Frequently Asked Questions for FreeBSD 8.X, 9.X and 10.X

Frequently Asked Questions for FreeBSD 8.X, 9.X and 10.X

Revision: 45497
Copyright © 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014 The FreeBSD Documentation Project

Abstract

This is the Frequently Asked Questions FAQ for FreeBSD versions 8.*X*, 9.*X* and 10.*X*. 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, send them to the FreeBSD documentation project mailing list.

The latest version of this document is always available from the FreeBSD website. It may also be downloaded as one large HTML file with HTTP or as a variety of other formats from the FreeBSD FTP server.

Copyright

Redistribution and use in source (XML DocBook) and 'compiled' forms (XML, HTML, PDF, PostScript, RTF and so forth) with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code (XML DocBook) must retain the above copyright notice, this list of conditions and the following disclaimer as the first lines of this file unmodified.

2. Redistributions in compiled form (transformed to other DTDs, converted to PDF, PostScript, RTF and other formats) must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Important

THIS DOCUMENTATION IS PROVIDED BY THE FREEBSD DOCUMENTATION PROJECT "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE FREEBSD DOCUMENTATION PROJECT BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENTATION, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

FreeBSD is a registered trademark of the FreeBSD Foundation.

Adobe, Acrobat, Acrobat Reader, Flash and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

IBM, AIX, OS/2, PowerPC, PS/2, S/390, and ThinkPad are trademarks of International Business Machines Corporation in the United States, other countries, or both.

IEEE, POSIX, and 802 are registered trademarks of Institute of Electrical and Electronics Engineers, Inc. in the United States.

Intel, Celeron, Centrino, Core, EtherExpress, i386, i486, Itanium, Pentium, and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Microsoft, IntelliMouse, MS-DOS, Outlook, Windows, Windows Media and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

NetBSD is a registered trademark of the NetBSD Foundation.

Motif, OSF/1, and UNIX are registered trademarks and IT DialTone and The Open Group are trademarks of The Open Group in the United States and other countries.

Silicon Graphics, SGI, and OpenGL are registered trademarks of Silicon Graphics, Inc., in the United States and/or other countries worldwide.

Sun, Sun Microsystems, Java, Java Virtual Machine, JDK, JRE, JSP, JVM, Netra, OpenJDK, Solaris, StarOffice, SunOS and VirtualBox are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and the FreeBSD Project was aware of the trademark claim, the designations have been followed by the “™” or the “®” symbol.

Table of Contents

1. Introduction 1
2. Documentation and Support 7
3. Installation 13
4. Hardware Compatibility 17
    4.1. General 17
    4.2. Architectures and Processors 18
    4.3. Hard Drives, Tape Drives, and CD and DVD Drives 19
    4.4. Keyboards and Mice 20
    4.5. Other Hardware 21
5. Troubleshooting 23
6. User Applications 31
7. Kernel Configuration 35
8. Disks, File Systems, and Boot Loaders 37
9. ZFS 45
10. System Administration 47
11. The X Window System and Virtual Consoles 55
12. Networking 63
13. Security 67
14. PPP 71
15. Serial Communications 81
16. Miscellaneous Questions 85
17. Advanced Topics 89
18. Acknowledgments 95
Bibliography 97

Chapter 1. Introduction

Q: What is FreeBSD?

A: FreeBSD is a modern operating system for desktops, laptops, servers, and embedded systems with support for a large number of platforms.

It is 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.

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, refer to the FreeBSD Handbook.

Q: What is the goal of the FreeBSD Project?

A: The goal of the FreeBSD Project is to provide a stable and fast general purpose operating system that may be used for any purpose without strings attached.

Q: Does the FreeBSD license have any restrictions?

A: Yes. Those restrictions do not control how the code is used, but how to treat the FreeBSD Project itself. The license itself is available at license and can be summarized like this:

- Do not claim that you wrote this.
- Do not sue us if it breaks.
- Do not remove or modify the license.

Many of us have a significant investment in the 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, we believe, is one of the most fundamental goals of Free Software and one that we enthusiastically support.

Code in our source tree which falls under the GNU General Public License (GPL) or GNU Library General Public License (LGPL) 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 whenever possible.

Q: Can FreeBSD replace my current operating system?

A: 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 much more. Most of these applications can be managed through the Ports Collection.

If an application is only available on one operating system, that operating system cannot just be replaced. Chances are, there is a very similar application on FreeBSD, however. As a solid office or Internet server or a reliable workstation, 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.

Users migrating to FreeBSD from another UNIX®-like environment will find FreeBSD to be similar. Windows® and Mac OS® users may be interested in instead using PC-BSD, a FreeBSD-based desktop distribution. Non-UNIX® users should expect to invest some additional time learning the UNIX® way of doing things. This FAQ and the FreeBSD Handbook are excellent places to start.

Q: Why is it called FreeBSD?

A:
- 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” and the other meaning “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.

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

A: James Howard wrote a good explanation of the history and differences between the various projects, called The BSD Family Tree which goes a fair way to answering this question. Some of the information is out of date, but the history portion in particular remains accurate.

Most of the BSDs share patches and code, even today. All of the BSDs have common ancestry.

The design goals of FreeBSD are described in Q:, above. The design goals of the other most popular BSDs may be summarized as follows:

- OpenBSD aims for operating system security above all else. The OpenBSD team wrote ssh(1) and pf(4), which have both been ported to FreeBSD.
- NetBSD aims to be easily ported to other hardware platforms.
- DragonFly BSD is a fork of FreeBSD 4.8 that has since developed many interesting features of its own, including the HAMMER file system and support for user-mode "vkernels".

Q: What is the latest version of FreeBSD?

A: At any point in the development of FreeBSD, there can be multiple parallel branches. 10.*X* releases are made from the *10-STABLE* branch, and 9.*X* releases are made from the *9-STABLE* branch.

Up until the release of 9.0, the 9.*X* series was the one known as *-STABLE*. However, as of 11.*X*, the 9.*X* branch will be designated for an "extended support" status and receive only fixes for major problems, such as security-related fixes.

Version 10.0 is the latest release from the *10-STABLE* branch; it was released in January 2014. Version 9.3 is the latest release from the *9-STABLE* branch; it was released in July 2014.

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* is meant for users who are prepared for its increased volatility, relative to *-STABLE*.

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 and in release(7).

Q: What is *FreeBSD-CURRENT*?

A: FreeBSD-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 in the Handbook for details on running *-CURRENT*.

Users not familiar with FreeBSD should not use FreeBSD-CURRENT. This branch sometimes evolves quite quickly and due to mistake can be un-buildable at times. People that use FreeBSD-CURRENT are expected to be able to analyze, debug, and report problems.

FreeBSD snapshot 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 Subversion 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 a stable and fully tested system is needed, stick to full releases or use the *-STABLE* snapshots.

Snapshot releases are directly available from snapshot.

Official snapshots are generated on a regular basis for all actively developed branches.

Q: What is the *FreeBSD-STABLE* concept?

