability). We encourage commercial software vendors to send their entries here for inclusion. See the Vendors page (http://www.FreeBSD.org/commercial/index.html) for a longer list.

1. Where can I get an Office Suite for FreeBSD?

The open-source **OpenOffice.org (http://www.openoffice.org)** office suite works natively on FreeBSD. The Linux version of **Oracle Open Office (http://www.oracle.com/us/products/applications/open-office/index.html)**, the value-added closed-source version of OpenOffice.org, also works on FreeBSD.

FreeBSD also includes a variety of text editors, spreadsheets, and drawing programs in the Ports Collection.

2. Where can I get **Motif®** for FreeBSD?

The Open Group has released the source code to **Motif 2.2.2**. You can install the `x11-toolkits/open-motif` package, or compile it from ports. Refer to the ports section of the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/ports.html) for more information on how to do this.

> **Note:** The **Open Motif** distribution only allows redistribution if it is running on an open source (http://www.opensource.org/) operating system.

In addition, there are commercial distributions of the **Motif** software available. These, however, are not for free, but their license allows them to be used in closed-source software. Contact Apps2go for the least expensive ELF **Motif 2.1.20** distribution for FreeBSD (i386).

There are two distributions, the "development edition" and the "runtime edition" (for much less). These distributions includes:

- **OSF/Motif manager**, **xmbind**, **panner**, **wsm**.
- Development kit with uil, mrm, xm, xmcxx, include and **Imake** files.
- Static and dynamic ELF libraries.
- Demonstration applets.

Be sure to specify that you want the FreeBSD version of **Motif** when ordering (do not forget to mention the architecture you want too)! Versions for NetBSD and OpenBSD are also sold by *Apps2go*. This is currently a FTP only download.

More info

Apps2go WWW page (http://www.apps2go.com/)

or

<sales@apps2go.com> or <support@apps2go.com>

or

phone (817) 431 8775 or +1 817 431-8775

3. Where can I get **CDE** for FreeBSD?

Xi Graphics used to sell **CDE** for FreeBSD, but no longer do.

KDE (http://www.kde.org/) is an open source X11 desktop which is similar to **CDE** in many respects. You might also like the look and feel of **xfce** (http://www.xfce.org/). **KDE** and **xfce** are both in the ports system (http://www.FreeBSD.org/ports/index.html).

4. Are there any Database systems for FreeBSD?

Yes! See the Commercial Vendors (http://www.FreeBSD.org/commercial/software_bycat.html#CATEGORY_DATABASE) section of FreeBSD's Web site.

Also see the Databases (http://www.FreeBSD.org/ports/databases.html) section of the Ports Collection.

5. Can I run **Oracle®** on FreeBSD?

Yes. Instructions on how to set up Linux **Oracle** on FreeBSD can be found under http://www.shadowcom.net/freebsd-oracle9i/.

Chapter 7 User Applications

1. So, where are all the user applications?

Please take a look at the ports page (http://www.FreeBSD.org/ports/index.html) for info on software packages ported to FreeBSD. The list currently tops 20,000 and is growing daily, so come back to check often or subscribe to the FreeBSD announcements mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-announce) for periodic updates on new entries.

Most ports should work on the 6.*X*, 7.*X* and 8.*X* branches. Each time a FreeBSD release is made, a snapshot of the ports tree at the time of release in also included in the `ports/` directory.

We also support the concept of a "package", essentially no more than a compressed binary distribution with a little extra intelligence embedded in it for doing whatever custom installation work is required. A package can be installed and uninstalled again easily without having to know the gory details of which files it includes.

Use the Packages package installation menu in sysinstall(8) (under the Configure menu item) or invoke the pkg_add(1) command on the specific package files you are interested in installing. Package files can usually be identified by their `.tbz` suffix and CD-ROM distribution people will have a `packages/All` directory on their CD which contains such files. They can also be downloaded over the net for various versions of FreeBSD at the following locations:

for 6.*X*-RELEASE/6-STABLE

> ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-6-stable
> (ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-6-stable/)

for 7.*X*-RELEASE/7-STABLE

> ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-7-stable
> (ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-7-stable/)

for 8.*X*-RELEASE/8-STABLE

> ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-8-stable
> (ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-8-stable/)

for 9-CURRENT

> ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-9-current
> (ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/i386/packages-9-current/)

or your nearest local mirror site.

Note that all ports may not be available as packages since new ones are constantly being added. It is always a good idea to check back periodically to see which packages are available at the ftp.FreeBSD.org (ftp://ftp.FreeBSD.org/pub/FreeBSD/) master site.

2. How do I configure INN (Internet News) for my machine?

After installing the `news/inn` package or port, an excellent place to start are the INN FAQ (http://www.eyrie.org/~eagle/faqs/inn.html).

3. Does FreeBSD support Java™?

Yes. Please see http://www.FreeBSD.org/java/ (http://www.FreeBSD.org/java/index.html).

4. Why can I not build this port on my 6.*X*, 7.*X* or 8.*X*-STABLE machine?

If you are running a FreeBSD version that lags significantly behind *-CURRENT* or *-STABLE*, you may need to update your Ports Collection; see the Keeping Up (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/porters-handbook/keeping-up.html) section of the Porter's Handbook for further information on how to do this. If you are up to date, then someone might have committed a change to the port which works for *-CURRENT* but which broke the port for *-STABLE*. Please submit a bug report on this with the send-pr(1) command, since the Ports Collection is supposed to work for both the *-CURRENT* and *-STABLE* branches.

5. I just tried to build `INDEX` using `make index`, and it failed. Why?

First, always make sure that you have a completely up-to-date Ports Collection. Errors that affect building `INDEX` from an up-to-date copy of the Ports Collection are high-visibility and are thus almost always fixed immediately.

However, if you are up-to-date, perhaps you are seeing another problem. `make index` has a known bug in dealing with incomplete copies of the Ports Collection. It assumes that you have a local copy of every single port that every other port that you have a local copy of depends on. To explain, if you have a copy of `foo/bar` on your disk, and `foo/bar` depends on `baz/quux`, then you must also have a copy of `baz/quux` on your disk, and the ports `baz/quux` depends on, and so on. Otherwise, `make index` has insufficient information to create its dependency tree.

This is particularly a problem for FreeBSD users who utilize cvsup(1) (or csup(1)) to track the Ports Collection but choose not to install certain categories by specifying them in `refuse`. In theory, one should be able to refuse categories, but in practice there are too many ports that depend on ports in other categories. Until someone comes up with a solution for this problem, the general rule is that if you want to build `INDEX`, you must have a complete copy of the Ports Collection.

There are rare cases where `INDEX` will not build due to odd cases involving `WITH_*` or `WITHOUT_*` variables being set in `make.conf`. If you suspect that this is the case, please try to make `INDEX` with those make variables turned off before reporting it to FreeBSD ports mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-ports).

6. Why is **CVSup** not integrated in the main FreeBSD tree?

The FreeBSD base system is designed as self-hosting — it should be possible to build the whole operating system starting with a very limited set of tools. Thus, the actual build tools needed to compile the FreeBSD sources are bundled with the sources themselves. This includes a C compiler (gcc(1)), make(1), awk(1), and similar tools.

Since **CVSup** is written in Modula-3, adding it to the FreeBSD base system would also require adding and maintaining a Modula-3 compiler. This would lead to both an increase in the disk space consumed by the FreeBSD sources and additional maintenance work. Thus, it is much easier for both the developers and users to keep **CVSup** as a separate port, which can be easily installed as a package bundled on the FreeBSD installation CDs.

However, FreeBSD users are not without an integrated **CVSup** compatible client anymore since FreeBSD 6.2-RELEASE. Thanks to Maxime Henrion <`mux@FreeBSD.org`>, **CVSup** was rewritten in C as csup(1) and it is the part of the base system by now. Although it does not implement all the features of **CVSup** at the moment, it is good enough (and really fast!) to keep your sources synchronized. For systems earlier than 6.2, it can be installed as a port or package (see `net/csup`).

7. I updated the sources, now how do I update my installed ports?

FreeBSD does not include a port upgrading tool, but it does have some tools to make the upgrade process somewhat easier. You can also install additional tools to simplify port handling, see the Upgrading Ports (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/ports-using.html) section in the FreeBSD Handbook.

8. Do I need to recompile every port each time I perform a major version update?

By all means! While a recent system will run with software compiled under an older release, you will end up with things randomly crashing and failing to work once you start installing other ports or updating a portion of what you already have.

When the system is upgraded, various shared libraries, loadable modules, and other parts of the system will be replaced with newer versions. Applications linked against the older versions may fail to start or, in other cases, fail to function properly.

For more information, see the section on upgrades (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/updating-upgrading-freebsdupdate.html#FREEBSDUPDATE-UPGRADE) in the FreeBSD Handbook.

9. Do I need to recompile every port each time I perform a minor version update?

In general, no. FreeBSD developers do their utmost to guarantee binary compatibility across all releases with the same major version number. Any exceptions will be documented in the Release Notes, and advice given there should be followed.

10. Why is `/bin/sh` so minimal? Why does FreeBSD not use `bash` or another shell?

Because POSIX® says that there shall be such a shell.

The more complicated answer: many people need to write shell scripts which will be portable across many systems. That is why POSIX specifies the shell and utility commands in great detail. Most scripts are written in Bourne shell, and because several important programming interfaces (make(1), system(3), popen(3), and analogues in higher-level scripting languages like Perl and Tcl) are specified to use the Bourne shell to interpret commands. Because the Bourne shell is so often and widely used, it is important for it to be quick to start, be deterministic in its behavior, and have a small memory footprint.

The existing implementation is our best effort at meeting as many of these requirements simultaneously as we can. In order to keep `/bin/sh` small, we have not provided many of the convenience features that other shells have. That is why the Ports Collection includes more featureful shells like `bash`, `scsh`, `tcsh`, and `zsh`. (You can compare for yourself the memory utilization of all these shells by looking at the "VSZ" and "RSS" columns in a `ps -u` listing.)

11. Why do **Netscape** and **Opera** take so long to start?

The usual answer is that DNS on your system is misconfigured. Both **Netscape** and **Opera** perform DNS checks when starting up. The browser will not appear on your desktop until the program either gets a response or determines that the system has no network connection.

12. I updated parts of the Ports Collection using **CVSup**, and now many ports fail to build with mysterious error messages! What happened? Is the Ports Collection broken in some major way?

If you only update parts of the Ports Collection, using one of its **CVSup** subcollections and not the `ports-all` **CVSup** collection, you should *always* update the `ports-base` subcollection too! The reasons are described in the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/cvsup.html#CVSUP-COLLEC-PBASE-WARN).

13. How do I create audio CDs from my MIDI files?

To create audio CDs from MIDI files, first install `audio/timidity++` from ports then install manually the GUS patches set by Eric A. Welsh, available at http://alleg.sourceforge.net/digmid.html. After **TiMidity++** has been installed properly, MIDI files may be converted to WAV files with the following command line:

```
% timidity -Ow -s 44100 -o /tmp/juke/01.wav 01.mid
```

The WAV files can then be converted to other formats or burned onto audio CDs, as described in the FreeBSD Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html).

Chapter 8 Kernel Configuration

1. I would like to customize my kernel. Is it difficult?

Not at all! Check out the kernel config section of the Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig.html).

> **Note:** The new `kernel` will be installed to the `/boot/kernel` directory along with its modules, while the old kernel and its modules will be moved to the `/boot/kernel.old` directory, so if you make a mistake the next time you play with your configuration you can boot the previous version of your kernel.

2. My kernel compiles fail because `_hw_float` is missing. How do I solve this problem?

You probably removed `npx0` (see npx(4)) from your kernel configuration file because you do not have a math co-processor. The `npx0` device is *MANDATORY*. Somewhere inside your hardware lies a device that provides hardware floating-point support, even if it is no longer a separate device as used in the good old 386 days. You *must* include the `npx0` device. Even if you manage to build a kernel without `npx0` support, it will not boot anyway.

3. Why is my kernel so big (over 10 MB)?

Chances are, you compiled your kernel in *debug mode*. Kernels built in debug mode contain many symbols that are used for debugging, thus greatly increasing the size of the kernel. Note that there will be little or no performance decrease from running a debug kernel, and it is useful to keep one around in case of a system panic.

However, if you are running low on disk space, or you simply do not want to run a debug kernel, make sure that both of the following are true:

- You do not have a line in your kernel configuration file that reads:

  ```
  makeoptions DEBUG=-g
  ```

- You are not running config(8) with the `-g` option.

Either of the above settings will cause your kernel to be built in debug mode. As long as you make sure you follow the steps above, you can build your kernel normally, and you should notice a fairly large size decrease; most kernels tend to be around 1.5 MB to 2 MB.

4. Why do I get interrupt conflicts with multi-port serial code?

When I compile a kernel with multi-port serial code, it tells me that only the first port is probed and the rest skipped due to interrupt conflicts. How do I fix this?

The problem here is that FreeBSD has code built-in to keep the kernel from getting trashed due to hardware or software conflicts. The way to fix this is to leave out the IRQ settings on all but one port. Here is an example:

```
#
# Multiport high-speed serial line - 16550 UARTS
#
device sio2 at isa? port 0x2a0 tty irq 5 flags 0x501 vector siointr
```

```
device sio3 at isa? port 0x2a8 tty flags 0x501 vector siointr
device sio4 at isa? port 0x2b0 tty flags 0x501 vector siointr
device sio5 at isa? port 0x2b8 tty flags 0x501 vector siointr
```

5. Why does every kernel I try to build fail to compile, even `GENERIC`?

There are a number of possible causes for this problem. They are, in no particular order:

- You are not using the `make buildkernel` and `make installkernel` targets, and your source tree is different from the one used to build the currently running system (e.g., you are compiling 8.1-RELEASE on a 7.3-RELEASE system). If you are attempting an upgrade, please read the `/usr/src/UPDATING` file, paying particular attention to the "COMMON ITEMS" section at the end.
- You are using the `make buildkernel` and `make installkernel` targets, but you failed to assert the completion of the `make buildworld` target. The `make buildkernel` target relies on files generated by the `make buildworld` target to complete its job correctly.
- Even if you are trying to build FreeBSD-STABLE, it is possible that you fetched the source tree at a time when it was either being modified, or broken for other reasons; only releases are absolutely guaranteed to be buildable, although FreeBSD-STABLE builds fine the majority of the time. If you have not already done so, try re-fetching the source tree and see if the problem goes away. Try using a different server in case the one you are using is having problems.

6. How can I verify which scheduler is in use on a running system?

Check for the existence of the `kern.sched.quantum` sysctl. If you have it, you should see something like this:

```
% sysctl kern.sched.quantum
kern.sched.quantum: 99960
```

If the `kern.sched.quantum` sysctl exists, you are using the 4BSD scheduler (sched_4bsd(4)). If not, you will get an error printed by sysctl(8) (which you can safely ignore):

```
% sysctl kern.sched.quantum
sysctl: unknown oid 'kern.sched.quantum'
```

The name of the scheduler currently being used is directly available as the value of the `kern.sched.name` sysctl:

```
% sysctl kern.sched.name
kern.sched.name: 4BSD
```

7. What is `kern.sched.quantum`?

`kern.sched.quantum` is the maximum number of ticks a process can run without being preempted. It is specific to the 4BSD scheduler, so you can use its presence or absence to determine which scheduler is in use.

Chapter 9 Disks, File Systems, and Boot Loaders

1. How can I add my new hard disk to my FreeBSD system?

See the Adding Disks (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/disks-adding.html) section in the FreeBSD Handbook.

2. How do I move my system over to my huge new disk?

The best way is to reinstall the OS on the new disk, then move the user data over. This is highly recommended if you have been tracking *-STABLE* for more than one release, or have updated a release instead of installing a new one. You can install booteasy on both disks with boot0cfg(8), and dual boot them until you are happy with the new configuration. Skip the next paragraph to find out how to move the data after doing this.

Should you decide not to do a fresh install, you need to partition and label the new disk with either sysinstall(8), or fdisk(8) and disklabel(8). You should also install booteasy on both disks with boot0cfg(8), so that you can dual boot to the old or new system after the copying is done. See the formatting-media article (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/formatting-media/index.html) for details on this process.

Now you have the new disk set up, and are ready to move the data. Unfortunately, you cannot just blindly copy the data. Things like device files (in `/dev`), flags, and links tend to screw that up. You need to use tools that understand these things, which means dump(8). Although it is suggested that you move the data in single user mode, it is not required.

You should never use anything but dump(8) and restore(8) to move the root file system. The tar(1) command may work — then again, it may not. You should also use dump(8) and restore(8) if you are moving a single partition to another empty partition. The sequence of steps to use `dump` to move a partitions data to a new partition is:

1. `newfs` the new partition.
2. `mount` it on a temporary mount point.
3. `cd` to that directory.
4. `dump` the old partition, piping output to the new one.

For example, if you are going to move root to `/dev/`*`ad1s1a`*, with *`/mnt`* as the temporary mount point, it is:

```
# newfs /dev/ad1s1a
# mount /dev/ad1s1a /mnt
# cd /mnt
# dump 0af - / | restore rf -
```

Rearranging your partitions with `dump` takes a bit more work. To merge a partition like `/var` into its parent, create the new partition large enough for both, move the parent partition as described above, then move the child partition into the empty directory that the first move created:

```
# newfs /dev/ad1s1a
# mount /dev/ad1s1a /mnt
# cd /mnt
# dump 0af - / | restore rf -
```

```
# cd var
# dump 0af - /var | restore rf -
```

To split a directory from its parent, say putting `/var` on its own partition when it was not before, create both partitions, then mount the child partition on the appropriate directory in the temporary mount point, then move the old single partition:

```
# newfs /dev/ad1s1a
# newfs /dev/ad1s1d
# mount /dev/ad1s1a /mnt
# mkdir /mnt/var
# mount /dev/ad1s1d /mnt/var
# cd /mnt
# dump 0af - / | restore rf -
```

You might prefer cpio(1), pax(1), tar(1) to dump(8) for user data. At the time of this writing, these are known to lose file flag information, so use them with caution.

3. Will a "dangerously dedicated" disk endanger my health?

The installation procedure allows you to chose two different methods in partitioning your hard disk(s). The default way makes it compatible with other operating systems on the same machine, by using fdisk(8) table entries (called "slices" in FreeBSD), with a FreeBSD slice that employs partitions of its own. Optionally, one can chose to install a boot-selector to switch between the possible operating systems on the disk(s). The alternative uses the entire disk for FreeBSD, and makes no attempt to be compatible with other operating systems.

So why it is called "dangerous"? A disk in this mode does not contain what normal PC utilities would consider a valid fdisk(8) table. Depending on how well they have been designed, they might complain at you once they are getting in contact with such a disk, or even worse, they might damage the BSD bootstrap without even asking or notifying you. In addition, the "dangerously dedicated" disk's layout is known to confuse many BIOSes, including those from AWARD (e.g. as found in HP Netserver and Micronics systems as well as many others) and Symbios/NCR (for the popular 53C8xx range of SCSI controllers). This is not a complete list, there are more. Symptoms of this confusion include the "`read error`" message printed by the FreeBSD bootstrap when it cannot find itself, as well as system lockups when booting.

Why have this mode at all then? It only saves a few kbytes of disk space, and it can cause real problems for a new installation. "Dangerously dedicated" mode's origins lie in a desire to avoid one of the most common problems plaguing new FreeBSD installers — matching the BIOS "geometry" numbers for a disk to the disk itself.

"Geometry" is an outdated concept, but one still at the heart of the PC's BIOS and its interaction with disks. When the FreeBSD installer creates slices, it has to record the location of these slices on the disk in a fashion that corresponds with the way the BIOS expects to find them. If it gets it wrong, you will not be able to boot.

"Dangerously dedicated" mode tries to work around this by making the problem simpler. In some cases, it gets it right. But it is meant to be used as a last-ditch alternative — there are better ways to solve the problem 99 times out of 100.

So, how do you avoid the need for "DD" mode when you are installing? Start by making a note of the geometry that your BIOS claims to be using for your disks. You can arrange to have the kernel print this as it boots by specifying `-v` at the `boot:` prompt, or using `boot -v` in the loader. Just before the installer starts, the kernel will print a list of BIOS geometries. Do not panic — wait for the installer to start and then use scrollback to read the numbers. Typically the BIOS disk units will be in the same order that FreeBSD lists your disks, first IDE, then SCSI.

When you are slicing up your disk, check that the disk geometry displayed in the FDISK screen is correct (ie. it matches the BIOS numbers); if it is wrong, use the **G** key to fix it. You may have to do this if there is absolutely nothing on the disk, or if the disk has been moved from another system. Note that this is only an issue with the disk that you are going to boot from; FreeBSD will sort itself out just fine with any other disks you may have.

Once you have got the BIOS and FreeBSD agreeing about the geometry of the disk, your problems are almost guaranteed to be over, and with no need for "DD" mode at all. If, however, you are still greeted with the dreaded "`read error`" message when you try to boot, it is time to cross your fingers and go for it — there is nothing left to lose.

To return a "dangerously dedicated" disk for normal PC use, there are basically two options. The first is, you write enough NULL bytes over the MBR to make any subsequent installation believe this to be a blank disk. You can do this for example with the following command:

```
# dd if=/dev/zero of=/dev/rda0 count=15
```

Alternatively, the undocumented DOS "feature"

```
C:\> fdisk /mbr
```

will to install a new master boot record as well, thus clobbering the BSD bootstrap.

4. Which partitions can safely use Soft Updates? I have heard that Soft Updates on `/` can cause problems.

Short answer: you can usually use Soft Updates safely on all partitions.

Long answer: There used to be some concern over using Soft Updates on the root partition. Soft Updates has two characteristics that caused this. First, a Soft Updates partition has a small chance of losing data during a system crash. (The partition will not be corrupted; the data will simply be lost.) Also, Soft Updates can cause temporary space shortages.

When using Soft Updates, the kernel can take up to thirty seconds to actually write changes to the physical disk. If you delete a large file, the file still resides on disk until the kernel actually performs the deletion. This can cause a very simple race condition. Suppose you delete one large file and immediately create another large file. The first large file is not yet actually removed from the physical disk, so the disk might not have enough room for the second large file. You get an error that the partition does not have enough space, although you know perfectly well that you just released a large chunk of space! When you try again mere seconds later, the file creation works as you expect. This has left more than one user scratching his head and doubting his sanity, the FreeBSD file system, or both.

If a system should crash after the kernel accepts a chunk of data for writing to disk, but before that data is actually written out, data could be lost or corrupted. This risk is extremely small, but generally manageable. Use of IDE write caching greatly increases this risk; it is strongly recommended that you disable IDE write caching when using Soft Updates.

These issues affect all partitions using Soft Updates. So, what does this mean for the root partition?

Vital information on the root partition changes very rarely. Files such as `/boot/kernel/kernel` and the contents of `/etc` only change during system maintenance, or when users change their passwords. If the system crashed during the thirty-second window after such a change is made, it is possible that data could be lost. This risk is negligible for most applications, but you should be aware that it exists. If your system cannot tolerate this much risk, do not use Soft Updates on the root file system!

`/` is traditionally one of the smallest partitions. If you put the `/tmp` directory on `/` and you have a busy `/tmp`, you might see intermittent space problems. Symlinking `/tmp` to `/var/tmp` will solve this problem.

5. What is inappropriate about my ccd(4)?

The symptom of this is:

```
# ccdconfig -C
ccdconfig: ioctl (CCDIOCSET): /dev/ccd0c: Inappropriate file type or format
```

This usually happens when you are trying to concatenate the `c` partitions, which default to type `unused`. The ccd(4) driver requires the underlying partition type to be `FS_BSDFFS`. Edit the disk label of the disks you are trying to concatenate and change the types of partitions to `4.2BSD`.

6. Why can I not edit the disk label on my ccd(4)?

The symptom of this is:

```
# disklabel ccd0
(it prints something sensible here, so let us try to edit it)
# disklabel -e ccd0
(edit, save, quit)
disklabel: ioctl DIOCWDINFO: No disk label on disk;
use "disklabel -r" to install initial label
```

This is because the disk label returned by ccd(4) is actually a “fake” one that is not really on the disk. You can solve this problem by writing it back explicitly, as in:

```
# disklabel ccd0 > /tmp/disklabel.tmp
# disklabel -Rr ccd0 /tmp/disklabel.tmp
# disklabel -e ccd0
(this will work now)
```

7. Can I mount other foreign file systems under FreeBSD?

FreeBSD supports a variety of other file systems.

UFS

> UFS CD-ROMs can be mounted directly on FreeBSD. Mounting disk partitions from Digital UNIX and other systems that support UFS may be more complex, depending on the details of the disk partitioning for the operating system in question.

ext2/ext3

> FreeBSD supports `ext2fs` and `ext3fs` partitions. See mount_ext2fs(8) for more information.

NTFS

FreeBSD includes a read-only NTFS driver. For more information, see mount_ntfs(8). A port of **ntfs-3g** (http://www.tuxera.com/community/) supports write operations on NTFS (see `sysutils/fusefs-ntfs`).

FAT

FreeBSD includes a read-write FAT driver. For more information, see mount_msdosfs(8).

ReiserFS

FreeBSD includes a read-only ReiserFS driver. For more information, see mount_reiserfs(8).

ZFS

As of this writing, FreeBSD includes a port of Sun™'s ZFS driver. The current recommendation is to use it only on amd64 platforms with sufficient memory. For more information, see zfs(8).

FreeBSD also supports network file systems such as NFS (see mount_nfs(8)), NetWare (see mount_nwfs(8)), and Microsoft-style SMB file systems (see mount_smbfs(8)). You can find ports based on FUSE (`sysutils/fusefs-kmod`) for many other file systems.

8. How do I mount a secondary DOS partition?

The secondary DOS partitions are found after *all* the primary partitions. For example, if you have an "E" partition as the second DOS partition on the second SCSI drive, there will be a device file for "slice 5" in `/dev`, so simply mount it:

```
# mount -t msdosfs /dev/da1s5 /dos/e
```

9. Is there a cryptographic file system for FreeBSD?

Yes. You can use either gbde(8) or geli(8), see the Encrypting Disk Partitions (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/disks-encrypting.html) section of the FreeBSD Handbook.

10. How can I use the Windows NT® loader to boot FreeBSD?

The general idea is that you copy the first sector of your native root FreeBSD partition into a file in the DOS/Windows NT partition. Assuming you name that file something like `c:\bootsect.bsd` (inspired by `c:\bootsect.dos`), you can then edit the `c:\boot.ini` file to come up with something like this:

```
[boot loader]
timeout=30
default=multi(0)disk(0)rdisk(0)partition(1)\WINDOWS
[operating systems]
multi(0)disk(0)rdisk(0)partition(1)\WINDOWS="Windows NT"
C:\BOOTSECT.BSD="FreeBSD"
C:\="DOS"
```

If FreeBSD is installed on the same disk as the Windows NT boot partition simply copy `/boot/boot1` to `C:\BOOTSECT.BSD`. However, if FreeBSD is installed on a different disk `/boot/boot1` will not work, `/boot/boot0` is needed.

`/boot/boot0` needs to be installed using sysinstall(8) by selecting the FreeBSD boot manager on the screen which asks if you wish to use a boot manager. This is because `/boot/boot0` has the partition table area filled with NULL characters but sysinstall(8) copies the partition table before copying `/boot/boot0` to the MBR.

> **Warning:** *Do not simply copy `/boot/boot0` instead of `/boot/boot1`; you will overwrite your partition table and render your computer un-bootable!*

When the FreeBSD boot manager runs it records the last OS booted by setting the active flag on the partition table entry for that OS and then writes the whole 512-bytes of itself back to the MBR so if you just copy `/boot/boot0` to `C:\BOOTSECT.BSD` then it writes an empty partition table, with the active flag set on one entry, to the MBR.

11. How do I boot FreeBSD and Linux from LILO?

If you have FreeBSD and Linux on the same disk, just follow LILO's installation instructions for booting a non-Linux operating system. Very briefly, these are:

Boot Linux, and add the following lines to `/etc/lilo.conf`:

```
other=/dev/hda2
        table=/dev/hda
        label=FreeBSD
```

(the above assumes that your FreeBSD slice is known to Linux as `/dev/hda2`; tailor to suit your setup). Then, run `lilo` as `root` and you should be done.

If FreeBSD resides on another disk, you need to add `loader=/boot/chain.b` to the LILO entry. For example:

```
other=/dev/dab4
        table=/dev/dab
        loader=/boot/chain.b
        label=FreeBSD
```

In some cases you may need to specify the BIOS drive number to the FreeBSD boot loader to successfully boot off the second disk. For example, if your FreeBSD SCSI disk is probed by BIOS as BIOS disk 1, at the FreeBSD boot loader prompt you need to specify:

```
Boot: 1:da(0,a)/boot/kernel/kernel
```

You can configure boot(8) to automatically do this for you at boot time.

The Linux+FreeBSD mini-HOWTO (http://tldp.org/HOWTO/Linux+FreeBSD.html) is a good reference for FreeBSD and Linux interoperability issues.

12. How do I boot FreeBSD and Linux using GRUB?

Booting FreeBSD using GRUB is very simple. Just add the following to your configuration file `/boot/grub/menu.lst` (or `/boot/grub/grub.conf` in some systems, e.g. Red Hat Linux and its derivatives).

```
title FreeBSD 6.1
root (hd0,a)
kernel /boot/loader
```

Where *hd0,a* points to your root partition on the first disk. If you need to specify which slice number should be used, use something like this *(hd0,2,a)*. By default, if the slice number is omitted, GRUB searches the first slice which has `a` partition.

13. How do I boot FreeBSD and Linux using **BootEasy?**

Install LILO at the start of your Linux boot partition instead of in the Master Boot Record. You can then boot LILO from **BootEasy**.

If you are running Windows and Linux this is recommended anyway, to make it simpler to get Linux booting again if you should need to reinstall Windows (which is a Jealous Operating System, and will bear no other Operating Systems in the Master Boot Record).

14. How do I change the boot prompt from `???` to something more meaningful?

You can not do that with the standard boot manager without rewriting it. There are a number of other boot managers in the `sysutils` ports category that provide this functionality.

15. I have a new removable drive, how do I use it?

Whether it is a removable drive like a Zip or an EZ drive (or even a floppy, if you want to use it that way), or a new hard disk, once it is installed and recognized by the system, and you have your cartridge/floppy/whatever slotted in, things are pretty much the same for all devices.

(this section is based on Mark Mayo's ZIP FAQ (http://www.vmunix.com/mark/FreeBSD/ZIP-FAQ.html))

If it is a ZIP drive or a floppy, you have already got a DOS file system on it, you can use a command like this:

```
# mount -t msdosfs /dev/fd0c /floppy
```

if it is a floppy, or this:

```
# mount -t msdosfs /dev/da2s4 /zip
```

for a ZIP disk with the factory configuration.

For other disks, see how they are laid out using fdisk(8) or sysinstall(8).

The rest of the examples will be for a ZIP drive on `da2`, the third SCSI disk.

Unless it is a floppy, or a removable you plan on sharing with other people, it is probably a better idea to stick a BSD file system on it. You will get long filename support, at least a 2X improvement in performance, and a lot more stability. First, you need to redo the DOS-level partitions/file systems. You can either use fdisk(8) or sysinstall(8), or

for a small drive that you do not want to bother with multiple operating system support on, just blow away the whole FAT partition table (slices) and just use the BSD partitioning:

```
# dd if=/dev/zero of=/dev/rda2 count=2
# disklabel -Brw da2 auto
```

You can use disklabel(8) or sysinstall(8) to create multiple BSD partitions. You will certainly want to do this if you are adding swap space on a fixed disk, but it is probably irrelevant on a removable drive like a ZIP.

Finally, create a new file system, this one is on our ZIP drive using the whole disk:

```
# newfs /dev/rda2c
```

and mount it:

```
# mount /dev/da2c /zip
```

and it is probably a good idea to add a line like this to `/etc/fstab` (see fstab(5)) so you can just type `mount /zip` in the future:

```
/dev/da2c /zip ffs rw,noauto 0 0
```

16. Why do I get "`Incorrect super block`" when mounting a CD-ROM?

You have to tell mount(8) the type of the device that you want to mount. This is described in the Handbook section on optical media (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html), specifically the section Using Data CDs (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html#MOUNTING-CD).

17. Why do I get "`Device not configured`" when mounting a CD-ROM?

This generally means that there is no CD-ROM in the CD-ROM drive, or the drive is not visible on the bus. Please see the Using Data CDs (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html#MOUNTING-CD) section of the Handbook for a detailed discussion of this issue.

18. Why do all non-English characters in filenames show up as "?" on my CDs when mounted in FreeBSD?

Your CD-ROM probably uses the "Joliet" extension for storing information about files and directories. This is discussed in the Handbook chapter on creating and using CD-ROMs (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html), specifically the section on Using Data CD-ROMs (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html#MOUNTING-CD).

19. I burned a CD under FreeBSD and now I can not read it under any other operating system. Why?

You most likely burned a raw file to your CD, rather than creating an ISO 9660 file system. Take a look at the Handbook chapter on creating CD-ROMs (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html), particularly the section on burning raw data CDs (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html#RAWDATA-CD).

20. How can I create an image of a data CD?

This is discussed in the Handbook section on duplicating data CDs (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html#IMAGING-CD). For more on working with CD-ROMs, see the Creating CDs Section (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/creating-cds.html) in the Storage chapter in the Handbook.

21. Why can I not `mount` an audio CD?

If you try to mount an audio CD, you will get an error like "`cd9660: /dev/acd0c: Invalid argument`". This is because `mount` only works on file systems. Audio CDs do not have file systems; they just have data. You need a program that reads audio CDs, such as the `audio/xmcd` port.

22. How do I `mount` a multi-session CD?

By default, mount(8) will attempt to mount the last data track (session) of a CD. If you would like to load an earlier session, you must use the `-s` command line argument. Please see mount_cd9660(8) for specific examples.

23. How do I let ordinary users mount floppies, CD-ROMs and other removable media?

Ordinary users can be permitted to mount devices. Here is how:

1. As `root` set the sysctl variable `vfs.usermount` to `1`.

   ```
   # sysctl -w vfs.usermount=1
   ```

2. As `root` assign the appropriate permissions to the block device associated with the removable media.

   For example, to allow users to mount the first floppy drive, use:

   ```
   # chmod 666 /dev/fd0
   ```

   To allow users in the group `operator` to mount the CD-ROM drive, use:

   ```
   # chgrp operator /dev/acd0c
   # chmod 640 /dev/acd0c
   ```

3. You will need to alter `/etc/devfs.conf` to make these changes permanent across reboots.

   As `root`, add the necessary lines to `/etc/devfs.conf`. For example, to allow users to mount the first floppy drive add:

   ```
   # Allow all users to mount the floppy disk.
   own       /dev/fd0   root:operator
   perm      /dev/fd0   0666
   ```

   To allow users in the group `operator` to mount the CD-ROM drive add:

   ```
   # Allow members of the group operator to mount CD-ROMs.
   own       /dev/acd0   root:operator
   perm      /dev/acd0   0660
   ```

4. Finally, add the line `vfs.usermount=1` to the file `/etc/sysctl.conf` so that it is reset at system boot time.

All users can now mount the floppy `/dev/fd0` onto a directory that they own:

```
% mkdir ~/my-mount-point
% mount -t msdosfs /dev/fd0 ~/my-mount-point
```

Users in group `operator` can now mount the CD-ROM `/dev/acd0c` onto a directory that they own:

```
% mkdir ~/my-mount-point
% mount -t cd9660 /dev/acd0c ~/my-mount-point
```

Unmounting the device is simple:

```
% umount ~/my-mount-point
```

Enabling `vfs.usermount`, however, has negative security implications. A better way to access MS-DOS formatted media is to use the `emulators/mtools` package in the Ports Collection.

> **Note:** The device name used in the previous examples must be changed according to your configuration.

24. The `du` and `df` commands show different amounts of disk space available. What is going on?

You need to understand what `du` and `df` really do. `du` goes through the directory tree, measures how large each file is, and presents the totals. `df` just asks the file system how much space it has left. They seem to be the same thing, but a file without a directory entry will affect `df` but not `du`.

When a program is using a file, and you delete the file, the file is not really removed from the file system until the program stops using it. The file is immediately deleted from the directory listing, however. You can see this easily enough with a program such as `more`. Assume you have a file large enough that its presence affects the output of `du` and `df`. (Since disks can be so large today, this might be a *very* large file!) If you delete this file while using `more` on it, `more` does not immediately choke and complain that it cannot view the file. The entry is simply removed from the directory so no other program or user can access it. `du` shows that it is gone — it has walked the directory tree and the file is not listed. `df` shows that it is still there, as the file system knows that `more` is still using that space. Once you end the `more` session, `du` and `df` will agree.

Note that Soft Updates can delay the freeing of disk space; you might need to wait up to 30 seconds for the change to be visible!

This situation is common on web servers. Many people set up a FreeBSD web server and forget to rotate the log files. The access log fills up `/var`. The new administrator deletes the file, but the system still complains that the partition is full. Stopping and restarting the web server program would free the file, allowing the system to release the disk space. To prevent this from happening, set up newsyslog(8).

25. How can I add more swap space?

In the Configuration and Tuning (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/config-tuning.html) section of the Handbook, you will find a section (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/adding-swap-space.html) describing how to do this.

26. Why does FreeBSD see my disk as smaller than the manufacturer says it is?

Disk manufacturers calculate gigabytes as a billion bytes each, whereas FreeBSD calculates them as 1,073,741,824 bytes each. This explains why, for example, FreeBSD's boot messages will report a disk that supposedly has 80 GB as holding 76,319 MB.

Also note that FreeBSD will (by default) reserve 8% of the disk space.

27. How is it possible for a partition to be more than 100% full?

A portion of each UFS partition (8%, by default) is reserved for use by the operating system and the `root` user. df(1) does not count that space when calculating the `Capacity` column, so it can exceed 100%. Also, you will notice that the `Blocks` column is always greater than the sum of the `Used` and `Avail` columns, usually by a factor of 8%.

For more details, look up the `-m` option in tunefs(8).

Chapter 10 System Administration

1. Where are the system start-up configuration files?

The primary configuration file is `/etc/defaults/rc.conf` (see rc.conf(5)). System startup scripts such as `/etc/rc` and `/etc/rc.d` (see rc(8)) just include this file. *Do not edit this file!* Instead, if there is any entry in `/etc/defaults/rc.conf` that you want to change, you should copy the line into `/etc/rc.conf` and change it there.

For example, if you wish to start named(8), the included DNS server, all you need to do is:

```
# echo 'named_enable="YES"' >> /etc/rc.conf
```

To start up local services, place shell scripts in the `/usr/local/etc/rc.d` directory. These shell scripts should be set executable, the default file mode is `555`.

2. How do I add a user easily?

Use the adduser(8) command, or the pw(8) command for more complicated situations.

To remove the user, use the rmuser(8) command or, if necessary, pw(8).

3. Why do I keep getting messages like "`root: not found`" after editing my `crontab` file?

This is normally caused by editing the system crontab (`/etc/crontab`) and then using crontab(1) to install it:

```
# crontab /etc/crontab
```

This is not the correct way to do things. The system crontab has a different format to the per-user crontabs which crontab(1) updates (the crontab(5) manual page explains the differences in more detail).

If this is what you did, the extra crontab is simply a copy of `/etc/crontab` in the wrong format it. Delete it with the command:

```
# crontab -r
```

Next time, when you edit `/etc/crontab`, you should not do anything to inform cron(8) of the changes, since it will notice them automatically.