A: Back when FreeBSD 2.0.5 was released, FreeBSD development branched in two. One branch was named -STABLE, one -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 10.0-RELEASE and beyond. For more detailed information on branches see "FreeBSD Release Engineering: Creating the Release Branch", the status of the branches and the upcoming release schedule can be found on the Release Engineering Information page.

10.0-STABLE is the actively developed *-STABLE* branch. The latest release on the 10.0-STABLE branch is 10.0-RELEASE, which was released in January 2014.

The *11-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.

Q: When are FreeBSD releases made?

A: 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, 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 pages on the FreeBSD Web site.

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

Q: Who is responsible for FreeBSD?

A: 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 of 9 people. There is a much larger team of more than 350 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.

Q: Where can I get FreeBSD?

A: Every significant release of FreeBSD is available via anonymous FTP from the FreeBSD FTP site:

- The latest *10-STABLE* release, 10.0-RELEASE can be found in the 10.0-RELEASE directory.
- Snapshot releases are made monthly for the -CURRENT and -STABLE branch, these being of service purely to bleeding-edge testers and developers.
- The latest *9-STABLE* release, 9.3-RELEASE can be found in the 9.3-RELEASE directory.

Information about obtaining FreeBSD on CD, DVD, and other media can be found in the Handbook.

Q: How do I access the Problem Report database?

A: The Problem Report database of all user change requests may be queried by using our web-based PR query interface.

The web-based problem report submission interface can be used to submit problem reports through a web browser.

Before submitting a problem report, read Writing FreeBSD Problem Reports, an article on how to write good problem reports.

Chapter 2. Documentation and Support

Q: What good books are there about FreeBSD?

A: 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 reference other recommended books.

Q: Is the documentation available in other formats, such as plain text (ASCII), or PostScript®?

A: Yes. The documentation is available in a number of different formats and compression schemes on the FreeBSD FTP site, in the /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 found under `/usr/share/locale` on a FreeBSD system. The current languages and encodings 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)

Name	Meaning
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.UTF-8	Simplified Chinese (China, UTF-8 encoding)
zh_TW.UTF-8	Traditional Chinese (Taiwan, UTF-8 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.

  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. For example, `article.pdf` , `book.html` , and so on.

     These files are then compressed using either the `zip` or `bz2` compression schemes. tar(1) can be used 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, download the compressed files, 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 `doc/en_US.ISO8859-1/books/faq/book.html-split.tar.bz2` To download and uncompress that file, type:

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

If the file is compressed, tar will automatically detect the appropriate format and decompress it correctly, resulting in 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.

Q: Where do I find info on the FreeBSD mailing lists? What FreeBSD news groups are available?

A: Refer to the Handbook entry on mailing-lists and the Handbook entry on newsgroups.

Q: Are there FreeBSD IRC (Internet Relay Chat) channels?

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

- Channel `#FreeBSDhelp` on EFNet is a channel dedicated to helping FreeBSD users. They are much more sympathetic to questions than `#FreeBSD` is.

- Channel `#FreeBSD` on Freenode 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. Other users can help with the basics, referring to the Handbook whenever possible and providing links for learning more about a particular topic. This is primarily an English speaking channel, though it does have users from all over the world. Non-native English speakers should try to ask the question in English first and then relocate to `##freebsd-lang` as appropriate.

- Channel `#FreeBSD` on DALNET is available at `irc.dal.net` in the US and `irc.eu.dal.net` in Europe.

- Channel `#FreeBSD` on UNDERNET 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 is a Russian language channel dedicated to helping FreeBSD users. This is also good place for non-technical discussions.

- Channel `#bsdchat` on Freenode is a Traditional Chinese (UTF-8 encoding) language channel dedicated to helping FreeBSD users. This is also good place for non-technical discussions.

The FreeBSD wiki has a good list of IRC channels.

Each of these channels are distinct and are not connected to each other. Since their chat styles differ, try each to find one suited to your chat style.

Q: Are there any web based forums to discuss FreeBSD?

A: The official FreeBSD forums are located at https://forums.FreeBSD.org/.

Q: Where can I get commercial FreeBSD training and support?

A: iXsystems, Inc., parent company of the FreeBSD Mall, provides commercial FreeBSD and PC-BSD software support, in addition to FreeBSD development and tuning solutions.

BSD Certification Group, Inc. provides system administration certifications for DragonFly BSD, FreeBSD, NetBSD, and OpenBSD. Refer to their site for more information.

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

Chapter 3. Installation

Nik Clayton

<nik@FreeBSD.org>

Q: Which platform should I download? I have a 64 bit capable Intel® CPU, but I only see `amd64`.

A: amd64 is the term FreeBSD uses for 64-bit compatible x86 architectures (also known as "x86-64" or "x64"). Most modern computers should use amd64. Older hardware should use i386. When installing on a non-x86-compatible architecture, select the platform which best matches the hardware.

Q: Which file do I download to get FreeBSD?

A: On the Getting FreeBSD page, select `[iso]` next to the architecture that matches the hardware.

Any of the following can be used:

file	description
`disc1.iso`	Contains enough to install FreeBSD and a minimal set of packages.
`dvd1.iso`	Similar to `disc1.iso` but with additional packages.
`memstick.img`	A bootable image sufficient for writing to a USB stick.
`bootonly.iso`	A minimal image that requires network access during installation to completely install FreeBSD.

pc98 users require these floppy images: `floppies/boot.flp`, `floppies/kern1.flp`, `floppies/kern2.flp`, and `floppies/mfsroot1.flp`. These images need to be written onto floppies by tools like dd(1).

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.

Q: What do I do if the install image does not boot?

A: This can be caused by not downloading the 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 SHA-256 checksum of

the downloaded boot image: if it is not *exactly* that on the server, then the download process is suspect.

When using a command line FTP client, type *binary* at the FTP command prompt after getting connected to the server and before starting the download of the image.

Q: Where are the instructions for installing FreeBSD?

A: Installation instructions for versions since FreeBSD 9.0 can be found at Handbook entry on installing FreeBSD. Older instructions can be found in the legacy entry on installing FreeBSD.

Q: What are the minimum requirements to run FreeBSD?

A: FreeBSD requires a 486 or better PC, 64 MB or more of RAM, and at least 1.1 GB of hard disk space.

Q: How can I make my own custom release or install disk?

A: Customized FreeBSD installation media can be created by building a custom release. Follow the instructions in the Release Engineering article.

Q: Can Windows® co-exist with FreeBSD?

A: If Windows® is installed first, then yes. FreeBSD's boot manager will then manage to boot Windows® and FreeBSD. If Windows® is installed afterwards, it will overwrite the boot manager. If that happens, see the next section.

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

A: This depends upon the boot manager. The FreeBSD boot selection menu can be reinstalled using boot0cfg(8). For example, to restore the boot menu onto the disk *ada0*:

```
# boot0cfg -B ada0
```

The non-interactive MBR bootloader can be installed using gpart(8):

```
# gpart bootcode -b /boot/mbr ada0
```

For more complex situations, including GPT disks, see gpart(8).