If you want something to be run once per day, week, or month, it is probably better to add shell scripts `/usr/local/etc/periodic`, and let the periodic(8) command run from the system `cron` schedule it with the other periodic system tasks.

The actual reason for the error is that the system crontab has an extra field, specifying which user to run the command as. In the default system crontab provided with FreeBSD, this is `root` for all entries. When this crontab is used as the `root` user's crontab (which is *not* the same as the system crontab), cron(8) assumes the string `root` is the first word of the command to execute, but no such command exists.

4. Why do I get the error, "`you are not in the correct group to su root`" when I try to `su` to `root`?

This is a security feature. In order to `su` to `root` (or any other account with superuser privileges), you must be in the `wheel` group. If this feature were not there, anybody with an account on a system who also found out `root`'s password would be able to gain superuser level access to the system. With this feature, this is not strictly true; su(1) will prevent them from even trying to enter the password if they are not in `wheel`.

To allow someone to `su` to `root`, simply put them in the `wheel` group.

5. I made a mistake in `rc.conf`, or another startup file, and now I cannot edit it because the file system is read-only. What should I do?

Restart the system using **`boot -s`** at the loader prompt to enter Single User mode. When prompted for a shell pathname, simply press **Enter**, and run `mount -urw /` to re-mount the root file system in read/write mode. You may also need to run `mount -a -t ufs` to mount the file system where your favorite editor is defined. If your favorite editor is on a network file system, you will need to either configure the network manually before you can mount network file systems, or use an editor which resides on a local file system, such as ed(1).

If you intend to use a full screen editor such as vi(1) or emacs(1), you may also need to run `export TERM=cons25` so that these editors can load the correct data from the termcap(5) database.

Once you have performed these steps, you can edit `/etc/rc.conf` as you usually would to fix the syntax error. The error message displayed immediately after the kernel boot messages should tell you the number of the line in the file which is at fault.

6. Why am I having trouble setting up my printer?

See the Handbook entry on printing (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/printing.html). It should cover most of your problem.

Some printers require a host-based driver to do any kind of printing. These so-called "WinPrinters" are not natively supported by FreeBSD. If your printer does not work in DOS or Windows, it is probably a WinPrinter. Your only hope of getting one of these to work is to check if the `print/pnm2ppa` port supports it.

7. How can I correct the keyboard mappings for my system?

Please see the Handbook section on using localization (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/using-localization.html), specifically the section on console setup (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/using-localization.html#SETTING-CONSOLE).

8. Why do I get messages like: "`unknown: <PNP0303> can't assign resources`" on boot?

The following is an excerpt from a post to the FreeBSD-CURRENT mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-current).

> The "can't assign resources" messages indicate that the devices are legacy ISA devices for which a non-PnP-aware driver is compiled into the kernel. These include devices such as keyboard controllers, the programmable interrupt controller chip, and several other bits of standard infrastructure. The resources cannot be assigned because there is already a driver using those addresses.

—Garrett Wollman <wollman@FreeBSD.org>, 24 April 2001

9. Why can I not get user quotas to work properly?

1. It is possible that your kernel is not configured to use quotas. If this is the case, you will need to add the following line to your kernel configuration file and recompile:

   ```
   options QUOTA
   ```

   Please read the Handbook entry on quotas (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/quotas.html) for full details.

2. Do not turn on quotas on `/`.

3. Put the quota file on the file system that the quotas are to be enforced on, i.e.:

File System	Quota file
/usr	/usr/admin/quotas
/home	/home/admin/quotas
...	...

10. Does FreeBSD support System V IPC primitives?

Yes, FreeBSD supports System V-style IPC, including shared memory, messages and semaphores, in the `GENERIC` kernel. In a custom kernel, enable this support by adding the following lines to your kernel config.

```
options     SYSVSHM          # enable shared memory
options     SYSVSEM          # enable for semaphores
options     SYSVMSG          # enable for messaging
```

Recompile and install your kernel.

11. What other mail-server software can I use instead of **sendmail**?

The **sendmail** (http://www.sendmail.org/) server is the default mail-server software for FreeBSD, but you can easily replace it with one of the other MTA (for instance, an MTA installed from the ports).

There are various alternative MTAs in the ports tree already, with `mail/exim`, `mail/postfix`, `mail/qmail`, and `mail/zmailer` being some of the most popular choices.

Diversity is nice, and the fact that you have many different mail-servers to chose from is considered a good thing; therefore try to avoid asking questions like "Is **sendmail** better than **qmail**?" in the mailing lists. If you do feel like asking, first check the mailing list archives. The advantages and disadvantages of each and every one of the available MTAs have already been discussed a few times.

12. I have forgotten the `root` password! What do I do?

Do not panic! Restart the system, type **`boot -s`** at the `Boot:` prompt to enter Single User mode. At the question about the shell to use, hit **Enter**. You will be dropped to a `#` prompt. Enter `mount -urw /` to remount your root file

system read/write, then run `mount -a` to remount all the file systems. Run `passwd root` to change the `root` password then run exit(1) to continue booting.

> **Note:** If you are still prompted to give the `root` password when entering the Single User mode, it means that the console has been marked as `insecure` in `/etc/ttys`. In this case it will be required to boot from an FreeBSD installation disk, choose the Fixit shell from sysinstall(8) and issue the commands mentioned above.

> **Note:** If you cannot mount your root partition from Single User mode, it is possible that the partitions are encrypted and it is impossible to mount them without the access keys. Your chances are depending on the chosen implementation. For more information see the section about encrypted disks in the FreeBSD Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/disks-encrypting.html).

13. How do I keep **Control+Alt+Delete** from rebooting the system?

If you are using syscons(4) (the default console driver) build and install a new kernel with the line in the configuration file:

```
options SC_DISABLE_REBOOT
```

This can also be done by setting the following sysctl(8) which does not require a reboot or kernel recompile:

```
# sysctl hw.syscons.kbd_reboot=0
```

> **Note:** The above two methods are exclusive: The sysctl(8) does not exist if you compile your kernel with the `SC_DISABLE_REBOOT` option.

If you use the pcvt(4) console driver, use the following kernel configuration line instead and rebuild the kernel:

```
options PCVT_CTRL_ALT_DEL
```

14. How do I reformat DOS text files to UNIX ones?

Use this perl(1) command:

```
% perl -i.bak -npe 's/\r\n/\n/g' file(s)
```

where *`file(s)`* is one or more files to process. The modification is done in-place, with the original file stored with a `.bak` extension.

Alternatively you can use the tr(1) command:

```
% tr -d '\r' < dos-text-file > unix-file
```

`dos-text-file` is the file containing DOS text while *`unix-file`* will contain the converted output. This can be quite a bit faster than using `perl`.

Yet another way to reformat DOS text files is to use the `converters/dosunix` port from the Ports Collection. Consult its documentation about the details.

15. How do I kill processes by name?

Use killall(1).

16. Why is su(1) bugging me about not being in `root`'s ACL?

The error comes from the **Kerberos** distributed authentication system. The problem is not fatal but annoying. You can either run su with the `-K` option, or uninstall **Kerberos** as described in the next question.

17. How do I uninstall **Kerberos**?

To remove **Kerberos** from the system, reinstall the `base` distribution for the release you are running. If you have the CD-ROM, you can mount it (we will assume on `/cdrom`) and run the commands below:

```
# cd /cdrom/base
# ./install.sh
```

Alternately, you can include the `NO_KERBEROS` option in your `/etc/make.conf` and rebuild world.

18. What happened to `/dev/MAKEDEV`?

FreeBSD 5.*X* and beyond use the devfs(8) device-on-demand system. Device drivers automatically create new device nodes as they are needed, obsoleting `/dev/MAKEDEV`.

19. How do I add pseudoterminals to the system?

If you have a lot of `telnet`, `ssh`, X, or `screen` users, you might run out of pseudoterminals. By default, FreeBSD 6.2 and earlier support 256 pseudoterminals, while FreeBSD 6.3 and later support 512 pseudoterminals.

> **Tip:** If needed, more pseudoterminals can be added. However, this requires patching the standard C library, the kernel, and `/etc/ttys`. For example, http://www.freebsd.org/~jhb/patches/pty_1152.patch expands the number of pseudoterminals to 1152. Note that the patch will only apply cleanly to FreeBSD 6.3 or later.

20. How do I re-read `/etc/rc.conf` and re-start `/etc/rc` without a reboot?

Go into single user mode and then back to multi user mode.

On the console do:

```
# shutdown now
(Note: without -r or -h)

# return
# exit
```

21. I tried to update my system to the latest *-STABLE*, but got *-BETA*`x`, *-RC* or *-PRERELEASE*! What is going on?

Short answer: it is just a name. *RC* stands for "Release Candidate". It signifies that a release is imminent. In FreeBSD, *-PRERELEASE* is typically synonymous with the code freeze before a release. (For some releases, the *-BETA* label was used in the same way as *-PRERELEASE*.)

Long answer: FreeBSD derives its releases from one of two places. Major, dot-zero, releases, such as 7.0-RELEASE and 8.0-RELEASE, are branched from the head of the development stream, commonly referred to as -CURRENT. Minor releases, such as 6.3-RELEASE or 5.2-RELEASE, have been snapshots of the active -STABLE branch. Starting with 4.3-RELEASE, each release also now has its own branch which can be tracked by people requiring an extremely conservative rate of development (typically only security advisories).

When a release is about to be made, the branch from which it will be derived from has to undergo a certain process. Part of this process is a code freeze. When a code freeze is initiated, the name of the branch is changed to reflect that it is about to become a release. For example, if the branch used to be called 6.2-STABLE, its name will be changed to 6.3-PRERELEASE to signify the code freeze and signify that extra pre-release testing should be happening. Bug fixes can still be committed to be part of the release. When the source code is in shape for the release the name will be changed to 6.3-RC to signify that a release is about to be made from it. Once in the RC stage, only the most critical bugs found can be fixed. Once the release (6.3-RELEASE in this example) and release branch have been made, the branch will be renamed to 6.3-STABLE.

For more information on version numbers and the various CVS branches, refer to the Release Engineering (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/releng/article.html) article.

22. I tried to install a new kernel, and the chflags(1) failed. How do I get around this?

Short answer: You are probably at security level greater than 0. Reboot directly to Single User mode to install the kernel.

Long answer: FreeBSD disallows changing system flags at security levels greater than 0. You can check your security level with the command:

```
# sysctl kern.securelevel
```

You cannot lower the security level; you have to boot to Single Mode to install the kernel, or change the security level in `/etc/rc.conf` then reboot. See the init(8) manual page for details on `securelevel`, and see `/etc/defaults/rc.conf` and the rc.conf(5) manual page for more information on `rc.conf`.

23. I cannot change the time on my system by more than one second! How do I get around this?

Short answer: You are probably at security level greater than 1. Reboot directly to Single User mode to change the date.

Long answer: FreeBSD disallows changing the time by more that one second at security levels greater than 1. You can check your security level with the command:

```
# sysctl kern.securelevel
```

You cannot lower the security level; you have to boot to Single User mode to change the date, or change the security level in `/etc/rc.conf` then reboot. See the init(8) manual page for details on `securelevel`, and see `/etc/defaults/rc.conf` and the rc.conf(5) manual page for more information on `rc.conf`.

24. Why is `rpc.statd` using 256 MB of memory?

No, there is no memory leak, and it is not using 256 MB of memory. For convenience, `rpc.statd` maps an obscene amount of memory into its address space. There is nothing terribly wrong with this from a technical standpoint; it just throws off things like top(1) and ps(1).

rpc.statd(8) maps its status file (resident on `/var`) into its address space; to save worrying about remapping it later when it needs to grow, it maps it with a generous size. This is very evident from the source code, where one can see that the length argument to mmap(2) is `0x10000000`, or one sixteenth of the address space on an IA32, or exactly 256 MB.

25. Why can I not unset the `schg` file flag?

You are running at an elevated (i.e., greater than 0) securelevel. Lower the securelevel and try again. For more information, see the FAQ entry on securelevel and the init(8) manual page.

26. Why does **SSH** authentication through `.shosts` not work by default in recent versions of FreeBSD?

The reason why `.shosts` authentication does not work by default in more recent versions of FreeBSD is because ssh(1) is not installed suid `root` by default. To "fix" this, you can do one of the following:

- As a permanent fix, set `ENABLE_SUID_SSH` to `true` in `/etc/make.conf` then rebuild and install ssh(1) (or run `make world`).
- As a temporary fix, change the mode on `/usr/bin/ssh` to `4555` by running `chmod 4555 /usr/bin/ssh` as `root`. Then add `ENABLE_SUID_SSH= true` to `/etc/make.conf` so the change takes effect the next time `make world` is run.

27. What is `vnlru`?

`vnlru` flushes and frees vnodes when the system hits the `kern.maxvnodes` limit. This kernel thread sits mostly idle, and only activates if you have a huge amount of RAM and are accessing tens of thousands of tiny files.

28. What do the various memory states displayed by `top` mean?

- `Active`: pages recently statistically used.
- `Inactive`: pages recently statistically unused.
- `Cache`: (most often) pages that have percolated from inactive to a status where they maintain their data, but can often be immediately reused (either with their old association, or reused with a new association). There can be certain immediate transitions from `active` to `cache` state if the page is known to be clean (unmodified), but that transition is a matter of policy, depending upon the algorithm choice of the VM system maintainer.
- `Free`: pages without data content, and can be immediately used in certain circumstances where cache pages might be ineligible. Free pages can be reused at interrupt or process state.
- `Wired`: pages that are fixed into memory, usually for kernel purposes, but also sometimes for special use in processes.

Pages are most often written to disk (sort of a VM sync) when they are in the inactive state, but active pages can also be synced. This depends upon the CPU tracking of the modified bit being available, and in certain situations there can

be an advantage for a block of VM pages to be synced, whether they are active or inactive. In most common cases, it is best to think of the inactive queue to be a queue of relatively unused pages that might or might not be in the process of being written to disk. Cached pages are already synced, not mapped, but available for immediate process use with their old association or with a new association. Free pages are available at interrupt level, but cached or free pages can be used at process state for reuse. Cache pages are not adequately locked to be available at interrupt level.

There are some other flags (e.g., busy flag or busy count) that might modify some of the described rules.

29. How much free memory is available?

There are a couple of kinds of "free memory". One kind is the amount of memory immediately available without paging anything else out. That is approximately the size of cache queue + size of free queue (with a derating factor, depending upon system tuning). Another kind of "free memory" is the total amount of VM space. That can be complex, but is dependent upon the amount of swap space and memory. Other kinds of "free memory" descriptions are also possible, but it is relatively useless to define these, but rather it is important to make sure that the paging rate is kept low, and to avoid running out of swap space.

30. What is `/var/empty`? I can not delete it!

`/var/empty` is a directory that the sshd(8) program uses when performing privilege separation. The `/var/empty` directory is empty, owned by `root` and has the `schg` flag set.

Although it is not recommended to delete this directory, to do so you will need to unset the `schg` flag first. See the chflags(1) manual page for more information (and bear in mind the answer to the question on unsetting the schg flag).

Chapter 11 The X Window System and Virtual Consoles

1. What is the X Window System?

The X Window System (commonly `X11`) is the most widely available windowing system capable of running on UNIX or UNIX like systems, including FreeBSD. The X.Org Foundation (http://www.x.org/wiki/) administers the X protocol standards (http://en.wikipedia.org/wiki/X_Window_System_core_protocol), with the current reference implementation, version 11 release 7.5, so you will often see references shortened to `X11`.

Many implementations are available for different architectures and operating systems. An implementation of the server-side code is properly known as an `X server`.

2. Which X implementations are available for FreeBSD?

Historically, the default implementation of X on FreeBSD has been XFree86™ which is maintained by The XFree86 Project, Inc. (http://www.xfree86.org) This software was installed by default on FreeBSD versions up until 4.10 and 5.2. Although Xorg itself maintained an implementation during that time period, it was basically only provided as a reference platform, as it had suffered greatly from bitrot over the years.

However, early in 2004, some XFree86 developers left that project over issues including the pace of code changes, future directions, and interpersonal conflicts, and are now contributing code directly to Xorg instead. At that time, Xorg updated its source tree to the last XFree86 release before its subsequent licensing change (**XFree86 version 4.3.99.903**), incorporated many changes that had previously been maintained separately, and has released that software as **X11R6.7.0**. A separate but related project, freedesktop.org (http://www.freedesktop.org/wiki/) (or `fd.o` for short), is working on rearchitecting the original XFree86 code to offload more work onto the graphics cards (with the goal of increased performance) and make it more modular (with the goal of increased maintainability, and thus faster releases as well as easier configuration). Xorg intends to incorporate the `freedesktop.org` changes in its future releases.

As of July 2004, in FreeBSD-CURRENT, XFree86 has been replaced with Xorg as the default implementation. Since then the default X11 implementation in FreeBSD is Xorg.

For further information, read the X11 (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/x11.html) section of the FreeBSD Handbook.

3. Why did the X projects split, anyway?

The answer to this question is outside the scope of this FAQ. Note that there are voluminous postings in various mailing list archives on the Internet; please use your favorite search engine to investigate the history instead of asking this question on the FreeBSD mailing lists. It may even be the case that only the participants will ever know for certain.

4. Why did FreeBSD choose to go with the Xorg ports by default?

The Xorg developers claimed that their goal is to release more often and incorporate new features more quickly. If they are able to do so, this will be very attractive. Also, their software still uses the traditional X license, while XFree86 is using their modified one.

5. I want to run X, how do I go about it?

If you would like to add X to an existing installation, you should use either the `x11/xorg` meta-port, which will build and install all the necessary components, or install Xorg from FreeBSD packages:

```
# pkg_add -r xorg
```

It is also possible to install Xorg from sysinstall(8) by choosing Configure, then Distributions, then The X.Org Distribution.

After the installation of Xorg was successful, follow the instructions from the X11 Configuration (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/x-config.html) section of the FreeBSD Handbook.

6. I *tried* to run X, but I get an "`KDENABIO failed (Operation not permitted)`" error when I type `startx`. What do I do now?

Your system is probably running at a raised `securelevel`. It is not possible to start X at a raised `securelevel` because X requires write access to io(4). For more information, see at the init(8) manual page.

So the question is what else you should do instead, and you basically have two choices: set your `securelevel` back down to zero (usually from `/etc/rc.conf`), or run xdm(1) at boot time (before the `securelevel` is raised).

See Q: 12. for more information about running xdm(1) at boot time.

7. Why does my mouse not work with X?

If you are using syscons(4) (the default console driver), you can configure FreeBSD to support a mouse pointer on each virtual screen. In order to avoid conflicting with X, syscons(4) supports a virtual device called `/dev/sysmouse`. All mouse events received from the real mouse device are written to the sysmouse(4) device via moused(8). If you wish to use your mouse on one or more virtual consoles, *and* use X, see Q: 4. and set up moused(8).

Then edit `/etc/X11/xorg.conf` and make sure you have the following lines:

```
Section "InputDevice"
   Option          "Protocol" "SysMouse"
   Option          "Device" "/dev/sysmouse"
.....
```

Starting with Xorg version 7.4, the `InputDevice` sections in `xorg.conf` are ignored in favor of autodetected devices. To restore the old behavior, add the following line to the `ServerLayout` or `ServerFlags` section:

```
Option "AutoAddDevices" "false"
```

Some people prefer to use `/dev/mouse` under X. To make this work, `/dev/mouse` should be linked to `/dev/sysmouse` (see sysmouse(4)) by adding the following line to `/etc/devfs.conf` (see devfs.conf(5)):

```
link    sysmouse    mouse
```

This link can be created by restarting devfs(5) with the following command (as `root`):

```
# /etc/rc.d/devfs restart
```

8. My mouse has a fancy wheel. Can I use it in X?

Yes.