Q: I booted from a CD, but the install program says no CD-ROM is found. Where did it go?

A: 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 the 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.

Q: Do I need to install the source?

A: In general, no. There is nothing in the base system which requires the presence of the source to operate. Some ports, like sysutils/lsof, will not build unless the source is installed. In particular, if the port builds a kernel module or directly operates on kernel structures, the source must be installed.

Q: Do I need to build a kernel?

A: Usually not. The supplied `GENERIC` kernel contains the drivers an ordinary computer will need. freebsd-update(8), the FreeBSD binary upgrade tool, cannot upgrade custom kernels, another reason to stick with the `GENERIC` kernel when possible. For computers with very limited RAM, such as embedded systems, it may be worthwhile to build a smaller custom kernel containing just the required drivers.

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

A: FreeBSD 7 and 8 use MD5 password hashing by default. Recent versions of FreeBSD use *SHA512* by default. 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 for backwards compatibility with legacy operating systems which still use the less secure password format. FreeBSD also supports the Blowfish and MD5 password formats. 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.

Q: What are the limits for memory?

A: 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).

Q: What are the limits for FFS file systems?

A: For FFS file systems, the largest file system is practically limited by the amount of memory required to fsck(8) the file system. fsck(8) requires one bit per fragment,

which with the default fragment size of 4 KB equates to 32 MB of memory per TB of disk. This does mean that on architectures which limit userland processes to 2 GB (e.g., i386™), the maximum fsck(8)'able filesystem is ~60 TB.

If there was not a fsck(8) memory limit the maximum filesystem size would be 2 ^ 64 (blocks) * 32 KB => 16 Exa * 32 KB => 512 ZettaBytes.

The maximum size of a single FFS file is approximately 2 PB with the default block size of 32 KB. Each 32 KB block can point to 4096 blocks. With triple indirect blocks, the calculation is 32 KB * 12 + 32 KB * 4096 + 32 KB * 4096^2 + 32 KB * 4096^3. Increasing the block size to 64 KB will increase the max file size by a factor of 16.

Q: Why do I get an error message, readin failed after compiling and booting a new kernel?

A: The world and kernel are out of sync. This is not supported. Be sure to use `make buildworld` and `make buildkernel` to update the kernel.

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

Q: Is there a tool to perform post-installation configuration tasks?

A: Yes, head/ users can set `WITH_BSDCONFIG` in `/etc/src.conf` . Users of 10.*X* and higher may also install sysutils/bsdconfig.

Chapter 4. Hardware Compatibility

4.1. General

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

A: This is discussed continually on the FreeBSD mailing lists but is to be expected since hardware changes so quickly. Read through the Hardware Notes for FreeBSD 10.0 or 9.3 and search the mailing list archives before asking about the latest and greatest hardware. Chances are a discussion about that type of hardware took place just last week.

Before purchasing a laptop, check the archives for FreeBSD laptop computer mailing list and FreeBSD general questions mailing list, or possibly a specific mailing list for a particular hardware type.

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

A: 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.

Q: Why does FreeBSD report less than 4 GB memory when installed on an i386™ machine?

A: 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. When creating a custom kernel configuration file, PAE can be enabled by adding the following line:

```
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.

4.2. Architectures and Processors

Q: Does FreeBSD support architectures other than the x86?

A: Yes. FreeBSD divides support into multiple tiers. Tier 1 architectures, such as i386 or amd64; are fully supported. Tiers 2 and 3 are supported on an if-possible basis. A full explanation of the tier system is available in the Committer's Guide.

A complete list of supported architectures can be found on the platforms page.

Q: Does FreeBSD support Symmetric Multiprocessing (SMP)?

A: FreeBSD supports symmetric multi-processor (SMP) on all non-embedded platforms (e.g, i386, amd64, etc.). SMP is also supported in arm and MIPS kernels, although some CPUs may not support this. FreeBSD's SMP implementation uses fine-grained locking, and performance scales nearly linearly with number of CPUs.

smp(4) has more details.

Q: What is microcode? How do I install Intel® CPU microcode updates?

A: Microcode is a method of programmatically implementing hardware level instructions. This allows for CPU bugs to be fixed without replacing the on board chip.

Install sysutils/devcpu-data, then add:

```
microcode_update_enable="YES"
```

to `/etc/rc.conf`

4.3. Hard Drives, Tape Drives, and CD and DVD Drives

Q: What kind of hard drives does FreeBSD support?

A: 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.

Q: Which SCSI or SAS controllers are supported?

A: See the complete list in the Hardware Notes for FreeBSD 10.0 or 9.3.

Q: What types of tape drives are supported?

A: FreeBSD supports all standard SCSI tape interfaces.

Q: Does FreeBSD support tape changers?

A: FreeBSD supports SCSI changers using the ch(4) device and the chio(1) command. The details of how to control the changer can be found in chio(1).

While AMANDA and some other products already understands changers, other applications only know how to move a tape from one point to another/ In this case, keep track of which slot a tape is in and which slot the tape currently in the drive needs to go back to.

Q: Which CD-ROM drives are supported by FreeBSD?

A: Any SCSI drive connected to a supported controller is supported. Most ATAPI compatible IDE CD-ROMs are supported.

Q: Which CD-RW drives are supported by FreeBSD?

A: 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 the sysutils/cdrtools port or package, then use `cdrecord` .

4.4. Keyboards and Mice

Q: Is it possible to use a mouse outside the X Window system?

A: The default console driver, syscons(4), provides the ability to 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.

For a PS/2 mouse, add `moused_enable="YES"` to `/etc/rc.conf` to start the mouse daemon at boot time. Additionally, 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.

Q: How do I cut and paste text with a mouse in the text console?

A: It is not possible to remove data using the mouse. However, it is possible to copy and paste. Once the mouse daemon is running as described in the previous question, hold down button 1 (left button) and move the mouse to select a region of text. Then, press button 2 (middle button) to paste it at the text cursor. Pressing button 3 (right button) will "extend" the selected region of text.

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

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

A: 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.

Q: How do I use my delete key in `sh` and `csh`?

A: For the Bourne Shell, add the following lines to `~/.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 `~/.cshrc`. See csh(1).

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

For more information, see this page.

4.5. Other Hardware

Q: Workarounds for no sound from my pcm(4) sound card?

A: 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
```

Q: Does FreeBSD support power management on my laptop?

A: FreeBSD supports the ACPI features found in modern hardware. Further information can be found in acpi(4).

Chapter 5. Troubleshooting

Q: Why is FreeBSD finding the wrong amount of memory on i386™ hardware?

A: 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 cannot be accessed by that address space.

What happens to the memory that should appear in that location is hardware dependent. 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 when watching 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.

Q: Why do my programs occasionally die with Signal 11 errors?

A: Signal 11 errors are caused when a 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, start investigating the cause.

These problems can usually be attributed to either:

1. If the problem is occurring only in a specific custom application, it is probably a bug in the 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).

It is probably not a FreeBSD bug if the problem occurs compiling a program, but the activity that the compiler is carrying out changes each time.

For example, if `make buildworld` fails while trying to compile `ls.c` into `ls.o` and, when run again, it fails in the same place, this is a broken build. Try updating source and try again. If the compile fails elsewhere, it is almost certainly due to hardware.

In the first case, use a debugger such as gdb(1) to find the point in the program which is attempting to access a bogus address and fix it.

In the second case, verify which piece of hardware is at fault.

Common causes of this include:

1. The hard disks might be overheating: Check that the fans are still working, as the disk and 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, ensure that the hardware is running at what it is specified to run at, at least while trying to solve this problem. If it is not, clock it back to the default settings.)

   Regarding overclocking, it is far cheaper to have a slow system than a fried system that needs replacing! Also the community is not sympathetic to problems on overclocked systems.

3. Dodgy memory: if multiple memory SIMMS/DIMMS are installed, pull them all out and try running the machine with each SIMM or DIMM individually to narrow the problem down to either the problematic DIMM/SIMM or perhaps even a combination.

4. Over-optimistic motherboard settings: the BIOS settings, and some motherboard jumpers, provide options to set various timings. The defaults are often sufficient, but sometimes setting the wait states on RAM too low, or setting the "RAM Speed: Turbo" option will cause strange behavior. A possible idea is to set to BIOS defaults, after noting the current settings first.

5. Unclean or insufficient power to the motherboard. Remove any unused I/O boards, hard disks, or CD-ROMs, or disconnect the power cable from them, to see if the power supply can manage a smaller load. Or try another power supply, preferably one with a little more power. For instance, if the current power supply is rated at 250 Watts, try one rated at 300 Watts.

Read the section on Signal 11 for a further explanation and a discussion on how memory testing software or hardware can still pass faulty memory. There is an extensive FAQ on this at the SIG11 problem FAQ.

Finally, if none of this has helped, it is possibly a bug in FreeBSD. Follow these instructions to send a problem report.

Q: 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?

A: The FreeBSD developers are interested in these errors, but need more information than just the error message. Copy the full crash message. Then consult the FAQ section on kernel panics, build a debugging kernel, and get a backtrace. This might sound difficult, but does not require any programming skills. Just follow the instructions.

Q: What is the meaning of the error maxproc limit exceeded by uid %i, please see tuning(7) and login.conf(5)?

A: 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. If the machine is heavily loaded, increase `kern.maxusers` . This will increase these other system limits in addition to the maximum number of processes.

To adjust the `kern.maxusers` value, see the File/Process Limits section of the Handbook. While that section refers to open files, the same limits apply to processes.

If the machine is lightly loaded but running a very large number of processes, adjust the `kern.maxproc` tunable by defining it in `/boot/loader.conf` . The tunable will not get adjusted until the system is rebooted. For more information about tuning tunables, see loader.conf(5). If these processes are being run by a single user, adjust `kern.maxprocperuid` to be one less than the new `kern.maxproc` value. It must be at least one less because one system program, init(8), must always be running.

Q: Why does Sendmail give me an error reading mail loops back to myself?

A: This is answered in the Sendmail FAQ as follows. This FAQ is recommended reading when “tweaking” the mail setup.

```
I'm getting these error messages:

553 MX list for domain.net points back to relay.domain.net
554 <user@domain.net>... Local configuration error

How can I solve this problem?

You have asked mail to the domain (e.g., domain.net) to be
forwarded to a specific host (in this case, relay.domain.net)
by using an MX record, but the relay machine does not recognize
itself as domain.net. Add domain.net to /etc/mail/local-host-
names
[known as /etc/sendmail.cw prior to version 8.10]
(if you are using FEATURE(use_cw_file)) or add “Cw domain.net”
to /etc/mail/sendmail.cf.
```

Q: Why do full screen applications on remote machines misbehave?

A: The remote machine may be setting the 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 the `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 provides the ability to run multiple concurrent sessions from one terminal. 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. Refer to the system administration manuals for the remote system.
- 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`.

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

A: 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 to store the hostname in a log file for future reference by the administrator.

The remedy: if the problem occurs whenever connecting the client computer to any server, the problem is with the client. If the problem only occurs when someone connects to the server computer, 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. If this is on the Internet, contact your ISP.

If the problem is with the server on a local network, configure the server to resolve address-to-hostname queries for the local address range. See hosts(5) and named(8) for more information. If this is on the Internet, the problem may be that the local server's resolver is not functioning correctly. To check, try to look up another host such as `www.yahoo.com` . If it does not work, that is the 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 `/etc/ssh/sshd_config`. If this is causing the problem, either fill in the missing information in `/etc/resolv.conf` or set `UseDNS` to `no` in `sshd_config` as a temporary workaround.

Q: Why does file: table is full show up repeatedly in dmesg(8)?

A: This error message indicates that the number of available file descriptors have been exhausted on the system. Refer to the kern.maxfiles section of the Tuning Kernel Limits section of the Handbook for a discussion and solution.

Q: Why does the clock on my computer keep incorrect time?

A: The 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
```

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 `/boot/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 kern.timecounter.hardware=i8254
kern.timecounter.hardware: TSC -> i8254
```