You need to tell X that you have a 5 button mouse. To do this, simply add the lines `Buttons 5` and `ZAxisMapping 4 5` to the "InputDevice" section of `/etc/X11/xorg.conf`. For example, you might have the following "InputDevice" section in `/etc/X11/xorg.conf`.

Example 11-1. "InputDevice" Section for Wheeled Mouse in Xorg Configuration File

```
Section "InputDevice"
   Identifier      "Mouse1"
   Driver          "mouse"
   Option          "Protocol" "auto"
   Option          "Device" "/dev/sysmouse"
   Option          "Buttons" "5"
   Option          "ZAxisMapping" "4 5"
EndSection
```

Example 11-2. ".emacs" Example for Naive Page Scrolling with Wheeled Mouse (optional)

```
;; wheel mouse
(global-set-key [mouse-4] 'scroll-down)
(global-set-key [mouse-5] 'scroll-up)
```

9. How do I use remote X displays?

For security reasons, the default setting is to not allow a machine to remotely open a window.

To enable this feature, simply start **X** with the optional `-listen_tcp` argument:

```
% startx -listen_tcp
```

10. What is a virtual console and how do I make more?

Virtual consoles, put simply, enable you to have several simultaneous sessions on the same machine without doing anything complicated like setting up a network or running X.

When the system starts, it will display a login prompt on the monitor after displaying all the boot messages. You can then type in your login name and password and start working (or playing!) on the first virtual console.

At some point, you will probably wish to start another session, perhaps to look at documentation for a program you are running or to read your mail while waiting for an FTP transfer to finish. Just do **Alt+F2** (hold down the **Alt** key and press the **F2** key), and you will find a login prompt waiting for you on the second "virtual console"! When you want to go back to the original session, do **Alt+F1**.

The default FreeBSD installation has eight virtual consoles enabled. **Alt+F1**, **Alt+F2**, **Alt+F3**, and so on will switch between these virtual consoles.

To enable more of them, edit `/etc/ttys` (see ttys(5)) and add entries for `ttyv8` to `ttyvc` after the comment on "Virtual terminals":

```
# Edit the existing entry for ttyv8 in /etc/ttys and change
# "off" to "on".
ttyv8   "/usr/libexec/getty Pc"         cons25  on secure
ttyv9   "/usr/libexec/getty Pc"         cons25  on secure
ttyva   "/usr/libexec/getty Pc"         cons25  on secure
ttyvb   "/usr/libexec/getty Pc"         cons25  on secure
```

Use as many or as few as you want. The more virtual terminals you have, the more resources that are used; this can be important if you have 8 MB RAM or less. You may also want to change the `secure` to `insecure`.

> **Important:** If you want to run an X server you *must* leave at least one virtual terminal unused (or turned off) for it to use. That is to say that if you want to have a login prompt pop up for all twelve of your Alt-function keys, you are out of luck — you can only do this for eleven of them if you also want to run an X server on the same machine.

The easiest way to disable a console is by turning it off. For example, if you had the full 12 terminal allocation mentioned above and you wanted to run X, you would change settings for virtual terminal 12 from:

```
ttyvb   "/usr/libexec/getty Pc"         cons25  on  secure
```

to:

```
ttyvb   "/usr/libexec/getty Pc"         cons25  off secure
```

If your keyboard has only ten function keys, you would end up with:

```
ttyv9   "/usr/libexec/getty Pc"         cons25  off secure
ttyva   "/usr/libexec/getty Pc"         cons25  off secure
ttyvb   "/usr/libexec/getty Pc"         cons25  off secure
```

(You could also just delete these lines.)

Next, the easiest (and cleanest) way to activate the virtual consoles is to reboot. However, if you really do not want to reboot, you can just shut down the X Window system and execute (as `root`):

```
# kill -HUP 1
```

It is imperative that you completely shut down X Window if it is running, before running this command. If you do not, your system will probably appear to hang or lock up after executing the `kill` command.

11. How do I access the virtual consoles from X?

Use **Ctrl+Alt+F*n*** to switch back to a virtual console. **Ctrl+Alt+F1** would return you to the first virtual console.

Once you are back to a text console, you can then use **Alt+F*n*** as normal to move between them.

To return to the X session, you must switch to the virtual console running X. If you invoked X from the command line, (e.g., using `startx`) then the X session will attach to the next unused virtual console, not the text console from which it was invoked. If you have eight active virtual terminals then X will be running on the ninth, and you would use **Alt+F9** to return.

12. How do I start **XDM** on boot?

There are two schools of thought on how to start xdm(1). One school starts `xdm` from `/etc/ttys` (see ttys(5)) using the supplied example, while the other simply runs `xdm` from from `rc.local` (see rc(8)) or from an `X` script in `/usr/local/etc/rc.d`. Both are equally valid, and one may work in situations where the other does not. In both cases the result is the same: X will pop up a graphical login prompt.

The ttys(5) method has the advantage of documenting which vty X will start on and passing the responsibility of restarting the X server on logout to init(8). The rc(8) method makes it easy to `kill xdm` if there is a problem starting the X server.

If loaded from rc(8), `xdm` should be started without any arguments (i.e., as a daemon). The `xdm` command must start *after* getty(8) runs, or else `getty` and `xdm` will conflict, locking out the console. The best way around this is to have the script sleep 10 seconds or so then launch `xdm`.

If you are to start `xdm` from `/etc/ttys`, there still is a chance of conflict between `xdm` and getty(8). One way to avoid this is to add the `vt` number in the `/usr/local/lib/X11/xdm/Xservers` file:

```
:0 local /usr/local/bin/X vt4
```

The above example will direct the X server to run in `/dev/ttyv3`. Note the number is offset by one. The X server counts the vty from one, whereas the FreeBSD kernel numbers the vty from zero.

13. Why do I get "`Couldn't open console`" when I run `xconsole`?

If you start **X** with `startx`, the permissions on `/dev/console` will *not* get changed, resulting in things like `xterm -C` and `xconsole` not working.

This is because of the way console permissions are set by default. On a multi-user system, one does not necessarily want just any user to be able to write on the system console. For users who are logging directly onto a machine with a VTY, the fbtab(5) file exists to solve such problems.

In a nutshell, make sure an uncommented line of the form is in `/etc/fbtab` (see fbtab(5)):

```
/dev/ttyv0 0600 /dev/console
```

It will ensure that whomever logs in on `/dev/ttyv0` will own the console.

14. Before, I was able to run XFree86 as a regular user. Why does it now say that I must be `root`?

All X servers need to be run as `root` in order to get direct access to your video hardware. Older versions of XFree86 (<= 3.3.6) installed all bundled servers to be automatically run as `root` (setuid to `root`). This is obviously a security hazard because X servers are large, complicated programs. Newer versions of XFree86 do not install the servers setuid to `root` for just this reason.

Obviously, running an X server as the `root` user is not acceptable, nor a good idea security-wise. There are two ways to be able to use X as a regular user. The first is to use `xdm` or another display manager (e.g., `kdm`); the second is to use the `Xwrapper`.

`xdm` is a daemon that handles graphical logins. It is usually started at boot time, and is responsible for authenticating users and starting their sessions; it is essentially the graphical counterpart of getty(8) and login(1). For more

information on `xdm` see the XFree86 documentation (http://www.xfree86.org/sos/resources.html), and the the FAQ entry on it.

`Xwrapper` is the X server wrapper; it is a small utility to enable one to manually run an X server while maintaining reasonable safety. It performs some sanity checks on the command line arguments given, and if they pass, runs the appropriate X server. If you do not want to run a display manager for whatever reason, this is for you. If you have installed the complete Ports Collection, you can find the port in `x11/wrapper`.

15. Why does my PS/2 mouse misbehave under X?

Your mouse and the mouse driver may have somewhat become out of synchronization.

In rare cases the driver may erroneously report synchronization problem and you may see the kernel message:

```
psmintr: out of sync (xxxx != yyyy)
```

and notice that your mouse does not work properly.

If this happens, disable the synchronization check code by setting the driver flags for the PS/2 mouse driver to `0x100`. Enter *UserConfig* by giving the `-c` option at the boot prompt:

```
boot: -c
```

Then, in the *UserConfig* command line, type:

```
UserConfig> flags psm0 0x100
UserConfig> quit
```

16. Why does my PS/2 mouse from MouseSystems not work?

There have been some reports that certain model of PS/2 mouse from MouseSystems works only if it is put into the "high resolution" mode. Otherwise, the mouse cursor may jump to the upper-left corner of the screen every so often.

Specify the flags `0x04` to the PS/2 mouse driver to put the mouse into the high resolution mode. Enter *UserConfig* by giving the `-c` option at the boot prompt:

```
boot: -c
```

Then, in the *UserConfig* command line, type:

```
UserConfig> flags psm0 0x04
UserConfig> quit
```

See the previous section for another possible cause of mouse problems.

17. How do I reverse the mouse buttons?

Run the command `xmodmap -e "pointer = 3 2 1"` from your `.xinitrc` or `.xsession`.

18. How do I install a splash screen and where do I find them?

The detailed answer for this question can be found in the Boot Time Splash Screens (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/boot-blocks.html#BOOT-SPLASH) section of the FreeBSD Handbook.

19. Can I use the **Windows** keys on my keyboard in X?

Yes. All you need to do is use xmodmap(1) to define what function you wish them to perform.

Assuming all "Windows" keyboards are standard then the keycodes for these three keys are the following:

- 115 — **Windows** key, between the left-hand **Ctrl** and **Alt** keys
- 116 — **Windows** key, to the right of the **AltGr** key
- 117 — **Menu** key, to the left of the right-hand **Ctrl** key

To have the left **Windows** key print a comma, try this.

```
# xmodmap -e "keycode 115 = comma"
```

You will probably have to re-start your window manager to see the result.

To have the **Windows** key-mappings enabled automatically every time you start X either put the `xmodmap` commands in your `~/.xinitrc` file or, preferably, create a file `~/.xmodmaprc` and include the `xmodmap` options, one per line, then add the following line to your `~/.xinitrc`:

```
xmodmap $HOME/.xmodmaprc
```

For example, you could map the 3 keys to be **F13**, **F14**, and **F15**, respectively. This would make it easy to map them to useful functions within applications or your window manager, as demonstrated further down.

To do this put the following in `~/.xmodmaprc`.

```
keycode 115 = F13
keycode 116 = F14
keycode 117 = F15
```

If you use the `x11-wm/fvwm2` port, for example, you could map the keys so that **F13** iconifies (or de-iconifies) the window the cursor is in, **F14** brings the window the cursor is in to the front or, if it is already at the front, pushes it to the back, and **F15** pops up the main Workplace (application) menu even if the cursor is not on the desktop, which is useful if you do not have any part of the desktop visible (and the logo on the key matches its functionality).

The following entries in `~/.fvwmrc` implement the aforementioned setup:

```
Key F13         FTIWS    A         Iconify
Key F14         FTIWS    A         RaiseLower
Key F15         A        A         Menu Workplace Nop
```

20. How can I get 3D hardware acceleration for OpenGL®?

The availability of 3D acceleration depends on the version of Xorg that you are using and the type of video chip you have. If you have an nVidia chip, you can use the binary drivers provided for FreeBSD by installing one of the following ports:

- The latest versions of nVidia cards are supported by the `x11/nvidia-driver` port.
- nVidia cards like the GeForce2 MX/3/4 series are supported by the 96XX series of drivers, available in the `x11/nvidia-driver-96xx` port.
- Even older cards, like GeForce and RIVA TNT are supported by the 71XX series of drivers, available in the `x11/nvidia-driver-71xx` port.

In fact, nVidia provides detailed information on which card is supported by which driver. This information is available directly on their web site: http://www.nvidia.com/object/IO_32667.html.

For Matrox G200/G400, you should check the `x11-servers/mga_hal` port.

For ATI Rage 128 and Radeon, see the ati(4), r128(4) and radeon(4) manual pages.

For 3dfx Voodoo 3, 4, 5, and Banshee cards, there is a `x11-servers/driglide` port.

Chapter 12 Networking

1. Where can I get information on “diskless booting”?

“Diskless booting” means that the FreeBSD box is booted over a network, and reads the necessary files from a server instead of its hard disk. For full details, please read the Handbook entry on diskless booting (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-diskless.html)

2. Can a FreeBSD box be used as a dedicated network router?

Yes. Please see the Handbook entry on advanced networking (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/advanced-networking.html), specifically the section on routing and gateways (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-routing.html).

3. Can I connect my Windows box to the Internet via FreeBSD?

Typically, people who ask this question have two PCs at home, one with FreeBSD and one with some version of Windows the idea is to use the FreeBSD box to connect to the Internet and then be able to access the Internet from the Windows box through the FreeBSD box. This is really just a special case of the previous question and works perfectly well.

If you are using dialup to connect to the Internet user-mode ppp(8) contains a `-nat` option. If you run ppp(8) with the `-nat` option, set `gateway_enable` to *YES* in `/etc/rc.conf`, and configure your Windows machine correctly, this should work fine. For more information, please see the ppp(8) manual page or the Handbook entry on user PPP (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/userppp.html).

If you are using kernel-mode PPP or have an Ethernet connection to the Internet, you need to use natd(8). Please look at the natd (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-natd.html) section of the Handbook for a tutorial.

4. Does FreeBSD support SLIP and PPP?

Yes. See the manual pages for slattach(8), sliplogin(8), ppp(8), and pppd(8). ppp(8) and pppd(8) provide support for both incoming and outgoing connections, while sliplogin(8) deals exclusively with incoming connections, and slattach(8) deals exclusively with outgoing connections.

For more information on how to use these, please see the Handbook chapter on PPP and SLIP (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/ppp-and-slip.html).

If you only have access to the Internet through a “shell account”, you may want to have a look at the `net/slirp` package. It can provide you with (limited) access to services such as ftp and http direct from your local machine.

5. Does FreeBSD support NAT or Masquerading?

Yes. If you want to use NAT over a user PPP connection, please see the Handbook entry on user PPP (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/userppp.html). If you want to use NAT over some other sort of network connection, please look at the natd (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-natd.html) section of the Handbook.

6. How do I connect two FreeBSD systems over a parallel line using PLIP?

Please see the PLIP section (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-plip.html) of the Handbook.

7. How can I set up Ethernet aliases?

If the alias is on the same subnet as an address already configured on the interface, then add `netmask 0xffffffff` to your ifconfig(8) command-line, as in the following:

```
# ifconfig ed0 alias 192.0.2.2 netmask 0xffffffff
```

Otherwise, just specify the network address and netmask as usual:

```
# ifconfig ed0 alias 172.16.141.5 netmask 0xffffff00
```

You can read more about this in the FreeBSD Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/configtuning-virtual-hosts.html).

8. How do I get my 3C503 to use the other network port?

If you want to use the other ports, you will have to specify an additional parameter on the ifconfig(8) command line. The default port is `link0`. To use the AUI port instead of the BNC one, use `link2`. These flags should be specified using the ifconfig_* variables in `/etc/rc.conf` (see rc.conf(5)).

9. Why am I having trouble with NFS and FreeBSD?

Certain PC network cards are better than others (to put it mildly) and can sometimes cause problems with network intensive applications like NFS.

See the Handbook entry on NFS (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-nfs.html) for more information on this topic.

10. Why can I not NFS-mount from a Linux box?

Some versions of the Linux NFS code only accept mount requests from a privileged port; try to issue the following command:

```
# mount -o -P linuxbox:/blah /mnt
```

11. Why can I not NFS-mount from a Sun box?

Sun workstations running SunOS™ 4.*X* only accept mount requests from a privileged port; try the following command:

```
# mount -o -P sunbox:/blah /mnt
```

12. Why does `mountd` keep telling me it "`can't change attributes`" and that I have a "`bad exports list`" on my FreeBSD NFS server?

The most frequent problem is not understanding the correct format of `/etc/exports`. Please review exports(5) and the NFS (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-nfs.html) entry in the Handbook, especially the section on configuring NFS (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/network-nfs.html#CONFIGURING-NFS).

13. Why am I having problems talking PPP to NeXTStep machines?

Try disabling the TCP extensions in `/etc/rc.conf` (see rc.conf(5)) by changing the following variable to `NO`:

```
tcp_extensions=NO
```

Xylogic's Annex boxes are also broken in this regard and you must use the above change to connect through them.

14. How do I enable IP multicast support?

FreeBSD supports multicast host operations by default. If you want your box to run as a multicast router, you need to recompile your kernel with the `MROUTING` option and run mrouted(8). FreeBSD will start mrouted(8) at boot time if the flag `mrouted_enable` is set to `YES` in `/etc/rc.conf`.

> **Note:** In recent FreeBSD releases, the mrouted(8) multicast routing daemon, the map-mbone(8) and mrinfo(8) utilities have been removed from the base system. These programs are now available in the FreeBSD Ports Collection as `net/mrouted`.

MBONE tools are available in their own ports category, mbone (http://www.FreeBSD.org/ports/mbone.html). If you are looking for the conference tools `vic` and `vat`, look there!

15. Which network cards are based on the DEC PCI chipset?

Here is a list compiled by Glen Foster <`gfoster@driver.nsta.org`>, with some more modern additions:

Table 12-1. Network Cards Based on the DEC PCI Chipset

Vendor	Model
ASUS	PCI-L101-TB
Accton	ENI1203
Cogent	EM960PCI
Compex	ENET32-PCI
D-Link	DE-530
Dayna	DP1203, DP2100
DEC	DE435, DE450
Danpex	EN-9400P3
JCIS	Condor JC1260
Linksys	EtherPCI

Vendor	Model
Mylex	LNP101
SMC	EtherPower 10/100 (Model 9332)
SMC	EtherPower (Model 8432)
TopWare	TE-3500P
Znyx (2.2.x)	ZX312, ZX314, ZX342, ZX345, ZX346, ZX348
Znyx (3.x)	ZX345Q, ZX346Q, ZX348Q, ZX412Q, ZX414, ZX442, ZX444, ZX474, ZX478, ZX212, ZX214 (10mbps/hd)

16. Why do I have to use the FQDN for hosts on my site?

See the answer in the FreeBSD Handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/mail-trouble.html).

17. Why do I get an error, "`Permission denied`", for all networking operations?

If you have compiled your kernel with the `IPFIREWALL` option, you need to be aware that the default policy is to deny all packets that are not explicitly allowed.

If you had unintentionally misconfigured your system for firewalling, you can restore network operability by typing the following while logged in as `root`:

```
# ipfw add 65534 allow all from any to any
```

You can also set `firewall_type="open"` in `/etc/rc.conf`.

For further information on configuring a FreeBSD firewall, see the Handbook chapter (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/firewalls.html).

18. Why is my `ipfw` "fwd" rule to redirect a service to another machine not working?

Possibly because you want to do network address translation (NAT) and not just forward packets. A "fwd" rule does exactly what it says; it forwards packets. It does not actually change the data inside the packet. Say we have a rule like:

```
01000 fwd 10.0.0.1 from any to foo 21
```

When a packet with a destination address of *`foo`* arrives at the machine with this rule, the packet is forwarded to *`10.0.0.1`*, but it still has the destination address of *`foo`*! The destination address of the packet is *not* changed to *`10.0.0.1`*. Most machines would probably drop a packet that they receive with a destination address that is not their own. Therefore, using a "fwd" rule does not often work the way the user expects. This behavior is a feature and not a bug.

See the FAQ about redirecting services, the natd(8) manual, or one of the several port redirecting utilities in the Ports Collection (http://www.FreeBSD.org/ports/index.html) for a correct way to do this.

19. How can I redirect service requests from one machine to another?

You can redirect FTP (and other service) request with the `sysutils/socket` port. Simply replace the service's command line to call `socket` instead, like so:

```
ftp stream tcp nowait nobody /usr/local/bin/socket socket ftp.example.com ftp
```

where *`ftp.example.com`* and *`ftp`* are the host and port to redirect to, respectively.