The 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
```

Q: What does the error swap_pager: indefinite wait buffer: mean?

A: 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 bad, disk errors will appear in `/var/log/messages` and in the output of `dmesg`. Otherwise, check the cables and connections.

Q: What is a lock order reversal?

A: The FreeBSD kernel uses a number of resource locks to arbitrate contention for certain resources. When multiple kernel threads try to obtain multiple resource locks, there's always the potential for a deadlock, where two threads have each obtained one of the locks and blocks forever waiting for the other thread to release one of the other locks. This sort of locking problem can be avoided if all threads obtain the locks in the same order.

A run-time lock diagnostic system called witness(4), enabled in FreeBSD-CURRENT and disabled by default for stable branches and releases, detects the potential for deadlocks due to locking errors, including errors caused by obtaining multiple resource locks with a different order from different parts of the kernel. The witness(4) framework tries to detect this problem as it happens, and reports it by printing a message to the system console about a lock order reversal (often referred to also as LOR).

It is possible to get false positives, as witness(4) is conservative. A true positive report *does not* mean that a system is dead-locked; instead it should be understood as a warning that a deadlock could have happened here.

Note

Problematic LORs tend to get fixed quickly, so check http://lists.FreeBSD.org/mailman/listinfo/freebsd-current before posting to the mailing lists.

Q: What does Called ... with the following non-sleepable locks held mean?

A: 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 Free-

BSD) 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.

For additional information about locking in FreeBSD see locking(9).

Q: Why does `buildworld/installworld` die with the message touch: not found?

A: 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 the CMOS clock is set to local time, run `adjkerntz -i` to adjust the kernel clock when booting into single-user mode.

Chapter 6. User Applications

Q: So, where are all the user applications?

A: Refer to the ports page for info on software packages ported to FreeBSD. The list currently tops 24,000 and is growing daily, so come back to check often or subscribe to the FreeBSD announcements mailing list for periodic updates on new entries.

Most ports should work on the 8.*X*, 9.*X*, and 10.*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.

FreeBSD supports packages, which provide a compressed binary distribution. Package can be installed and uninstalled again easily. On FreeBSD, pkg(7) to install packages.

Q: How do I download the Ports tree? Should I be using SVN?

A: Any of the methods listed here work:

- Use portsnap for most use cases. Refer to Using the Ports Collection for instructions on how to use this tool.
- Use SVN if custom patches to the ports tree are needed. Refer to Using Subversion for details.
- Use CTM, as described in Using CTM to receive patches by email over an unreliable Internet connection.

Q: Does FreeBSD support Java™?

A: Yes. Refer to http://www.FreeBSD.org/java/ for more information.

Q: Why can I not build this port on my 9.*X* -, or 10.*X* -STABLE machine?

A: If the installed FreeBSD version lags significantly behind -*CURRENT* or -*STABLE*, update the Ports Collection using the instructions in Using the Ports Collection. If the system is up-to-date, someone might have committed a change to the port which works for -*CURRENT* but which broke the port for -*STABLE*. Submit a bug report, since the Ports Collection is supposed to work for both the -*CURRENT* and -*STABLE* branches.

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

A: First, make sure that the Ports Collection is up-to-date. 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.

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, try to make `INDEX` with those make variables turned off before reporting it to FreeBSD ports mailing list.

Q: I updated the sources, now how do I update my installed ports?

A: FreeBSD does not include a port upgrading tool, but it does have some tools to make the upgrade process somewhat easier. Additional tools are available to simplify port handling and are described the Upgrading Ports section in the FreeBSD Handbook.

Q: Do I need to recompile every port each time I perform a major version update?

A: Yes! While a recent system will run with software compiled under an older release, things will randomly crash and fail to work once other ports are installed or updated.

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 in the FreeBSD Handbook.

Q: Do I need to recompile every port each time I perform a minor version update?

A: 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.

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

A: 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 (sh(1)), 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. To keep `/bin/sh` small, we have not provided many of the convenience features that other shells have. That is why other more featureful shells like `bash`, `scsh`, tcsh(1), and `zsh` are available. Compare the memory utilization of these shells by looking at the “VSZ” and “RSS” columns in a `ps -u` listing.

Q: How do I create audio CDs from my MIDI files?

A: 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.

Q: Where can I get an Office Suite for FreeBSD?

A: The open source Apache OpenOffice and LibreOffice office suites work natively on FreeBSD.

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

Q: Can I convert from pkgng to the old package tools?

A: Short answer: no.

Longer answer: converting back is non-trivial and requires lots of manual editing of internal package database files. However, if `pkg2ng` has just been run, remove `/var/db/pkg/local.sqlite` and extract `/var/backups/pkgdb.bak.tbz` .

Chapter 7. Kernel Configuration

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

A: Not at all! Check out the kernel config section of the Handbook.

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. If a mistake is made in the configuration, simply boot the previous version of the kernel.

Q: Why is my kernel so big?

A: `GENERIC` kernels shipped with FreeBSD and later are compiled in *debug mode*. Kernels built in debug mode contain many symbols in separate files that are used for debugging, thus greatly increasing the size of `/boot/kernel/`. Note that there will be little or no performance loss from running a debug kernel, and it is useful to keep one around in case of a system panic.

However, when running low on disk space, there are different options to reduce the size of `/boot/kernel/`.

To not install the symbol files, make sure the following line exists in `/etc/src.conf`:

```
WITHOUT_KERNEL_SYMBOLS=yes
```

For more information see src.conf(5).

If you do not want to build a debug kernel, make sure that both of the following are true:

- This line does not exist in the kernel configuration file:

  ```
  makeoptions DEBUG=-g
  ```

- Do not run config(8) with `-g`.

Either of the above settings will cause the kernel to be built in debug mode.

To build and install only the specified modules, list them in `/etc/make.conf` :

```
MODULES_OVERRIDE= accf_http ipfw
```

Replace *accf_httpd ipfw* with a list of needed modules. Only the listed modules will be built. This reduces the size of the kernel directory and decreases the amount of time needed to build the kernel. For more information, read `/usr/share/examples/etc/make.conf` .

Unneeded devices can be removed from the kernel to further reduce the size. See Q: for more information.

To put any of these options into effect, follow the instructions to build and install the new kernel.

Most kernels (`/boot/kernel/kernel`) tend to be around 12 MB to 16 MB.

Q: Why does every kernel I try to build fail to compile, even `GENERIC`?

A: There are a number of possible causes for this problem:

- The source tree is different from the one used to build the currently running system. When attempting an upgrade, read `/usr/src/UPDATING` , paying particular attention to the "COMMON ITEMS" section at the end.
- The `make buildkernel` command did not complete successfuly. The `make buildkernel` target relies on files generated by the `make buildworld` target to complete its job correctly.
- Even when building FreeBSD-STABLE, it is possible that the source tree was fetched at a time when it was either being modified or it was broken. Only releases are guaranteed to be buildable, although FreeBSD-STABLE builds fine the majority of the time. Try re-fetching the source tree and see if the problem goes away. Try using a different mirror in case the previous one is having problems.

Q: Which scheduler is in use on a running system?

A: 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: ULE
```

Q: What is `kern.sched.quantum` ?

A: `kern.sched.quantum` is the maximum number of ticks a process can run without being preempted in the 4BSD scheduler.

Chapter 8. Disks, File Systems, and Boot Loaders

Q: How can I add my new hard disk to my FreeBSD system?

A: See the Adding Disks section in the FreeBSD Handbook.

Q: How do I move my system over to my huge new disk?

A: The best way is to reinstall the operating system on the new disk, then move the user data over. This is highly recommended when tracking *-STABLE* for more than one release or when updating a release instead of installing a new one. Install booteasy on both disks with boot0cfg(8) and dual boot until you are happy with the new configuration. Skip the next paragraph to find out how to move the data after doing this.

Alternatively, partition and label the new disk with either sade(8) or gpart(8). If the disks are MBR-formatted, booteasy can be installed on both disks with boot0cfg(8) so that the computer can dual boot to the old or new system after the copying is done.

Once the new disk set up, the data cannot just be copied. Instead, use tools that understand device files and syste flags, such as dump(8). Although it is recommended to move the data while in single-user mode, it is not required.