20. Where can I get a bandwidth management tool?

There are three bandwidth management tools available for FreeBSD. dummynet(4) is integrated into FreeBSD as part of ipfw(4). ALTQ (http://www.sonycsl.co.jp/person/kjc/programs.html) has been integrated into FreeBSD as part of pf(4). Bandwidth Manager from Emerging Technologies (http://www.etinc.com/) is a commercial product.

21. Why do I get "`/dev/bpf0: device not configured`"?

You are running a program that requires the Berkeley Packet Filter (bpf(4)), but it is not in your kernel. Add this to your kernel config file and build a new kernel:

```
device bpf        # Berkeley Packet Filter
```

22. How do I mount a disk from a Windows machine that is on my network, like smbmount in Linux?

Use the **SMBFS** toolset. It includes a set of kernel modifications and a set of userland programs. The programs and information are available as mount_smbfs(8) in the base system.

23. What are these messages about: "`Limiting icmp/open port/closed port response`" in my log files?

This is the kernel telling you that some activity is provoking it to send more ICMP or TCP reset (RST) responses than it thinks it should. ICMP responses are often generated as a result of attempted connections to unused UDP ports. TCP resets are generated as a result of attempted connections to unopened TCP ports. Among others, these are the kinds of activities which may cause these messages:

- Brute-force denial of service (DoS) attacks (as opposed to single-packet attacks which exploit a specific vulnerability).
- Port scans which attempt to connect to a large number of ports (as opposed to only trying a few well-known ports).

The first number in the message tells you how many packets the kernel would have sent if the limit was not in place, and the second number tells you the limit. You can control the limit using the `net.inet.icmp.icmplim` sysctl variable like this, where `300` is the limit in packets per second:

```
# sysctl -w net.inet.icmp.icmplim=300
```

If you do not want to see messages about this in your log files, but you still want the kernel to do response limiting, you can use the `net.inet.icmp.icmplim_output` sysctl variable to disable the output like this:

```
# sysctl -w net.inet.icmp.icmplim_output=0
```

Finally, if you want to disable response limiting, you can set the `net.inet.icmp.icmplim` sysctl variable (see above for an example) to `0`. Disabling response limiting is discouraged for the reasons listed above.

24. What are these "`arp: unknown hardware address format`" error messages?

This means that some device on your local Ethernet is using a MAC address in a format that FreeBSD does not recognize. This is probably caused by someone experimenting with an Ethernet card somewhere else on the network. You will see this most commonly on cable modem networks. It is harmless, and should not affect the performance of your FreeBSD machine.

25. Why do I keep seeing messages like: "`192.168.0.10 is on fxp1 but got reply from 00:15:17:67:cf:82 on rl0`", and how do I disable it?

Because a packet is coming from outside the network unexpectedly. To disable them, set `net.link.ether.inet.log_arp_wrong_iface` to `0`.

26. I have just installed **CVSup** but trying to execute it produces errors. What is wrong?

First, see if the error message you are receiving is like the one shown below.

```
/usr/libexec/ld-elf.so.1: Shared object "libXaw.so.6" not found
```

Errors like these are caused by installing the `net/cvsup` port on a machine which does not have the **Xorg** suite. If you want to use the GUI included with **CVSup** you will need to install **Xorg** now. Alternatively if you just wish to use **CVSup** from a command line you should delete the package previously installed. Then install the `net/cvsup-without-gui` or the `net/csup` port. If you have a recent FreeBSD release you may use csup(1). This is covered in more detail in the CVSup section (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/cvsup.html) of the Handbook.

Chapter 13 Security

1. What is a sandbox?

"Sandbox" is a security term. It can mean two things:

- A process which is placed inside a set of virtual walls that are designed to prevent someone who breaks into the process from being able to break into the wider system.

  The process is said to be able to "play" inside the walls. That is, nothing the process does in regards to executing code is supposed to be able to breech the walls so you do not have to do a detailed audit of its code to be able to say certain things about its security.

  The walls might be a user ID, for example. This is the definition used in the security(7) and named(8) man pages.

  Take the `ntalk` service, for example (see inetd(8)). This service used to run as user ID `root`. Now it runs as user ID `tty`. The `tty` user is a sandbox designed to make it more difficult for someone who has successfully hacked into the system via `ntalk` from being able to hack beyond that user ID.

- A process which is placed inside a simulation of the machine. This is more hard-core. Basically it means that someone who is able to break into the process may believe that he can break into the wider machine but is, in fact, only breaking into a simulation of that machine and not modifying any real data.

  The most common way to accomplish this is to build a simulated environment in a subdirectory and then run the processes in that directory chroot'd (i.e. `/` for that process is this directory, not the real `/` of the system).

  Another common use is to mount an underlying file system read-only and then create a file system layer on top of it that gives a process a seemingly writeable view into that file system. The process may believe it is able to write to those files, but only the process sees the effects — other processes in the system do not, necessarily.

  An attempt is made to make this sort of sandbox so transparent that the user (or hacker) does not realize that he is sitting in it.

UNIX implements two core sandboxes. One is at the process level, and one is at the userid level.

Every UNIX process is completely firewalled off from every other UNIX process. One process cannot modify the address space of another. This is unlike Windows where a process can easily overwrite the address space of any other, leading to a crash.

A UNIX process is owned by a particular userid. If the user ID is not the `root` user, it serves to firewall the process off from processes owned by other users. The user ID is also used to firewall off on-disk data.

2. What is securelevel?

The securelevel is a security mechanism implemented in the kernel. Basically, when the securelevel is positive, the kernel restricts certain tasks; not even the superuser (i.e., `root`) is allowed to do them. At the time of this writing, the securelevel mechanism is capable of, among other things, limiting the ability to:

- Unset certain file flags, such as `schg` (the system immutable flag).
- Write to kernel memory via `/dev/mem` and `/dev/kmem`.
- Load kernel modules.
- Alter firewall rules.

To check the status of the securelevel on a running system, simply execute the following command:

```
# sysctl kern.securelevel
```

The output will contain the name of the sysctl(8) variable (in this case, `kern.securelevel`) and a number. The latter is the current value of the securelevel. If it is positive (i.e., greater than 0), at least some of the securelevel's protections are enabled.

You cannot lower the securelevel of a running system; being able to do that would defeat its purpose. If you need to do a task that requires that the securelevel be non-positive (e.g., an `installworld` or changing the date), you will have to change the securelevel setting in `/etc/rc.conf` (you want to look for the `kern_securelevel` and `kern_securelevel_enable` variables) and reboot.

For more information on securelevel and the specific things all the levels do, please consult the init(8) manual page.

> **Warning:** Securelevel is not a silver bullet; it has many known deficiencies. More often than not, it provides a false sense of security.
>
> One of its biggest problems is that in order for it to be at all effective, all files used in the boot process up until the securelevel is set must be protected. If an attacker can get the system to execute their code prior to the securelevel being set (which happens quite late in the boot process since some things the system must do at start-up cannot be done at an elevated securelevel), its protections are invalidated. While this task of protecting all files used in the boot process is not technically impossible, if it is achieved, system maintenance will become a nightmare since one would have to take the system down, at least to single-user mode, to modify a configuration file.
>
> This point and others are often discussed on the mailing lists, particularly the FreeBSD security mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-security). Please search the archives here (http://www.FreeBSD.org/search/index.html) for an extensive discussion. Some people are hopeful that securelevel will soon go away in favor of a more fine-grained mechanism, but things are still hazy in this respect.
>
> Consider yourself warned.

3. BIND (`named`) is listening on some high-numbered ports. What is going on?

BIND uses a random high-numbered port for outgoing queries. Recent versions of it choose a new, random UDP port for each query. This may cause problems for some network configurations, especially if a firewall blocks incoming UDP packets on particular ports. If you want to get past that firewall, you can try the `avoid-v4-udp-ports` and `avoid-v6-udp-ports` options to avoid selecting random port numbers within a blocked range.

> **Warning:** If a port number (like 53) is specified via the `query-source` or `query-source-v6` options in `/etc/namedb/named.conf`, randomized port selection will not be used. It is strongly recommended that these options not be used to specify fixed port numbers.

Congratulations, by the way. It is good practice to read your sockstat(1) output and notice odd things!

4. The **sendmail** daemon is listening on port 587 as well as the standard port 25! What is going on?

Recent versions of **sendmail** support a mail submission feature that runs over port 587. This is not yet widely supported, but is growing in popularity.

5. What is this UID 0 `toor` account? Have I been compromised?

Do not worry. `toor` is an "alternative" superuser account (toor is root spelt backwards). Previously it was created when the bash(1) shell was installed but now it is created by default. It is intended to be used with a non-standard shell so you do not have to change `root`'s default shell. This is important as shells which are not part of the base distribution (for example a shell installed from ports or packages) are likely to be installed in `/usr/local/bin` which, by default, resides on a different file system. If `root`'s shell is located in `/usr/local/bin` and `/usr` (or whatever file system contains `/usr/local/bin`) is not mounted for some reason, `root` will not be able to log in to fix a problem (although if you reboot into single user mode you will be prompted for the path to a shell).

Some people use `toor` for day-to-day `root` tasks with a non-standard shell, leaving `root`, with a standard shell, for single user mode or emergencies. By default you cannot log in using `toor` as it does not have a password, so log in as `root` and set a password for `toor` if you want to use it.

6. Why is `suidperl` not working properly?

For security reasons, `suidperl` is not installed by default. If you want `suidperl` to be built during upgrades from source, edit `/etc/make.conf` and add `ENABLE_SUIDPERL=true` before you build `perl`.

Chapter 14 PPP

1. I cannot make ppp(8) work. What am I doing wrong?

You should first read the ppp(8) manual page and the PPP section of the handbook (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/ppp-and-slip.html#USERPPP). Enable logging with the following command:

```
set log Phase Chat Connect Carrier lcp ipcp ccp command
```

This command may be typed at the ppp(8) command prompt or it may be entered in the `/etc/ppp/ppp.conf` configuration file (the start of the `default` section is the best place to put it). Make sure that `/etc/syslog.conf` (see syslog.conf(5)) contains the lines below and the file `/var/log/ppp.log` exists:

```
!ppp
*.*         /var/log/ppp.log
```

You can now find out a lot about what is going on from the log file. Do not worry if it does not all make sense. If you need to get help from someone, it may make sense to them.

2. Why does ppp(8) hang when I run it?

This is usually because your hostname will not resolve. The best way to fix this is to make sure that `/etc/hosts` is consulted by your resolver first by editing `/etc/host.conf` and putting the `hosts` line first. Then, simply put an entry in `/etc/hosts` for your local machine. If you have no local network, change your `localhost` line:

```
127.0.0.1        foo.example.com foo localhost
```

Otherwise, simply add another entry for your host. Consult the relevant manual pages for more details.

You should be able to successfully `ping -c1 ‘hostname‘` when you are done.

3. Why will ppp(8) not dial in `-auto` mode?

First, check that you have got a default route. By running `netstat -rn` (see netstat(1)), you should see two entries like this:

```
Destination        Gateway            Flags     Refs     Use     Netif Expire
default            10.0.0.2           UGSc        0        0      tun0
10.0.0.2           10.0.0.1           UH          0        0      tun0
```

This is assuming that you have used the addresses from the handbook, the manual page, or from the `ppp.conf.sample` file. If you do not have a default route, it may be because you forgot to add the `HISADDR` line to the `ppp.conf` file.

Another reason for the default route line being missing is that you have mistakenly set up a default router in your `/etc/rc.conf` (see rc.conf(5)) file and you have omitted the line below from `ppp.conf`:

```
delete ALL
```

If this is the case, go back to the Final System Configuration (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/userppp.html#USERPPP-FINAL) section of the handbook.

4. What does "`No route to host`" mean?

This error is usually due that the following section is missing in your `/etc/ppp/ppp.linkup` file:

```
MYADDR:
  delete ALL
  add 0 0 HISADDR
```

This is only necessary if you have a dynamic IP address or do not know the address of your gateway. If you are using interactive mode, you can type the following after entering `packet mode` (packet mode is indicated by the capitalized PPP in the prompt):

```
delete ALL
add 0 0 HISADDR
```

Refer to the PPP and Dynamic IP addresses (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/userppp.html#USERPPP-DYNAMICIP) section of the handbook for further details.

5. Why does my connection drop after about 3 minutes?

The default PPP timeout is 3 minutes. This can be adjusted with the following line:

```
set timeout NNN
```

where *NNN* is the number of seconds of inactivity before the connection is closed. If *NNN* is zero, the connection is never closed due to a timeout. It is possible to put this command in the `ppp.conf` file, or to type it at the prompt in interactive mode. It is also possible to adjust it on the fly while the line is active by connecting to **ppp**'s server socket using telnet(1) or pppctl(8). Refer to the ppp(8) man page for further details.

6. Why does my connection drop under heavy load?

If you have Link Quality Reporting (LQR) configured, it is possible that too many LQR packets are lost between your machine and the peer. The ppp(8) program deduces that the line must therefore be bad, and disconnects. Prior to FreeBSD version 2.2.5, LQR was enabled by default. It is now disabled by default. LQR can be disabled with the following line:

```
disable lqr
```

7. Why does my connection drop after a random amount of time?

Sometimes, on a noisy phone line or even on a line with call waiting enabled, your modem may hang up because it thinks (incorrectly) that it lost carrier.

There is a setting on most modems for determining how tolerant it should be to temporary losses of carrier. On a U.S. Robotics® Sportster® for example, this is measured by the `S10` register in tenths of a second. To make your modem more forgiving, you could add the following send-expect sequence to your dial string:

```
set dial "...... ATS10=10 OK ......"
```

Refer to your modem manual for details.

8. Why does my connection hang after a random amount of time?

Many people experience hung connections with no apparent explanation. The first thing to establish is which side of the link is hung.

If you are using an external modem, you can simply try using ping(8) to see if the TD light is flashing when you transmit data. If it flashes (and the RD light does not), the problem is with the remote end. If TD does not flash, the problem is local. With an internal modem, you will need to use the `set server` command in your `ppp.conf` file. When the hang occurs, connect to ppp(8) using pppctl(8). If your network connection suddenly revives (PPP was revived due to the activity on the diagnostic socket) or if you cannot connect (assuming the `set socket` command succeeded at startup time), the problem is local. If you can connect and things are still hung, enable local async logging with `set log local async` and use ping(8) from another window or terminal to make use of the link. The async logging will show you the data being transmitted and received on the link. If data is going out and not coming back, the problem is remote.

Having established whether the problem is local or remote, you now have two possibilities:

- If the problem is remote, read on entry Q: 9..
- If the problem is local, read on entry Q: 10..

9. The remote end is not responding. What can I do?

There is very little you can do about this. Most ISPs will refuse to help if you are not running a Microsoft OS. You can `enable lqr` in your `ppp.conf` file, allowing ppp(8) to detect the remote failure and hang up, but this detection is relatively slow and therefore not that useful. You may want to avoid telling your ISP that you are running user-PPP.

First, try disabling all local compression by adding the following to your configuration:

```
disable pred1 deflate deflate24 protocomp acfcomp shortseq vj
deny pred1 deflate deflate24 protocomp acfcomp shortseq vj
```

Then reconnect to ensure that this makes no difference. If things improve or if the problem is solved completely, determine which setting makes the difference through trial and error. This will provide good ammunition when you contact your ISP (although it may make it apparent that you are not running a Microsoft product).

Before contacting your ISP, enable async logging locally and wait until the connection hangs again. This may use up quite a bit of disk space. The last data read from the port may be of interest. It is usually ASCII data, and may even describe the problem (“`Memory fault`”, “`Core dumped`”).

If your ISP is helpful, they should be able to enable logging on their end, then when the next link drop occurs, they may be able to tell you why their side is having a problem. Feel free to send the details to Brian Somers <`brian@FreeBSD.org`>, or even to ask your ISP to contact him directly.

10. ppp(8) has hung. What can I do?

Your best bet here is to rebuild ppp(8) with debugging information, and then use gdb(1) to grab a stack trace from the **ppp** process that is stuck. To rebuild the **ppp** utility with debugging information, you can type:

```
# cd /usr/src/usr.sbin/ppp
# env DEBUG_FLAGS='-g' make clean
# env DEBUG_FLAGS='-g' make install
```

Then you should restart **ppp** and wait until it hangs again. When the debug build of **ppp** hangs, start **gdb** on the stuck process by typing:

```
# gdb ppp `pgrep ppp`
```

At the **gdb** prompt, you can use the `bt` or `where` commands to get a stack trace. Save the output of your **gdb** session, and "detach" from the running process by the `quit` command of **gdb**.

Finally, send the log of your **gdb** session to Brian Somers <`brian@FreeBSD.org`>.

11. Why does nothing happen after the "Login OK!" message?

Prior to FreeBSD version 2.2.5, once the link was established, ppp(8) would wait for the peer to initiate the Line Control Protocol (LCP). Many ISPs will not initiate negotiations and expect the client to do so. To force ppp(8) to initiate the LCP, use the following line:

```
set openmode active
```

> **Note:** It usually does no harm if both sides initiate negotiation, so openmode is now active by default. However, the next section explains when it *does* do some harm.

12. I keep seeing errors about magic being the same. What does it mean?

Occasionally, just after connecting, you may see messages in the log that say "`Magic is same`". Sometimes, these messages are harmless, and sometimes one side or the other exits. Most PPP implementations cannot survive this problem, and even if the link seems to come up, you will see repeated configure requests and configure acknowledgments in the log file until ppp(8) eventually gives up and closes the connection.

This normally happens on server machines with slow disks that are spawning a getty(8) on the port, and executing ppp(8) from a login script or program after login. There were reports of it happening consistently when using slirp. The reason is that in the time taken between getty(8) exiting and ppp(8) starting, the client-side ppp(8) starts sending Line Control Protocol (LCP) packets. Because ECHO is still switched on for the port on the server, the client ppp(8) sees these packets "reflect" back.

One part of the LCP negotiation is to establish a magic number for each side of the link so that "reflections" can be detected. The protocol says that when the peer tries to negotiate the same magic number, a NAK should be sent and a new magic number should be chosen. During the period that the server port has ECHO turned on, the client ppp(8) sends LCP packets, sees the same magic in the reflected packet and NAKs it. It also sees the NAK reflect (which also

means ppp(8) must change its magic). This produces a potentially enormous number of magic number changes, all of which are happily piling into the server's tty buffer. As soon as ppp(8) starts on the server, it is flooded with magic number changes and almost immediately decides it has tried enough to negotiate LCP and gives up. Meanwhile, the client, who no longer sees the reflections, becomes happy just in time to see a hangup from the server.

This can be avoided by allowing the peer to start negotiating with the following line in your `ppp.conf` file:

```
set openmode passive
```

This tells ppp(8) to wait for the server to initiate LCP negotiations. Some servers however may never initiate negotiations. If this is the case, you can do something like:

```
set openmode active 3
```

This tells ppp(8) to be passive for 3 seconds, and then to start sending LCP requests. If the peer starts sending requests during this period, ppp(8) will immediately respond rather than waiting for the full 3 second period.

13. LCP negotiations continue until the connection is closed. What is wrong?

There is currently an implementation mis-feature in ppp(8) where it does not associate LCP, CCP & IPCP responses with their original requests. As a result, if one PPP implementation is more than 6 seconds slower than the other side, the other side will send two additional LCP configuration requests. This is fatal.