When the disks are formatted with UFS, never use anything but dump(8) and restore(8) to move the root file system. These commands should also be used when moving a single partition to another empty partition. The sequence of steps to use `dump` to move the data from one UFS partitions 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, to move `/dev/ada1s1a` with `/mnt` as the temporary mount point, type:

```
# newfs /dev/ada1s1a
# mount /dev/ada1s1a /mnt
# cd /mnt
# dump 0af - / | restore rf -
```

Rearranging 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/ada1s1a
# mount /dev/ada1s1a /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/ada1s1a
# newfs /dev/ada1s1d
# mount /dev/ada1s1a /mnt
# mkdir /mnt/var
# mount /dev/ada1s1d /mnt/var
# cd /mnt
# dump 0af - / | restore rf -
```

The cpio(1) and pax(1) utilities are also available for moving user data. These are known to lose file flag information, so use them with caution.

Q: Which partitions can safely use Soft Updates? I have heard that Soft Updates on `/` can cause problems. What about Journaled Soft Updates?

A: Short answer: Soft Updates can usually be safely used on all partitions.

Long answer: Soft Updates has two characteristics that may be undesirable on certain partitions. First, a Soft Updates partition has a small chance of losing data during a system crash. The partition will not be corrupted as the data will simply be lost. Second, Soft Updates can cause temporary space shortages.

When using Soft Updates, the kernel can take up to thirty seconds to write changes to the physical disk. When a large file is deleted the file still resides on disk until the kernel actually performs the deletion. This can cause a very simple race condition. Suppose one large file is deleted and another large file is immediately created. 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. This will produce an error that the partition does not have enough space, even though a large chunk of space has just been released. A few seconds later, the file creation works as expected.

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. This risk is extremely small, but generally manageable.

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. 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 be aware that it exists. If the system cannot tolerate this much risk, do not use Soft Updates on the root file system!

`/` is traditionally one of the smallest partitions. If `/tmp` is on `/`, there may be intermittent space problems. Symlinking `/tmp` to `/var/tmp` will solve this problem.

Finally, dump(8) does not work in live mode (-L) on a filesystem, with Journaled Soft Updates (SU+J).

Q: Can I mount other foreign file systems under FreeBSD?

A: 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 ext2fs(5) for more information.

NTFS
: FUSE based NTFS support is available as a port (sysutils/fusefs-ntfs). For more information see ntfs-3g.

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

ZFS
: 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 includes the Network File System NFS and the FreeBSD Ports Collection provides several FUSE applications to support many other file systems.

Q: How do I mount a secondary DOS partition?

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

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

Q: Is there a cryptographic file system for FreeBSD?

A: Yes, gbde(8) and geli(8). See the Encrypting Disk Partitions section of the FreeBSD Handbook.

Q: How do I boot FreeBSD and Linux® using GRUB?

A: To boot FreeBSD using GRUB, add the following to either `/boot/grub/menu.lst` or `/boot/grub/grub.conf` , depending upon which is used by the Linux® distribution.

```
title FreeBSD 9.1
 root (hd0,a)
 kernel /boot/loader
```

Where *hd0,a* points to the root partition on the first disk. To specify the slice number, use something like this *(hd0,2,a)* . By default, if the slice number is omitted, GRUB searches the first slice which has the a partition.

Q: How do I boot FreeBSD and Linux® using BootEasy?

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

This is recommended when running Windows® and Linux® as it makes it simpler to get Linux® booting again if Windows® is reinstalled.

Q: How do I change the boot prompt from `???` to something more meaningful?

A: This cannot be accomplished with the standard boot manager without rewriting it. There are a number of other boot managers in the `sysutils` category of the Ports Collection.

Q: How do I use a new removable drive?

A: If the drive already has a file system on it, use a command like this:

```
# mount -t msdosfs /dev/da0s1 /mnt
```

If the drive will only be used with FreeBSD systems, partition it with UFS or ZFS. This will provide long filename support, improvement in performance, and stability. If the drive will be used by other operating systems, a more portable choice, such as msdosfs, is better.

```
# dd if=/dev/zero of=/dev/da0 count=2
# gpart create -s GPT /dev/da0
```

```
# gpart add -t freebsd-ufs /dev/da0
```

Finally, create a new file system:

```
# newfs /dev/da0p1
```

and mount it:

```
# mount /dev/da0s1 /mnt
```

It is a good idea to add a line to `/etc/fstab` (see fstab(5)) so you can just type `mount /mnt` in the future:

```
/dev/da0p1 /mnt ufs rw,noauto 0 0
```

Q: Why do I get Incorrect super block when mounting a CD?

A: The type of device to mount must be specified. This is described in the Handbook section on Using Data CDs.

Q: Why do I get Device not configured when mounting a CD?

A: This generally means that there is no CD in the drive, or the drive is not visible on the bus. Refer to the Using Data CDs section of the Handbook for a detailed discussion of this issue.

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

A: The CD probably uses the "Joliet" extension for storing information about files and directories. This is discussed in the Handbook section on Using Data CD-ROMs.

Q: A CD burned under FreeBSD cannot be read under any other operating system. Why?

A: This means a raw file was burned to the CD, rather than creating an ISO 9660 file system. Take a look at the Handbook section on Using Data CDs.

Q: How can I create an image of a data CD?

A: This is discussed in the Handbook section on Writing Data to an ISO File System. For more on working with CD-ROMs, see the Creating CDs Section in the Storage chapter in the Handbook.

Q: Why can I not `mount` an audio CD?

A: Trying to mount an audio CD will produce 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. Instead, use a program that reads audio CDs, such as the audio/xmcd package or port.

Q: How do I `mount` a multi-session CD?

A: By default, mount(8) will attempt to mount the last data track (session) of a CD. To load an earlier session, use the `-s` command line argument. Refer to mount_cd9660(8) for specific examples.

Q: How do I let ordinary users mount CD-ROMs, DVDs, USB drives, and other removable media?

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

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

To make this persist across reboots, add the line `vfs.usermount=1` to `/etc/sysctl.conf` so that it is reset at system boot time.

Users can only mount devices they have read permissions to. To allow users to mount a device permissions must be set in `/etc/devfs.conf` .

For example, to allow users to mount the first USB drive add:

```
# Allow all users to mount a USB drive.
     own       /dev/da0       root:operator
     perm      /dev/da0       0666
```

All users can now mount devices they could read onto a directory that they own:

```
% mkdir ~/my-mount-point
% mount -t msdosfs /dev/da0 ~/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 the configuration.

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

A: This is due to how these commands actually work. `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 the file is deleted, 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. As an example, consider a file that is large enough that its presence affects the output of `du` and `df`. If this file is deleted while using `more` on it, `more` does not immediately choke and complain that it cannot view the file. The entry is removed from the directory so no other program or user can access it. However, `du` shows that it is gone as 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 the `more` session ends, `du` and `df` will agree.

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).

Note that Soft Updates can delay the freeing of disk space and it can take up to 30 seconds for the change to be visible.

Q: How can I add more swap space?

A: This sectionof the Handbook describes how to do this.

Q: Why does FreeBSD see my disk as smaller than the manufacturer says it is?

A: 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.

Q: How is it possible for a partition to be more than 100% full?

A: 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%. 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 `-m` in tunefs(8).

Q: Why does FreeBSD pause for a long time at boot when the system has large amounts of ram?

A: FreeBSD does a short memory test early in the boot process. This test usually only takes several seconds, however if the system has many 10s or 100s of gigabytes of memory it can take up to a few minutes. This test can be disabled by setting `hw.memtest.tests` to `0` in `/boot/loader.conf` .