Consider two implementations, `A` and `B`. `A` starts sending LCP requests immediately after connecting and `B` takes 7 seconds to start. When `B` starts, `A` has sent 3 LCP REQs. We are assuming the line has ECHO switched off, otherwise we would see magic number problems as described in the previous section. `B` sends a REQ, then an ACK to the first of `A`'s REQs. This results in `A` entering the OPENED state and sending and ACK (the first) back to `B`. In the meantime, `B` sends back two more ACKs in response to the two additional REQs sent by `A` before `B` started up. `B` then receives the first ACK from `A` and enters the OPENED state. `A` receives the second ACK from `B` and goes back to the REQ-SENT state, sending another (forth) REQ as per the RFC. It then receives the third ACK and enters the OPENED state. In the meantime, `B` receives the forth REQ from `A`, resulting in it reverting to the ACK-SENT state and sending another (second) REQ and (forth) ACK as per the RFC. `A` gets the REQ, goes into REQ-SENT and sends another REQ. It immediately receives the following ACK and enters OPENED.

This goes on until one side figures out that they are getting nowhere and gives up.

The best way to avoid this is to configure one side to be `passive` — that is, make one side wait for the other to start negotiating. This can be done with the following command:

```
set openmode passive
```

Care should be taken with this option. You should also use this command to limit the amount of time that ppp(8) waits for the peer to begin negotiations:

```
set stopped N
```

Alternatively, the following command (where *N* is the number of seconds to wait before starting negotiations) can be used:

```
set openmode active N
```

Check the manual page for details.

14. Why does ppp(8) lock up when I shell out to test it?

When you execute the `shell` or `!` command, ppp(8) executes a shell (or if you have passed any arguments, ppp(8) will execute those arguments). The **ppp** program will wait for the command to complete before continuing. If you attempt to use the PPP link while running the command, the link will appear to have frozen. This is because ppp(8) is waiting for the command to complete.

If you wish to execute commands like this, use the `!bg` command instead. This will execute the given command in the background, and ppp(8) can continue to service the link.

15. Why does ppp(8) over a null-modem cable never exit?

There is no way for ppp(8) to automatically determine that a direct connection has been dropped. This is due to the lines that are used in a null-modem serial cable. When using this sort of connection, LQR should always be enabled with the following line:

```
enable lqr
```

LQR is accepted by default if negotiated by the peer.

16. Why does ppp(8) dial for no reason in `-auto` mode?

If ppp(8) is dialing unexpectedly, you must determine the cause, and set up Dial filters (dfilters) to prevent such dialing.

To determine the cause, use the following line:

```
set log +tcp/ip
```

This will log all traffic through the connection. The next time the line comes up unexpectedly, you will see the reason logged with a convenient timestamp next to it.

You can now disable dialing under these circumstances. Usually, this sort of problem arises due to DNS lookups. To prevent DNS lookups from establishing a connection (this will *not* prevent ppp(8) from passing the packets through an established connection), use the following:

```
set dfilter 1 deny udp src eq 53
set dfilter 2 deny udp dst eq 53
set dfilter 3 permit 0/0 0/0
```

This is not always suitable, as it will effectively break your demand-dial capabilities — most programs will need a DNS lookup before doing any other network related things.

In the DNS case, you should try to determine what is actually trying to resolve a host name. A lot of the time, sendmail(8) is the culprit. You should make sure that you tell **sendmail** not to do any DNS lookups in its configuration file. See the section on using email with a dialup connection (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/smtp-dialup.html) in the FreeBSD Handbook for details on how to create your own configuration file and what should go into it. You may also want to add the following line to your `.mc` file:

```
define('confDELIVERY_MODE', 'd')dnl
```

This will make **sendmail** queue everything until the queue is run (usually, sendmail is invoked with `-bd -q30m`, telling it to run the queue every 30 minutes) or until a `sendmail -q` is done (perhaps from your `ppp.linkup` file).

17. What do these CCP errors mean?

I keep seeing the following errors in my log file:

```
CCP: CcpSendConfigReq
CCP: Received Terminate Ack (1) state = Req-Sent (6)
```

This is because ppp(8) is trying to negotiate Predictor1 compression, and the peer does not want to negotiate any compression at all. The messages are harmless, but if you wish to remove them, you can disable Predictor1 compression locally too:

```
disable pred1
```

18. Why does ppp(8) not log my connection speed?

In order to log all lines of your modem "conversation", you must enable the following:

```
set log +connect
```

This will make ppp(8) log everything up until the last requested "expect" string.

If you wish to see your connect speed and are using PAP or CHAP (and therefore do not have anything to "chat" after the CONNECT in the dial script — no `set login` script), you must make sure that you instruct ppp(8) to "expect" the whole CONNECT line, something like this:

```
set dial "ABORT BUSY ABORT NO\\sCARRIER TIMEOUT 4 \
  \"\" ATZ OK-ATZ-OK ATDT\\T TIMEOUT 60 CONNECT \\c \\n"
```

Here, we get our CONNECT, send nothing, then expect a line-feed, forcing ppp(8) to read the whole CONNECT response.

19. Why does ppp(8) ignore the `\` character in my chat script?

The **ppp** utility parses each line in your config files so that it can interpret strings such as `set phone "123 456 789"` correctly and realize that the number is actually only *one* argument. In order to specify a `"` character, you must escape it using a backslash (`\`).

When the chat interpreter parses each argument, it re-interprets the argument in order to find any special escape sequences such as `\P` or `\T` (see the manual page). As a result of this double-parsing, you must remember to use the correct number of escapes.

If you wish to actually send a `\` character to (say) your modem, you would need something like:

```
set dial "\"\" ATZ OK-ATZ-OK AT\\\\X OK"
```

It will result in the following sequence:

```
ATZ
OK
AT\X
OK
```

Or:

```
set phone 1234567
set dial "\"\" ATZ OK ATDT\\T"
```

It will result in the following sequence:

```
ATZ
OK
ATDT1234567
```

20. Why does ppp(8) get a "`Segmentation fault`", but I see no `ppp.core` file?

The **ppp** utility (or any other program for that matter) should never dump core. Because ppp(8) runs with an effective user ID of `0`, the operating system will not write core image of ppp(8) to disk before terminating it. If, however ppp(8) is actually terminating due to a segmentation violation or some other signal that normally causes core to be dumped, *and* you are sure you are using the latest version (see the start of this section), then you should install the system sources and do the following:

```
#  cd /usr/src/usr.sbin/ppp
# echo STRIP= >> /etc/make.conf
# echo CFLAGS+=-g >> /etc/make.conf
# make install clean
```

You will now have a debuggable version of ppp(8) installed. You will have to be `root` to run ppp(8) as all of its privileges have been revoked. When you start ppp(8), take a careful note of what your current directory was at the time.

Now, if and when ppp(8) receives the segmentation violation, it will dump a core file called `ppp.core`. You should then do the following:

```
% su
# gdb /usr/sbin/ppp ppp.core
(gdb) bt
.....
(gdb) f 0
....
(gdb) i args
....
(gdb) l
.....
```

All of this information should be given alongside your question, making it possible to diagnose the problem.

If you are familiar with gdb(1), you may wish to find out some other bits and pieces such as what actually caused the dump or the addresses and values of the relevant variables.

21. Why does the process that forces a dial in `-auto` mode never connect?

This was a known problem with ppp(8) set up to negotiate a dynamic local IP number with the peer in `-auto` mode. It has been fixed a long time ago — search the manual page for `iface`.

The problem was that when that initial program calls connect(2), the IP number of the tun(4) interface is assigned to the socket endpoint. The kernel creates the first outgoing packet and writes it to the tun(4) device. ppp(8) then reads the packet and establishes a connection. If, as a result of ppp(8)'s dynamic IP assignment, the interface address is changed, the original socket endpoint will be invalid. Any subsequent packets sent to the peer will usually be dropped. Even if they are not, any responses will not route back to the originating machine as the IP number is no longer owned by that machine.

There are several theoretical ways to approach this problem. It would be nicest if the peer would re-assign the same IP number if possible. The current version of ppp(8) does this, but most other implementations do not.

The easiest method from our side would be to never change the tun(4) interface IP number, but instead to change all outgoing packets so that the source IP number is changed from the interface IP to the negotiated IP on the fly. This is essentially what the `iface-alias` option in the latest version of ppp(8) is doing (with the help of libalias(3) and ppp(8)'s `-nat` switch) — it is maintaining all previous interface addresses and NATing them to the last negotiated address.

Another alternative (and probably the most reliable) would be to implement a system call that changes all bound sockets from one IP to another. ppp(8) would use this call to modify the sockets of all existing programs when a new IP number is negotiated. The same system call could be used by DHCP clients when they are forced to call the `bind()` function for their sockets.

Yet another possibility is to allow an interface to be brought up without an IP number. Outgoing packets would be given an IP number of `255.255.255.255` up until the first `SIOCAIFADDR` ioctl(2) is done. This would result in fully binding the socket. It would be up to ppp(8) to change the source IP number, but only if it is set to `255.255.255.255`, and only the IP number and IP checksum would need to change. This, however is a bit of a hack as the kernel would be sending bad packets to an improperly configured interface, on the assumption that some other mechanism is capable of fixing things retrospectively.

22. Why do most games not work with the `-nat` switch?

The reason games and the like do not work when libalias(3) is in use is that the machine on the outside will try to open a connection or send (unsolicited) UDP packets to the machine on the inside. The NAT software does not know that it should send these packets to the interior machine.

To make things work, make sure that the only thing running is the software that you are having problems with, then either run tcpdump(1) on the tun(4) interface of the gateway or enable ppp(8) TCP/IP logging (`set log +tcp/ip`) on the gateway.

When you start the offending software, you should see packets passing through the gateway machine. When something comes back from the outside, it will be dropped (that is the problem). Note the port number of these packets then shut down the offending software. Do this a few times to see if the port numbers are consistent. If they are, then the following line in the relevant section of `/etc/ppp/ppp.conf` will make the software functional:

`nat port` *`proto internalmachine`*`:`*`port port`*

where *`proto`* is either `tcp` or `udp`, *`internalmachine`* is the machine that you want the packets to be sent to and *`port`* is the destination port number of the packets.

You will not be able to use the software on other machines without changing the above command, and running the software on two internal machines at the same time is out of the question — after all, the outside world is seeing your entire internal network as being just a single machine.

If the port numbers are not consistent, there are three more options:

1. Submit support in libalias(3). Examples of "special cases" can be found in `/usr/src/sys/netinet/libalias/alias_*.c` (`alias_ftp.c` is a good prototype). This usually involves reading certain recognized outgoing packets, identifying the instruction that tells the outside machine to initiate a connection back to the internal machine on a specific (random) port and setting up a "route" in the alias table so that the subsequent packets know where to go.

   This is the most difficult solution, but it is the best and will make the software work with multiple machines.

2. Use a proxy. The application may support `socks5` for example, or (as in the `cvsup` case) may have a "passive" option that avoids ever requesting that the peer open connections back to the local machine.

3. Redirect everything to the internal machine using `nat addr`. This is the sledge-hammer approach.

23. Has anybody made a list of useful port numbers?

Not yet, but this is intended to grow into such a list (if any interest is shown). In each example, *`internal`* should be replaced with the IP number of the machine playing the game.

- **Asheron's Call**

  `nat port udp` *`internal`* `:65000 65000`

  Manually change the port number within the game to `65000`. If you have got a number of machines that you wish to play on assign a unique port number for each (i.e. `65001`, `65002`, etc) and add a `nat port` line for each one.

- **Half Life**

  `nat port udp` *`internal`*`:27005 27015`

- **PCAnywhere 8.0**

  `nat port udp` *`internal`*`:5632 5632`

  `nat port tcp` *`internal`*`:5631 5631`

- **Quake**

  `nat port udp` *`internal`*`:6112 6112`

- **Quake 2**

  `nat port udp` *`internal`*`:27901 27910`

  `nat port udp` *`internal`*`:60021 60021`

  `nat port udp` *`internal`*`:60040 60040`

- **Red Alert**

  `nat port udp` *`internal`*`:8675 8675`

  `nat port udp` *`internal`*`:5009 5009`

24. What are FCS errors?

FCS stands for `F`rame `C`heck `S`equence. Each PPP packet has a checksum attached to ensure that the data being received is the data being sent. If the FCS of an incoming packet is incorrect, the packet is dropped and the HDLC FCS count is increased. The HDLC error values can be displayed using the `show hdlc` command.

If your link is bad (or if your serial driver is dropping packets), you will see the occasional FCS error. This is not usually worth worrying about although it does slow down the compression protocols substantially. If you have an external modem, make sure your cable is properly shielded from interference — this may eradicate the problem.

If your link freezes as soon as you have connected and you see a large number of FCS errors, this may be because your link is not 8-bit clean. Make sure your modem is not using software flow control (XON/XOFF). If your datalink *must* use software flow control, use the command `set accmap 0x000a0000` to tell ppp(8) to escape the `^Q` and `^S` characters.

Another reason for seeing too many FCS errors may be that the remote end has stopped talking PPP. You may want to enable `async` logging at this point to determine if the incoming data is actually a login or shell prompt. If you have a shell prompt at the remote end, it is possible to terminate ppp(8) without dropping the line by using the `close lcp` command (a following `term` command) will reconnect you to the shell on the remote machine.

If nothing in your log file indicates why the link might have been terminated, you should ask the remote administrator (your ISP?) why the session was terminated.

25. Why do Mac OS and Windows 98 connections freeze when running PPPoE on the gateway?

Thanks to Michael Wozniak <`mwozniak@netcom.ca`> for figuring this out and Dan Flemming <`danflemming@mac.com`> for the Mac solution:

This is due to what is called a “Black Hole” router. Mac OS and Windows 98 (and maybe other Microsoft OSs) send TCP packets with a requested segment size too big to fit into a PPPoE frame (MTU is `1500` by default for Ethernet) *and* have the “do not fragment” bit set (default of TCP) and the Telco router is not sending ICMP “must fragment” back to the WWW site you are trying to load. (Alternatively, the router is sending the ICMP packet correctly, but the firewall at the WWW site is dropping it.) When the www server is sending you frames that do not fit into the PPPoE pipe the Telco router drops them on the floor and your page does not load (some pages/graphics do as they are smaller than a MSS). This seems to be the default of most Telco PPPoE configurations.

One fix is to use **regedit** on your 95/98 system to add the following registry entry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Class\NetTrans\0000\MaxMTU
```

It should be a string with a value `1436`, as some ADSL routers are reported to be unable to deal with packets larger than this. This registry key has been changed to `Tcpip\Parameters\Interfaces\`*`ID for adapter`*`\MTU` in Windows 2000 and becomes a `DWORD`.

Refer to the Microsoft Knowledge Base documents Q158474 - Windows TCPIP Registry Entries (http://support.microsoft.com/support/kb/articles/Q158/4/74.asp) and Q120642 - TCPIP & NBT Configuration Parameters for Windows NT (http://support.microsoft.com/support/kb/articles/Q120/6/42.asp) for more information on changing Windows MTU to work with a NAT router.

Another regedit possibility under Windows 2000 to set the `Tcpip\Parameters\Interfaces\`*`ID for adapter`*`\EnablePMTUBHDetect DWORD` to `1` as mentioned in the Microsoft document 120642 mentioned above.

Unfortunately, Mac OS does not provide an interface for changing TCP/IP settings. However, there are several commercial programs available that will allow users to customize TCP/IP settings. Mac OS NAT users should search for their MTU settings and enter `1450` instead of `1500`.

The ppp(8) has an `enable tcpmssfixup` command that will automatically adjust the MSS to an appropriate value. This facility is enabled by default. If you are stuck with an older version of ppp(8), you may want to look at the `net/tcpmssd` port.

26. None of this helps — I am desperate! What can I do?

If all else fails, send as much information as you can, including your config files, how you are starting ppp(8), the relevant parts of your log file and the output of the `netstat -rn` command (before and after connecting) to the FreeBSD general questions mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-questions) or the comp.unix.bsd.freebsd.misc (news:comp.unix.bsd.freebsd.misc) news group, and someone should point you in the right direction.

Chapter 15 Serial Communications

This section answers common questions about serial communications with FreeBSD. PPP and SLIP are covered in the Networking section.

1. How do I tell if FreeBSD found my serial ports?

As the FreeBSD kernel boots, it will probe for the serial ports in your system for which the kernel was configured. You can either watch your system closely for the messages it prints or run this command after your system is up and running:

```
% dmesg | grep -E "^sio[0-9]"
```

Here is some example output from the above command:

```
sio0: <16550A-compatible COM port> port 0x3f8-0x3ff irq 4 flags 0x10 on acpi0
sio0: type 16550A
sio1: <16550A-compatible COM port> port 0x2f8-0x2ff irq 3 on acpi0
sio1: type 16550A
```

This shows two serial ports. The first is on IRQ 4, is using port address `0x3f8`, and has a 16550A-type UART chip. The second uses the same kind of chip but is on IRQ 3 and is at port address `0x2f8`. Internal modem cards are treated just like serial ports — except that they always have a modem "attached" to the port.

The `GENERIC` kernel includes support for two serial ports using the same IRQ and port address settings in the above example. If these settings are not right for your system, or if you have added modem cards or have more serial ports than your kernel is configured for, just reconfigure your kernel. See section about building a kernel for more details.

2. How do I tell if FreeBSD found my modem cards?

Refer to the answer to the previous question.

3. How do I access the serial ports on FreeBSD?

The third serial port, `sio2` (see sio(4), known as `COM3` in DOS), is on `/dev/cuad2` for dial-out devices, and on `/dev/ttyd2` for dial-in devices. What is the difference between these two classes of devices?

You use `ttyd`*X* for dial-ins. When opening `/dev/ttyd`*X* in blocking mode, a process will wait for the corresponding `cuad`*X* device to become inactive, and then wait for the carrier detect line to go active. When you open the `cuad`*X* device, it makes sure the serial port is not already in use by the `ttyd`*X* device. If the port is available, it "steals" it from the `ttyd`*X* device. Also, the `cuad`*X* device does not care about carrier detect. With this scheme and an auto-answer modem, you can have remote users log in and you can still dial out with the same modem and the system will take care of all the conflicts.

4. How do I enable support for a multiport serial card?

Again, the section on kernel configuration provides information about configuring your kernel. For a multiport serial card, place an sio(4) line for each serial port on the card in the device.hints(5) file. But place the IRQ specifiers on only one of the entries. All of the ports on the card should share one IRQ. For consistency, use the last serial port to specify the IRQ. Also, specify the following option in the kernel configuration file:

```
options COM_MULTIPORT
```

The following `/boot/device.hints` example is for an AST 4-port serial card on IRQ 12:

```
hint.sio.4.at="isa"
hint.sio.4.port="0x2a0"
hint.sio.4.flags="0x701"
hint.sio.5.at="isa"
hint.sio.5.port="0x2a8"
hint.sio.5.flags="0x701"
hint.sio.6.at="isa"
hint.sio.6.port="0x2b0"
hint.sio.6.flags="0x701"
hint.sio.7.at="isa"
hint.sio.7.port="0x2b8"
hint.sio.7.flags="0x701"
hint.sio.7.irq="12"
```

The flags indicate that the master port has minor number `7` (`0x700`), and all the ports share an IRQ (`0x001`).

5. Can FreeBSD handle multiport serial cards sharing IRQs?

Not yet. You will have to use a different IRQ for each card.

6. Can I set the default serial parameters for a port?

See the Serial Communications (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/serial.html#SERIAL-HW-CONFIG) section in the FreeBSD Handbook.

7. How can I enable dialup logins on my modem?

Please read the section about Dial-in Services (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialup.html) in the FreeBSD Handbook.

8. How can I connect a dumb terminal to my FreeBSD box?

You can find this information in the Terminals (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/term.html) section of the FreeBSD Handbook.

9. Why can I not run `tip` or `cu`?

On your system, the programs tip(1) and cu(1) can only access the `/var/spool/lock` directory via user `uucp` and group `dialer`. You can use the group `dialer` to control who has access to your modem or remote systems. Just add yourself to group `dialer`.

Alternatively, you can let everyone on your system run tip(1) and cu(1) by typing:

```
# chmod 4511 /usr/bin/cu
# chmod 4511 /usr/bin/tip
```

10. My stock Hayes modem is not supported — what can I do?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#HAYES-UNSUPPORTED) in the FreeBSD Handbook.

11. How am I expected to enter these AT commands?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#DIRECT-AT) in the FreeBSD Handbook.

12. Why does the `@` sign for the `pn` capability not work?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#GT-FAILURE) in the FreeBSD Handbook.

13. How can I dial a phone number on the command line?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#DIAL-COMMAND-LINE) in the FreeBSD Handbook.

14. Do I have to type in the bps rate every time I do that?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#SET-BPS) in the FreeBSD Handbook.

15. How can I more easily access a number of hosts through a terminal server?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#TERMINAL-SERVER) in the FreeBSD Handbook.

16. Can tip try more than one line for each site?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#TIP-MULTILINE) in the FreeBSD Handbook.

17. Why do I have to hit **Ctrl+P** twice to send **Ctrl+P** once?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#MULTI-CONTROLP) in the FreeBSD Handbook.

18. Why is everything I type suddenly in UPPER CASE?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#UPPERCASE) in the FreeBSD Handbook.

19. How can I do file transfers with `tip`?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#TIP-FILETRANSFER) in the FreeBSD Handbook.

20. How can I run zmodem with `tip`?

See this answer (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/dialout.html#ZMODEM-TIP) in the FreeBSD Handbook.

Chapter 16 Miscellaneous Questions

1. FreeBSD uses far more swap space than Linux. Why?

FreeBSD only appears to use more swap than Linux. In actual fact, it does not. The main difference between FreeBSD and Linux in this regard is that FreeBSD will proactively move entirely idle, unused pages of main memory into swap in order to make more main memory available for active use. Linux tends to only move pages to swap as a last resort. The perceived heavier use of swap is balanced by the more efficient use of main memory.

Note that while FreeBSD is proactive in this regard, it does not arbitrarily decide to swap pages when the system is truly idle. Thus you will not find your system all paged out when you get up in the morning after leaving it idle overnight.

2. Why does `top` show very little free memory even when I have very few programs running?

The simple answer is that free memory is wasted memory. Any memory that your programs do not actively allocate is used within the FreeBSD kernel as disk cache. The values shown by top(1) labeled as `Inact`, `Cache`, and `Buf` are all cached data at different aging levels. This cached data means the system does not have to access a slow disk again for data it has accessed recently, thus increasing overall performance. In general, a low value shown for `Free` memory in top(1) is good, provided it is not *very* low.

3. Why will `chmod` not change the permissions on symlinks?

Symlinks do not have permissions, and by default, chmod(1) will follow symlinks to change the permissions on the source file, if possible. So if you have a file, `foo`, and a symlink to that file, `bar`, then this command will always succeed.

```
% chmod g-w bar
```

However, the permissions on `bar` will not have changed.

When changing modes of the file hierarchies rooted in the files instead of the files themselves, you have to use either `-H` or `-L` together with the `-R` option to make this work. See the chmod(1) and symlink(7) manual pages for more info.

> **Warning:** The `-R` option does a *recursive* chmod(1). Be careful about specifying directories or symlinks to directories to chmod(1). If you want to change the permissions of a directory referenced by a symlink, use chmod(1) without any options and follow the symlink with a trailing slash (`/`). For example, if `foo` is a symlink to directory `bar`, and you want to change the permissions of `foo` (actually `bar`), you would do something like:
>
> ```
> % chmod 555 foo/
> ```
>
> With the trailing slash, chmod(1) will follow the symlink, `foo`, to change the permissions of the directory, `bar`.

4. Can I run DOS binaries under FreeBSD?

Yes, you can use `emulators/doscmd`, a DOS emulation program, available in the FreeBSD Ports Collection.

If **doscmd** will not suffice, the add-on utility `emulators/pcemu` emulates an 8088 and enough BIOS services to run many DOS text mode applications. It requires the X Window System.

You may also try `emulators/dosbox` from the FreeBSD Ports Collection. The main focus of this application is emulating old DOS games using the local file system for files.

5. What do I need to do to translate a FreeBSD document into my native language?

See the Translation FAQ (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/fdp-primer/translations.html) in the FreeBSD Documentation Project Primer.

6. Why does my email to any address at `FreeBSD.org` bounce?

The `FreeBSD.org` mail system implements some of the stricter **Postfix** checks on incoming mail and rejects mail that is either misconfigured or is potential spam. Your mail might bounce for one of the following reasons:

- The email is being sent from a known spam domain or IP block.

  The FreeBSD mail servers reject email from known spam sources. If you have service through a company or domain who generates or relays spam, please switch to a service provider who does not.

- The body of the email only contains HTML.

  Mail should be sent in plain text only. Please configure your mail user agent to send plain text.

- The mailer at `FreeBSD.org` cannot resolve the IP address of the connecting host back to a symbolic name.

  Working reverse DNS is a standard requirement for accepting mail from a host. Set up reverse DNS for your mail server's IP address. Many home services (DSL, cable, dialup, etc.) will not give you this option. In this case, relay your email through your service provider's mail server.

- The hostname given in the EHLO/HELO part of the SMTP exchange cannot be resolved to an IP address.

  A fully qualified, resolvable host name is necessary in this part of the SMTP dialogue before mail will be accepted. If you do not have a host name that is registered in the DNS, then you should use your service provider's mail server to relay your mail.

- Your message had a message ID ending with the string "localhost".

  Some mail user agents generate bad message IDs which will not be accepted. You will need to persuade your mail user agent to generate a valid message ID or else configure your mail transfer agent to rewrite them.

7. Where can I find a free FreeBSD account?

While FreeBSD does not provide open access to any of their servers, others do provide open access UNIX systems. The charge varies and limited services may be available.

Arbornet, Inc (http://www.arbornet.org/), also known as *M-Net*, has been providing open access to UNIX systems since 1983. Starting on an Altos running System III, the site switched to BSD/OS in 1991. In June of 2000, the site switched again to FreeBSD. *M-Net* can be accessed via **telnet** and **SSH** and provides basic access to the entire FreeBSD software suite. However, network access is limited to members and patrons who donate to the system, which is run as a non-profit organization. *M-Net* also provides an bulletin board system and interactive chat.

Grex (http://www.grex.org/) provides a site very similar to *M-Net* including the same bulletin board and interactive chat software. However, the machine is a Sun 4M and is running SunOS.

8. What is `sup`, and how do I use it?

SUP (http://www.FreeBSD.org/cgi/ports.cgi?^sup) stands for Software Update Protocol, and was developed by CMU for keeping their development trees in sync. It was used to keep remote sites in sync with the Project's central development sources.

SUP is not bandwidth friendly, and has been retired. The current recommended method to keep your sources up to date is CVSup (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/synching.html#CVSUP)

9. What is the cute little red guy's name?

He does not have one, and is just called "the BSD daemon". If you insist upon using a name, call him "beastie". Note that "beastie" is pronounced "BSD".

You can learn more about the BSD daemon on his home page (http://www.mckusick.com/beastie/index.html).

10. Can I use the BSD daemon image?

Perhaps. The BSD daemon is copyrighted by Marshall Kirk McKusick. You will want to check his Statement on the Use of the BSD Daemon Figure (http://www.mckusick.com/beastie/mainpage/copyright.html) for detailed usage terms.

In summary, you are free to use the image in a tasteful manner, for personal use, so long as appropriate credit is given. If you want to use him commercially, you must contact Kirk McKusick <`mckusick@FreeBSD.org`>. More details are available on the BSD Daemon's home page (http://www.mckusick.com/beastie/index.html).

11. Do you have any BSD daemon images I could use?

You will find eps and Xfig drawings under `/usr/share/examples/BSD_daemon/`.

12. I have seen an acronym or other term on the mailing lists and I do not understand what it means. Where should I look?

Please see the FreeBSD Glossary (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/freebsd-glossary.html).

13. Why should I care what color the bikeshed is?

The really, really short answer is that you should not. The somewhat longer answer is that just because you are capable of building a bikeshed does not mean you should stop others from building one just because you do not like the color they plan to paint it. This is a metaphor indicating that you need not argue about every little feature just because you know enough to do so. Some people have commented that the amount of noise generated by a change is inversely proportional to the complexity of the change.

The longer and more complete answer is that after a very long argument about whether sleep(1) should take fractional second arguments, Poul-Henning Kamp <`phk@FreeBSD.org`> posted a long message entitled "A bike shed (any color will do) on greener grass...

(http://www.FreeBSD.org/cgi/getmsg.cgi?fetch=506636+517178+/usr/local/www/db/text/1999/freebsd-hackers/19991003.freebsd-hackers)". The appropriate portions of that message are quoted below.

> "What is it about this bike shed?" Some of you have asked me.
>
> It is a long story, or rather it is an old story, but it is quite short actually. C. Northcote Parkinson wrote a book in the early 1960s, called "Parkinson's Law", which contains a lot of insight into the dynamics of management.
>
> *[snip a bit of commentary on the book]*
>
> In the specific example involving the bike shed, the other vital component is an atomic power-plant, I guess that illustrates the age of the book.
>
> Parkinson shows how you can go into the board of directors and get approval for building a multi-million or even billion dollar atomic power plant, but if you want to build a bike shed you will be tangled up in endless discussions.
>
> Parkinson explains that this is because an atomic plant is so vast, so expensive and so complicated that people cannot grasp it, and rather than try, they fall back on the assumption that somebody else checked all the details before it got this far. Richard P. Feynmann gives a couple of interesting, and very much to the point, examples relating to Los Alamos in his books.
>
> A bike shed on the other hand. Anyone can build one of those over a weekend, and still have time to watch the game on TV. So no matter how well prepared, no matter how reasonable you are with your proposal, somebody will seize the chance to show that he is doing his job, that he is paying attention, that he is *here*.
>
> In Denmark we call it "setting your fingerprint". It is about personal pride and prestige, it is about being able to point somewhere and say "There! *I* did that." It is a strong trait in politicians, but present in most people given the chance. Just think about footsteps in wet cement.

—Poul-Henning Kamp <phk@FreeBSD.org> on freebsd-hackers (http://lists.FreeBSD.org/mailman/listinfo/freebsd-hackers), October 2, 1999

Chapter 17 The FreeBSD Funnies

1. How cool is FreeBSD?

Q. Has anyone done any temperature testing while running FreeBSD? I know Linux runs cooler than DOS, but have never seen a mention of FreeBSD. It seems to run really hot.

A. No, but we have done numerous taste tests on blindfolded volunteers who have also had 250 micrograms of LSD-25 administered beforehand. 35% of the volunteers said that FreeBSD tasted sort of orange, whereas Linux tasted like purple haze. Neither group mentioned any significant variances in temperature. We eventually had to throw the results of this survey out entirely anyway when we found that too many volunteers were wandering out of the room during the tests, thus skewing the results. We think most of the volunteers are at Apple now, working on their new "scratch and sniff" GUI. It is a funny old business we are in!

Seriously, both FreeBSD and Linux use the HLT (halt) instruction when the system is idle thus lowering its energy consumption and therefore the heat it generates. Also if you have APM (advanced power management) configured, then FreeBSD can also put the CPU into a low power mode.

2. Who is scratching in my memory banks??

Q. Is there anything "odd" that FreeBSD does when compiling the kernel which would cause the memory to make a scratchy sound? When compiling (and for a brief moment after recognizing the floppy drive upon startup, as well), a strange scratchy sound emanates from what appears to be the memory banks.

A. Yes! You will see frequent references to "daemons" in the BSD documentation, and what most people do not know is that this refers to genuine, non-corporeal entities that now possess your computer. The scratchy sound coming from your memory is actually high-pitched whispering exchanged among the daemons as they best decide how to deal with various system administration tasks.

If the noise gets to you, a good `fdisk /mbr` from DOS will get rid of them, but do not be surprised if they react adversely and try to stop you. In fact, if at any point during the exercise you hear the satanic voice of Bill Gates coming from the built-in speaker, take off running and do not ever look back! Freed from the counterbalancing influence of the BSD daemons, the twin demons of DOS and Windows are often able to re-assert total control over your machine to the eternal damnation of your soul. Now that you know, given a choice you would probably prefer to get used to the scratchy noises, no?

3. How many FreeBSD hackers does it take to change a lightbulb?

One thousand, one hundred and sixty-nine:

Twenty-three to complain to -CURRENT about the lights being out;

Four to claim that it is a configuration problem, and that such matters really belong on -questions;

Three to submit PRs about it, one of which is misfiled under doc and consists only of "it's dark";

One to commit an untested lightbulb which breaks buildworld, then back it out five minutes later;

Eight to flame the PR originators for not including patches in their PRs;

Five to complain about buildworld being broken;

Thirty-one to answer that it works for them, and they must have cvsupped at a bad time;

One to post a patch for a new lightbulb to -hackers;

One to complain that he had patches for this three years ago, but when he sent them to -CURRENT they were just ignored, and he has had bad experiences with the PR system; besides, the proposed new lightbulb is non-reflexive;

Thirty-seven to scream that lightbulbs do not belong in the base system, that committers have no right to do things like this without consulting the Community, and WHAT IS -CORE DOING ABOUT IT!?

Two hundred to complain about the color of the bicycle shed;

Three to point out that the patch breaks style(9);

Seventeen to complain that the proposed new lightbulb is under GPL;

Five hundred and eighty-six to engage in a flame war about the comparative advantages of the GPL, the BSD license, the MIT license, the NPL, and the personal hygiene of unnamed FSF founders;

Seven to move various portions of the thread to -chat and -advocacy;

One to commit the suggested lightbulb, even though it shines dimmer than the old one;

Two to back it out with a furious flame of a commit message, arguing that FreeBSD is better off in the dark than with a dim lightbulb;

Forty-six to argue vociferously about the backing out of the dim lightbulb and demanding a statement from -core;

Eleven to request a smaller lightbulb so it will fit their Tamagotchi if we ever decide to port FreeBSD to that platform;

Seventy-three to complain about the SNR on -hackers and -chat and unsubscribe in protest;

Thirteen to post "unsubscribe", "How do I unsubscribe?", or "Please remove me from the list", followed by the usual footer;

One to commit a working lightbulb while everybody is too busy flaming everybody else to notice;

Thirty-one to point out that the new lightbulb would shine 0.364% brighter if compiled with TenDRA (although it will have to be reshaped into a cube), and that FreeBSD should therefore switch to TenDRA instead of GCC;

One to complain that the new lightbulb lacks fairings;

Nine (including the PR originators) to ask "what is MFC?";

Fifty-seven to complain about the lights being out two weeks after the bulb has been changed.

Nik Clayton <`nik@FreeBSD.org`> adds:

I was laughing quite hard at this.

And then I thought, "Hang on, shouldn't there be '1 to document it.' in that list somewhere?"

And then I was enlightened :-)

Thomas Abthorpe <`tabthorpe@FreeBSD.org`> says: "None, *real* FreeBSD hackers are not afraid of the dark!"

4. Where does data written to `/dev/null` go?

It goes into a special data sink in the CPU where it is converted to heat which is vented through the heatsink / fan assembly. This is why CPU cooling is increasingly important; as people get used to faster processors, they become careless with their data and more and more of it ends up in `/dev/null`, overheating their CPUs. If you delete `/dev/null` (which effectively disables the CPU data sink) your CPU may run cooler but your system will quickly become constipated with all that excess data and start to behave erratically. If you have a fast network connection you

can cool down your CPU by reading data out of `/dev/random` and sending it off somewhere; however you run the risk of overheating your network connection and / or angering your ISP, as most of the data will end up getting converted to heat by their equipment, but they generally have good cooling, so if you do not overdo it you should be OK.

Paul Robinson adds:

There are other methods. As every good sysadmin knows, it is part of standard practice to send data to the screen of interesting variety to keep all the pixies that make up your picture happy. Screen pixies (commonly mis-typed or re-named as “pixels”) are categorized by the type of hat they wear (red, green or blue) and will hide or appear (thereby showing the color of their hat) whenever they receive a little piece of food. Video cards turn data into pixie-food, and then send them to the pixies — the more expensive the card, the better the food, so the better behaved the pixies are. They also need constant stimulation — this is why screen savers exist.

To take your suggestions further, you could just throw the random data to console, thereby letting the pixies consume it. This causes no heat to be produced at all, keeps the pixies happy and gets rid of your data quite quickly, even if it does make things look a bit messy on your screen.

Incidentally, as an ex-admin of a large ISP who experienced many problems attempting to maintain a stable temperature in a server room, I would strongly discourage people sending the data they do not want out to the network. The fairies who do the packet switching and routing get annoyed by it as well.

Chapter 18 Advanced Topics

1. How can I learn more about FreeBSD's internals?

At this time, there is only one book on FreeBSD-specific OS internals, namely "The Design and Implementation of the FreeBSD Operating System" by Marshall Kirk McKusick and George V. Neville-Neil, ISBN 0-201-70245-2, which focuses on version 5.*X* of FreeBSD.

Additionally, much general UNIX knowledge is directly applicable to FreeBSD.

For a list of relevant books, please check the Handbook's Operating System Internals Bibliography (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/bibliography-osinternals.html).

2. How can I contribute to FreeBSD?

Please see the article on Contributing to FreeBSD (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/contributing/article.html) for specific advice on how to do this. Assistance is more than welcome!

3. What are snapshots and releases?

There are currently three active/semi-active branches in the FreeBSD CVS Repository (http://www.FreeBSD.org/cgi/cvsweb.cgi). (Earlier branches are only changed very rarely, which is why there are only three active branches of development):

- `RELENG_7` AKA *7-STABLE*
- `RELENG_8` AKA *8-STABLE*
- `HEAD` AKA *-CURRENT* AKA *9-CURRENT*

`HEAD` is not an actual branch tag, like the other two; it is simply a symbolic constant for "*the current, non-branched development stream*" which we simply refer to as *-CURRENT*.

Right now, *-CURRENT* is the 9.*X* development stream; the `7-STABLE` branch, RELENG_7, forked off from *-CURRENT* in February 2008 and the `8-STABLE` branch, RELENG_8, forked off from *-CURRENT* in November 2009.

4. How do I make my own custom release?

Please see the Release Engineering (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/releng/article.html) article.

5. Why does `make world` clobber my existing installed binaries?

Yes, this is the general idea; as its name might suggest, `make world` rebuilds every system binary from scratch, so you can be certain of having a clean and consistent environment at the end (which is why it takes so long).

If the environment variable `DESTDIR` is defined while running `make world` or `make install`, the newly-created binaries will be deposited in a directory tree identical to the installed one, rooted at `${DESTDIR}`. Some random combination of shared libraries modifications and program rebuilds can cause this to fail in `make world` however.

6. Why isn't `cvsup.FreeBSD.org` a round robin DNS entry to share the load amongst the various **CVSup** servers?

While **CVSup** mirrors update from the master **CVSup** server hourly, this update might happen at any time during the hour. This means that some servers have newer code than others, even though all servers have code that is less than an hour old. If `cvsup.FreeBSD.org` was a round robin DNS entry that simply redirected users to a random **CVSup** server, running **CVSup** twice in a row could download code older than the code already on the system.

7. Can I follow *-CURRENT* with limited Internet access?

Yes, you can do this *without* downloading the whole source tree by using the CTM facility (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/synching.html#CTM).

8. How did you split the distribution into 1392 KB files?

Newer BSD based systems have a `-b` option to split(1) that allows them to split files on arbitrary byte boundaries.

Here is an example from `/usr/src/release/Makefile`.