For more details, see loader.conf(5).

Chapter 9. ZFS

Q: What is the minimum amount of RAM one should have to run ZFS?

A: A minimum of 4GB of RAM is required for comfortable usage, but individual workloads can vary widely.

Q: What is the ZIL and when does it get used?

A: The ZIL ((ZFS intent log) is a write log used to implement posix write commitment semantics across crashes. Normally writes are bundled up into transaction groups and written to disk when filled ("Transaction Group Commit"). However syscalls like fsync(2) require a commitment that the data is written to stable storage before returning. The ZIL is needed for writes that have been acknowledged as written but which are not yet on disk as part of a transaction. The transaction groups are timestamped. In the event of a crash the last valid timestamp is found and missing data is merged in from the ZIL.

Q: Do I need a SSD for ZIL?

A: By default, ZFS stores the ZIL in the pool with all the data. If an application has a heavy write load, storing the ZIL in a separate device that has very fast synchronous, sequential write performance can improve overall system. For other workloads, a SSD is unlikely to make much of an improvement.

Q: What is the L2ARC?

A: The L2ARC is a read cache stored on a fast device such as an SSD. This cache is not persistent across reboots. Note that RAM is used as the first layer of cache and the L2ARC is only needed if there is insufficient RAM.

L2ARC needs space in the ARC to index it. So, perversely, a working set that fits perfectly in the ARC will not fit perfectly any more if a L2ARC is used because part of the ARC is holding the L2ARC index, pushing part of the working set into the L2ARC which is slower than RAM.

Q: Is enabling deduplication advisable?

A: Generally speaking, no.

Deduplication takes up a significant amount of RAM and may slow down read and write disk access times. Unless one is storing data that is very heavily duplicated, such as virtual machine images or user backups, it is possible that deduplication will do more harm than good. Another consideration is the inability to revert deduplication status. If data is written when deduplication is enabled, disabling dedup will not cause those blocks which were deduplicated to be replicated until they are next modified.

Deduplication can also lead to some unexpected situations. In particular, deleting files may become much slower.

Q: I cannot delete or create files on my ZFS pool. How can I fix this?

A: This could happen because the pool is 100% full. ZFS requires space on the disk to write transaction metadata. To restore the pool to a usable state, truncate the file to delete:

```
% truncate -s 0 unimportant-file
```

File truncation works because a new transaction is not started, new spare blocks are created instead.

Note

On systems with additional ZFS dataset tuning, such as deduplication, the space may not be immediately available

Q: Does ZFS support TRIM for Solid State Drives?

A: ZFS TRIM support was added to FreeBSD 10-CURRENT with revision r240868. ZFS TRIM support was added to all FreeBSD-STABLE branches in r252162 and r251419, respectively.

ZFS TRIM is enabled by default, and can be turned off by adding this line to `/etc/sysctl.conf` :

```
vfs.zfs.trim_disable=1
```

Note

ZFS TRIM may not work with all configurations, such as a ZFS filesystem on a GELI-backed device.

Chapter 10. System Administration

Q: Where are the system start-up configuration files?

A: The primary configuration file is `/etc/defaults/rc.conf` which is described in rc.conf(5). System startup scripts such as `/etc/rc` and `/etc/rc.d`, which are described in rc(8), include this file. *Do not edit this file!* Instead, to edit an entry in `/etc/defaults/rc.conf`, copy the line into `/etc/rc.conf` and change it there.

For example, if to start named(8), the included DNS server:

```
# 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`.

Q: How do I add a user easily?

A: 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).

Q: Why do I keep getting messages like root: not found after editing `/etc/crontab`?

A: This is normally caused by editing the system crontab. This is not the correct way to do things as the system crontab has a different format to the per-user crontabs. The system crontab has an extra field, specifying which user to run the command as. cron(8) assumes this user is the first word of the command to execute. Since no such command exists, this error message is displayed.

To delete the extra, incorrect crontab:

```
# crontab -r
```

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

A: This is a security feature. In order to `su` to `root`, or any other account with superuser privileges, the user account must be a member of 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.

To allow someone to `su` to `root`, put them in the `wheel` group using `pw`:

```
# pw groupmod wheel -m lisa
```

The above example will add user `lisa` to the group `wheel`.

Q: 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?

A: Restart the system using **`boot -s`** at the loader prompt to enter single-user mode. When prompted for a shell pathname, press Enter and run `mount -urw /` to remount 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 that editor is on a network file system, either configure the network manually before mounting the network file systems, or use an editor which resides on a local file system, such as ed(1).

In order to use a full screen editor such as vi(1) or emacs(1), run `export TERM=xterm` on FreeBSD 9.0+, or `export TERM=cons25` on FreeBSD 8.X so that these editors can load the correct data from the termcap(5) database.

After performing these steps, edit `/etc/rc.conf` to fix the syntax error. The error message displayed immediately after the kernel boot messages should indicate the number of the line in the file which is at fault.

Q: Why am I having trouble setting up my printer?

A: See the Handbook entry on printing for troubleshooting tips.

Q: How can I correct the keyboard mappings for my system?

A: Refer to the Handbook section on using localization, specifically the section on console setup.

Q: Why can I not get user quotas to work properly?

A: 1. It is possible that the kernel is not configured to use quotas. In this case, add the following line to the kernel configuration file and recompile the kernel:

```
options QUOTA
```

Refer to the Handbook entry on quotas 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:

File System	Quota file
`/usr`	`/usr/admin/quotas`
`/home`	`/home/admin/quotas`
…	…

Q: Does FreeBSD support System V IPC primitives?

A: Yes, FreeBSD supports System V-style IPC, including shared memory, messages and semaphores, in the `GENERIC` kernel. With a custom kernel, support may be loaded with the `sysvshm.ko` , `sysvsem.ko` and `sysvmsg.ko` kernel modules, or enabled in the custom kernel by adding the following lines to the kernel configuration file:

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

Recompile and install the kernel.

Q: What other mail-server software can I use instead of Sendmail?

A: The Sendmail server is the default mail-server software for FreeBSD, but it can be replaced with another MTA installed from the Ports Collection. Available ports include mail/exim, mail/postfix, and mail/qmail. Search the mailing lists for discussions regarding the advantages and disadvantages of the available MTAs.

Q: I have forgotten the `root` password! What do I do?

A: 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 which will display a `#` prompt. Enter `mount -urw /` to remount the 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 a FreeBSD installation disk, choose the Live CD or Shell at the beginning of the install process and issue the commands mentioned above. Mount the specific partition in this case and then chroot to it. For example, replace `mount -urw /` with `mount /dev/ada0p1 /mnt; chroot /mnt` for a system on *`ada0p1`*.

Note

If the root partition cannot be mounted from single-user mode, it is possible that the partitions are encrypted and it

is impossible to mount them without the access keys. For more information see the section about encrypted disks in the FreeBSD Handbook.

Q: How do I keep Control+Alt+Delete from rebooting the system?

A: When using syscons(4), the default console driver, build and install a new kernel with this 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 the kernel is compiled with `SC_DISABLE_REBOOT` .

Q: How do I reformat DOS text files to UNIX® ones?

A: 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, use tr(1):

```
% 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.