```
ZIPNSPLIT=              gzip --no-name -9 -c | split -b 1392k -
```

9. I have written a kernel extension, who do I send it to?

Please take a look at the article on Contributing to FreeBSD (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/contributing/article.html) to learn how to submit code.

And thanks for the thought!

10. How are Plug N Play ISA cards detected and initialized?

By: Frank Durda IV <`uhclem@nemesis.lonestar.org`>

In a nutshell, there a few I/O ports that all of the PnP boards respond to when the host asks if anyone is out there. So when the PnP probe routine starts, it asks if there are any PnP boards present, and all the PnP boards respond with their model # to a I/O read of the same port, so the probe routine gets a wired-OR "yes" to that question. At least one bit will be on in that reply. Then the probe code is able to cause boards with board model IDs (assigned by Microsoft/Intel) lower than `X` to go "off-line". It then looks to see if any boards are still responding to the query. If the answer was `0`, then there are no boards with IDs above `X`. Probe will then ask for boards below `X`. Finally, probe requests boards greater than `X - (limit / 4)` to go off-line. It then repeats this query. By repeating this semi-binary search of IDs-in-range enough times, the probing code will eventually identify all PnP boards present in a given machine with a number of iterations that is much lower than what 2^{64} would take.

The IDs are two 32-bit fields (hence 2^{64}) + 8-bit checksum. The first 32 bits are a vendor identifier. They never come out and say it, but it appears to be assumed that different types of boards from the same vendor could have different 32-bit vendor IDs. The idea of needing 32 bits just for unique manufacturers is a bit excessive.

The lower 32 bits are a serial #, or something else that makes this one board unique. The vendor must never produce a second board that has the same lower 32 bits unless the upper 32 bits are also different. So you can have multiple boards of the same type in the machine and the full 64 bits will still be unique.

The 32-bit groups can never be all zero. This allows the wired-OR to show non-zero bits during the initial binary search.

Once the system has identified all the board IDs present, it will reactivate each board, one at a time (via the same I/O ports), and find out what resources the given board needs, what interrupt choices are available, etc. A scan is made over all the boards to collect this information.

This info is then combined with info from any ECU files on the hard disk or wired into the MLB BIOS. The ECU and BIOS PnP support for hardware on the MLB is usually synthetic, and the peripherals do not really do genuine PnP. However by examining the BIOS info plus the ECU info, the probe routines can cause the devices that are PnP to avoid those devices the probe code cannot relocate.

Then the PnP devices are visited once more and given their I/O, DMA, IRQ and Memory-map address assignments. The devices will then appear at those locations and remain there until the next reboot, although there is nothing that says you cannot move them around whenever you want.

There is a lot of oversimplification above, but you should get the general idea.

Microsoft took over some of the primary printer status ports to do PnP, on the logic that no boards decoded those addresses for the opposing I/O cycles. I found a genuine IBM printer board that did decode writes of the status port during the early PnP proposal review period, but Microsoft said “tough”. So they do a write to the printer status port for setting addresses, plus that use that address + `0x800`, and a third I/O port for reading that can be located anywhere between `0x200` and `0x3ff`.

11. Can you assign a major number for a device driver I have written?

FreeBSD releases after February 2003 has a facility for dynamically and automatically allocating major numbers for device drivers at runtime (see devfs(5)), so there is no need for this.

12. What about alternative layout policies for directories?

In answer to the question of alternative layout policies for directories, the scheme that is currently in use is unchanged from what I wrote in 1983. I wrote that policy for the original fast file system, and never revisited it. It works well at keeping cylinder groups from filling up. As several of you have noted, it works poorly for find. Most file systems are created from archives that were created by a depth first search (aka ftw). These directories end up being striped across the cylinder groups thus creating a worst possible scenario for future depth first searches. If one knew the total number of directories to be created, the solution would be to create `(total / fs_ncg)` per cylinder group before moving on. Obviously, one would have to create some heuristic to guess at this number. Even using a small fixed number like say 10 would make an order of magnitude improvement. To differentiate restores from normal operation (when the current algorithm is probably more sensible), you could use the clustering of up to 10 if they were all done within a ten second window. Anyway, my conclusion is that this is an area ripe for experimentation.

Kirk McKusick <`mckusick@FreeBSD.org`>, September 1998

13. How can I make the most of the data I see when my kernel panics?

Here is typical kernel panic:

```
Fatal trap 12: page fault while in kernel mode
fault virtual address   = 0x40
fault code              = supervisor read, page not present
instruction pointer     = 0x8:0xf014a7e5
stack pointer           = 0x10:0xf4ed6f24
frame pointer           = 0x10:0xf4ed6f28
code segment            = base 0x0, limit 0xfffff, type 0x1b
```

```
                        = DPL 0, pres 1, def32 1, gran 1
processor eflags        = interrupt enabled, resume, IOPL = 0
current process         = 80 (mount)
interrupt mask          =
trap number             = 12
panic: page fault
```

When you see a message like this, it is not enough to just reproduce it and send it in. The instruction pointer value is important; unfortunately, it is also configuration dependent. In other words, the value varies depending on the exact kernel image that you are using. If you are using a `GENERIC` kernel image from one of the snapshots, then it is possible for somebody else to track down the offending function, but if you are running a custom kernel then only *you* can tell us where the fault occurred.

What you should do is this:

1. Write down the instruction pointer value. Note that the `0x8:` part at the beginning is not significant in this case: it is the `0xf0xxxxxx` part that we want.
2. When the system reboots, do the following:

   ```
   % nm -n kernel.that.caused.the.panic | grep f0xxxxxx
   ```

   where `f0xxxxxx` is the instruction pointer value. The odds are you will not get an exact match since the symbols in the kernel symbol table are for the entry points of functions and the instruction pointer address will be somewhere inside a function, not at the start. If you do not get an exact match, omit the last digit from the instruction pointer value and try again, i.e.:

   ```
   % nm -n kernel.that.caused.the.panic | grep f0xxxxx
   ```

   If that does not yield any results, chop off another digit. Repeat until you get some sort of output. The result will be a possible list of functions which caused the panic. This is a less than exact mechanism for tracking down the point of failure, but it is better than nothing.

However, the best way to track down the cause of a panic is by capturing a crash dump, then using kgdb(1) to generate a stack trace on the crash dump.

In any case, the method is this:

1. Make sure that the following line is included in your kernel configuration file (`/usr/src/sys/`*`arch`*`/conf/`*`MYKERNEL`*):

   ```
   makeoptions     DEBUG=-g          # Build kernel with gdb(1) debug symbols
   ```

2. Change to the `/usr/src` directory:

   ```
   # cd /usr/src
   ```

3. Compile the kernel:

   ```
   # make buildkernel KERNCONF=MYKERNEL
   ```

4. Wait for make(1) to finish compiling.
5. `# make installkernel KERNCONF=MYKERNEL`
6. Reboot.

Note: If you do not use the `KERNCONF` make variable a `GENERIC` kernel will be built and installed.

The make(1) process will have built two kernels. `/usr/obj/usr/src/sys/MYKERNEL/kernel` and `/usr/obj/usr/src/sys/MYKERNEL/kernel.debug`. `kernel` was installed as `/boot/kernel/kernel`, while `kernel.debug` can be used as the source of debugging symbols for kgdb(1).

To make sure you capture a crash dump, you need edit `/etc/rc.conf` and set `dumpdev` to point to your swap partition (or `AUTO`). This will cause the rc(8) scripts to use the dumpon(8) command to enable crash dumps. You can also run dumpon(8) manually. After a panic, the crash dump can be recovered using savecore(8); if `dumpdev` is set in `/etc/rc.conf`, the rc(8) scripts will run savecore(8) automatically and put the crash dump in `/var/crash`.

> **Note:** FreeBSD crash dumps are usually the same size as the physical RAM size of your machine. That is, if you have 512 MB of RAM, you will get a 512 MB crash dump. Therefore you must make sure there is enough space in `/var/crash` to hold the dump. Alternatively, you run savecore(8) manually and have it recover the crash dump to another directory where you have more room. It is possible to limit the size of the crash dump by using `options MAXMEM=`*`N`* where *`N`* is the size of kernel's memory usage in KBs. For example, if you have 1 GB of RAM, you can limit the kernel's memory usage to 128 MB by this way, so that your crash dump size will be 128 MB instead of 1 GB.

Once you have recovered the crash dump, you can get a stack trace with kgdb(1) as follows:

```
% kgdb /usr/obj/usr/src/sys/MYKERNEL/kernel.debug /var/crash/vmcore.0
(kgdb) backtrace
```

Note that there may be several screens worth of information; ideally you should use script(1) to capture all of them. Using the unstripped kernel image with all the debug symbols should show the exact line of kernel source code where the panic occurred. Usually you have to read the stack trace from the bottom up in order to trace the exact sequence of events that lead to the crash. You can also use kgdb(1) to print out the contents of various variables or structures in order to examine the system state at the time of the crash.

> **Tip:** Now, if you are really insane and have a second computer, you can also configure kgdb(1) to do remote debugging such that you can use kgdb(1) on one system to debug the kernel on another system, including setting breakpoints, single-stepping through the kernel code, just like you can do with a normal user-mode program.

> **Note:** If you have `DDB` enabled and the kernel drops into the debugger, you can force a panic (and a crash dump) just by typing `panic` at the `ddb` prompt. It may stop in the debugger again during the panic phase. If it does, type `continue` and it will finish the crash dump.

14. Why has `dlsym()` stopped working for ELF executables?

The ELF toolchain does not, by default, make the symbols defined in an executable visible to the dynamic linker. Consequently `dlsym()` searches on handles obtained from calls to `dlopen(NULL, flags)` will fail to find such symbols.

If you want to search, using `dlsym()`, for symbols present in the main executable of a process, you need to link the executable using the `--export-dynamic` option to the ELF linker (ld(1)).

15. How can I increase or reduce the kernel address space on i386?

By default, the kernel address space is 1 GB (2 GB for PAE) for i386. If you run a network-intensive server (e.g. a large FTP or HTTP server), or you want to use ZFS, you might find that is not enough.

Add the following line to your kernel configuration file to increase available space and rebuild your kernel:

```
options KVA_PAGES=N
```

To find the correct value of `N`, divide the desired address space size (in megabytes) by four. (For example, it is `512` for 2 GB.)

Chapter 19 Acknowledgments

This innocent little Frequently Asked Questions document has been written, rewritten, edited, folded, spindled, mutilated, eviscerated, contemplated, discombobulated, cogitated, regurgitated, rebuilt, castigated, and reinvigorated over the last decade, by a cast of hundreds if not thousands. Repeatedly.

We wish to thank every one of the people responsible, and we encourage you to to join them (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/contributing/article.html) in making this FAQ even better.

Bibliography

FreeBSD Unleashed, Michael Urban and Brian Tiemann, Sams, 1st edition, 992 pages, October 2001, ISBN 0-67232-206-4.

4.4BSD System Manager's Manual, Computer Systems Research Group, University of California, Berkeley, O'Reilly and Associates, 1st edition, June 1994, 804 pages, ISBN 1-56592-080-5.

4.4BSD User's Reference Manual, Computer Systems Research Group, University of California, Berkeley, O'Reilly and Associates, 1st edition, June 1994, 905 pages, ISBN 1-56592-075-9.

4.4BSD User's Supplementary Documents, Computer Systems Research Group, University of California, Berkeley, O'Reilly and Associates, 1st edition, June 1994, 712 pages, ISBN 1-56592-076-7.

4.4BSD Programmer's Reference Manual, Computer Systems Research Group, University of California, Berkeley, O'Reilly and Associates, 1st edition, June 1994, 866 pages, ISBN 1-56592-078-3.

4.4BSD Programmer's Supplementary Documents, Computer Systems Research Group, University of California, Berkeley, O'Reilly and Associates, 1st edition, June 1994, 596 pages, ISBN 1-56592-079-1.

The Design and Implementation of the 4.4BSD Operating System, M. K. McKusick, Kirk Marshall, Keith Bostic, Michael J Karels, and John Quarterman, Addison-Wesley, Reading, 1996, ISBN 0-201-54979-4.

The Design and Implementation of the FreeBSD Operating System, M. K. McKusick and George V. Neville-Neil, Addison-Wesley, Boston, 2004, ISBN 0-201-70245-2.

Unix System Administration Handbook, Evi Nemeth, Garth Snyder, Scott Seebass, Trent R. Hein, and John Quarterman, Prentice-Hall, 3rd edition, 2000, ISBN 0-13-020601-6.

The Complete FreeBSD, Greg Lehey, Walnut Creek, 3rd edition, June 1999, 773 pages, ISBN 1-57176-246-9.

The FreeBSD Handbook, FreeBSD Documentation Project, BSDi, 1st edition, November 1999, 489 pages, ISBN 1-57176-241-8.

[McKusick et al, 1994] *Berkeley Software Architecture Manual, 4.4BSD Edition*, M. K. McKusick, M. J. Karels, S. J. Leffler, W. N. Joy, and R. S. Faber, 5:1-42.

FreeBSD for PC 98'ers (in Japanese), SHUWA System Co, LTD., ISBN 4-87966-468-5 C3055 P2900E.

FreeBSD (in Japanese), CUTT, ISBN 4-906391-22-2.

Complete Introduction to FreeBSD (in Japanese), Shoeisha Co., Ltd, ISBN 4-88135-473-6 P3600E.

Personal UNIX Starter Kit FreeBSD (in Japanese), ASCII, ISBN 4-7561-1733-3 P3000E.

FreeBSD Handbook (Japanese translation), ASCII, ISBN 4-7561-1580-2 P3800E.

FreeBSD mit Methode (in German), Computer und Literature Verlag/Vertrieb Hanser, 1998, ISBN 3-932311-31-0.

FreeBSD install and Utilization Manual (in Japanese), Mainichi Communications Inc..

Building Internet Server with FreeBSD (in Indonesia Language), Elex Media Komputindo, Onno W Purbo, Dodi Maryanto, Syahrial Hubbany, and Widjil Widodo.

The FreeBSD Corporate Networker's Guide, Addison-Wesley.

UNIX in a Nutshell, O'Reilly & Associates, Inc., 1990, ISBN 093717520X.

What You Need To Know When You Can't Find Your Unix System Administrator, O'Reilly & Associates, Inc., 1995, Linda Mui, ISBN 1-56592-104-6.

FreeBSD User's Reference Manual (Japanese translation), Mainichi Communications Inc., Jpman Project, Japan FreeBSD Users Group, 1998, ISBN 4-8399-0088-4 P3800E.

Online Guide for newcomers to the UNIX environment (http://unixhelp.ed.ac.uk/), Edinburgh University (http://www.ed.ac.uk/).

DNS and BIND, O'Reilly & Associates, Inc, ISBN 1-56592-512-2, Paul Albitz Albitz and Cricket Liu, 1998, 3rd edition.

Sendmail, O'Reilly & Associates, Inc, 1997, 2nd edition, Brian Costales, ISBN 1-56592-222-0.

Essential System Administration, Æleen Frisch, 2nd edition, O'Reilly & Associates, 1995, ISBN 1-56592-127-5.

TCP/IP Network Administration, Craig Hunt, 2nd edition, O'Reilly & Associates, Inc, 1997, ISBN 1-56592-322-7.

Managing NFS and NIS, Hal Stern, O'Reilly & Associates, Inc, 1991, ISBN 0-937175-75-7.

FreeBSD System Administration's Manual (http://www.pc.mycom.co.jp/FreeBSD/sam.html), Jpman Project, Japan FreeBSD Users Group (http://www.jp.FreeBSD.org), Mainichi Communications Inc. (http://www.pc.mycom.co.jp/), 1998, ISBN 4-8399-0109-0 P3300E.

X Window System Toolkit, Digital Press, Paul Asente, ISBN 1-55558-051-3.

C: A Reference Manual, Prentice Hall, 1995, 4th edition, Samuel P. Harbison and Guy L. Jr. Steele, ISBN 0-13-326224-3.

The C Programming Language, Prentice Hall, 1998, Brian Kernighan and Dennis Ritchie, ISBN 0-13-110362-9.

Porting UNIX Software, Greg Lehey, O'Reilly & Associates, Inc., 1995, ISBN 1-56592-126-7.

The Standard C Library, Prentice Hall, 1992, P. J. Plauger, ISBN 0-13-131509-9.

Advanced Programming in the UNIX Environment, Addison-Wesley, 1992, W. Richard Stevens, ISBN 0-201-56317-7.

UNIX Network Programming, W. Richard Stevens, Prentice Hall, 1998, 2nd edition, ISBN 0-13-490012-X.

Writing Serial Drivers for UNIX, Bill Wells, December 1994, Dr. Dobb's Journal, pp68-71, pp97-99.

UNIX System Architecture, Prentice-Hall, Inc, 1990, Prabhat K. Andleigh, ISBN 0-13-949843-5.

Porting UNIX to the 386, William Jolitz, Dr. Dobb's Journal, January 1991-July 1992.

TCP/IP Illustrated, Volume 1: The Protocols, W. Richard Stevens, Addison-Wesley, 1996, ISBN 0-201-63346-9.

Unix Systems for Modern Architectures, Addison-Wesley, Curt Schimmel, 1994, ISBN 0-201-63338-8.

TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP and the UNIX Domain Protocols, Addison-Wesley, 1996, W. Richard Stevens, ISBN 0-201-63495-3.

UNIX Internals -- The New Frontiers, Uresh Vahalia, Prentice Hall, 1996, ISBN 0-13-101908-2.

TCP/IP Illustrated, Volume 2: The Implementation, Gary R. Wright and W. Richard Stevens, 1995, Addison-Wesley, ISBN 0-201-63354-X.

Firewalls and Internet Security: Repelling the Wily Hacker, William R. CHeswick and Steven M. Bellovin, Addison-Wesley, 1995, ISBN 0-201-63357-4.

Practical UNIX Security, Simson Garfinkel and Gene Spafford, 1996, 2nd edition, O'Reilly & Associates, Inc, ISBN 1-56592-148-8.

PGP Pretty Good Privacy, Simson Garfinkel, O'Reilly & Associates, Inc, 1995, ISBN 1-56592-098-8.

Pentium Processor System Architecture, Don Anderson and Tom Shanley, Addison-Wesley, 1995, 2nd edition, ISBN 0-201-40992-5.

Programmer's Guide to the EGA, VGA, and Super VGA Cards, Richard F. Ferraro, 3rd edition, Addison-Wesley, 1995, ISBN 0-201-62490-7.

80486 System Architecture, Tom Shanley, Addison-Wesley, 1995, 3rd edition, ISBN 0-201-40994-1.

ISA System Architecture, Tom Shanley, Addison-Wesley, 3rd edition, 1995, ISBN 0-201-40996-8.

PCI System Architecture, Tom Shanley, Addison-Wesley, 1995, 3rd edition, ISBN 0-201-40993-3.

The Undocumented PC, Frank Van Gilluwe, Addison-Wesley, 1994, ISBN 0-201-62277-7.

Bell System Technical Journal, Unix Time-Sharing System, American Telephone & Telegraph Company, July-August 1978, Vol 57, No 6, Part 2, ISSN0005-8580.

Lion's Commentary on UNIX, John Lion, ITP Media Group, 1996, 6th edition, ISBN 1573980137.

The New Hacker's Dictionary, Eric S. Raymond, MIT Press, 1996, 3rd edition, ISBN 0-262-68092-0.

A quarter century of UNIX, Peter H. Salus, Addison-Wesley, 1994, ISBN 0-201-54777-5.

The UNIX-HATERS Handbook, Steven Strassman, Daniel Weise, and Simon Garfinkel, IDG Books Worldwide, Inc, 1994, ISBN 1-56884-203-1.

Life with UNIX — special edition, Don Libes and Sandy Ressler, Prentice-Hall, 1989, ISBN 0-13-536657-7.

The BSD Family Tree (ftp://ftp.uk.FreeBSD.org/pub/FreeBSD/FreeBSD-current/src/share/misc/bsd-family-tree), 1997.

Absolute BSD, Michael Lucas, No Starch Press, June 2002, ISBN 1-886411-74-3.

The C/C++ Users Journal, R&D Publications Inc., ISSN 1075-2838.

Sys Admin — The Journal for UNIX System Administrators, Miller Freeman, Inc, ISSN 1061-2688.