Q: How do I kill processes by name?

A: Use pkill(1).

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

A: Go into single-user mode and then back to multi-user mode:

```
# shutdown now
# return
# exit
```

Q: I tried to update my system to the latest *-STABLE*, but got *-BETAx*, *-RC* or *-PRERELEASE*! What is going on?

A: 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 9.0-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 Subversion branches, refer to the Release Engineering article.

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

A: Short answer: the security level is 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. To check the current security level:

```
# sysctl kern.securelevel
```

The security level cannot be lowered in multi-user mode, so boot to single-user 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`.

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

A: Short answer: the system is at a 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. To check the security level:

```
# sysctl kern.securelevel
```

The security level cannot be lowered in multi-user mode. Either boot to single-user mode to change the date or change the security level in `/etc/rc.conf` and 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`.

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

A: 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.

Q: Why can I not unset the `schg` file flag?

A: The system is running a securelevel greater than `0`. Lower the securelevel and try again. For more information, see the FAQ entry on securelevel and the init(8) manual page.

Q: What is `vnlru`?

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

Q: What do the various memory states displayed by `top` mean?

A:

- `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.

Q: How much free memory is available?

A: 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.

Q: What is `/var/empty` ?

A: `/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. This directory should not be deleted.

Q: I just changed `/etc/newsyslog.conf` . How can I check if it does what I expect?

A: To see what newsyslog(8) will do, use the following:

```
% newsyslog -nrvv
```

Q: My time is wrong, how can I change the timezone?

A: Use tzsetup(8).

Chapter 11. The X Window System and Virtual Consoles

Q: What is the X Window System?

A: 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 administers the X protocol standards, with the current reference implementation, version 11 release 7.7, so references are often 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`.

Q: I want to run Xorg, how do I go about it?

A: To install Xorg do one of the following:

Use the x11/xorg meta-port, which builds and installs every Xorg component.

Use x11/xorg-minimal, which builds and installs only the necessary Xorg components.

Install Xorg from FreeBSD packages:

```
# pkg_add -r xorg
```

or on systems using pkg:

```
# pkg install xorg
```

After the installation of Xorg, follow the instructions from the X11 Configuration section of the FreeBSD Handbook.

Q: I *tried* to run X, but I get a No devices detected. error when I type `startx`. What do I do now?

A: The 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.

There are two solutions to the problem: set the `securelevel` back down to zero or run xdm(1) (or an alternative display manager) at boot time before the `securelevel` is raised.

See Q: for more information about running xdm(1) at boot time.

Q: Why does my mouse not work with X?

A: When using syscons(4), the default console driver, FreeBSD can be configured to support a mouse pointer on each virtual screen. 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). To use the mouse on one or more virtual consoles, *and* use X, see Q: and set up moused(8).

Then edit `/etc/X11/xorg.conf` and make sure the following lines exist:

```
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`):

```
# service devfs restart
```

Q: My mouse has a fancy wheel. Can I use it in X?

A: Yes, if X is configured for a 5 button mouse. To do this, add the lines `Buttons 5` and `ZAxisMapping 4 5` to the "InputDevice" section of `/etc/X11/xorg.conf`, as seen in this example:

```
Section "InputDevice"
   Identifier      "Mouse1"
   Driver          "mouse"
   Option          "Protocol" "auto"
   Option          "Device" "/dev/sysmouse"
   Option          "Buttons" "5"
   Option          "ZAxisMapping" "4 5"
EndSection
```

To use the mouse in Emacs, also add the following lines to`~/.emacs`:

```
;; wheel mouse
(global-set-key [mouse-4] 'scroll-down)
(global-set-key [mouse-5] 'scroll-up)
```

Q: My laptop has a Synaptics touchpad. Can I use it in X?

A: Yes, after configuring a few things to make it work.

In order to use the Xorg synaptics driver, first remove `moused_enable` from `rc.conf`.

To enable synaptics, add the following line to `/boot/loader.conf` :

```
hw.psm.synaptics_support="1"
```

Add the following to `/etc/X11/xorg.conf` :

```
Section "InputDevice"
Identifier  "Touchpad0"
Driver      "synaptics"
Option      "Protocol" "psm"
Option      "Device" "/dev/psm0"
EndSection
```

And be sure to add the following into the "ServerLayout" section:

```
InputDevice    "Touchpad0" "SendCoreEvents"
```

Q: How do I use remote X displays?

A: For security reasons, the default setting is to not allow a machine to remotely open a window.

To enable this feature, start X with the optional `-listen_tcp` argument:

```
% startx -listen_tcp
```

Q: What is a virtual console and how do I make more?

A: Virtual consoles provide 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. Type in your login name and password to start working on the first virtual console.

To start another session, perhaps to look at documentation for a program or to read mail while waiting for an FTP transfer to finish, hold down Alt and press F2. This will display the login prompt for the second virtual console. To go back to the original session, press 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 virtual consoles, 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"         xterm  on secure
ttyv9   "/usr/libexec/getty Pc"         xterm  on secure
ttyva   "/usr/libexec/getty Pc"         xterm  on secure
ttyvb   "/usr/libexec/getty Pc"         xterm  on secure
```

The more virtual terminals, the more resources that are used. This can be problematic on systems with 8 MB RAM or less. Consider changing `secure` to `insecure`.

Note

Versions of FreeBSD prior to 9.0 used the " cons25" terminal type, and not " xterm". Use the format of existing entries in when adding entries to `/etc/ttys` .

Important

In order to run an X server, at least one virtual terminal must be left to `off` for it to use. This means that only eleven of the Alt-function keys can be used as virtual consoles so that one is left for the X server.

For example, to run X and eleven virtual consoles, the setting for virtual terminal 12 should be:

```
ttyvb   "/usr/libexec/getty Pc"         xterm  off secure
```

The easiest way to activate the virtual consoles is to reboot.

Q: How do I access the virtual consoles from X?

A: Use Ctrl+Alt+F***n*** to switch back to a virtual console. Press Ctrl+Alt+F1 to return to the first virtual console.

Once at a text console, use Alt+F***n*** to move between them.

To return to the X session, switch to the virtual console running X. If X was started from the command line using `startx`, the X session will attach to the next unused

virtual console, not the text console from which it was invoked. For eight active virtual terminals, X will run on the ninth, so use Alt+F9.

Q: How do I start XDM on boot?

A: 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 runs `xdm` 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. `xdm` 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`.

When starting `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 `/usr/local/lib/X11/xdm/Xservers`:

```
: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.

Q: Why do I get Couldn't open console when I run `xconsole`?

A: When X is started 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.

Q: Why does my PS/2 mouse misbehave under X?

A: The mouse and the mouse driver may have become out of synchronization. In rare cases, the driver may also erroneously report synchronization errors:

```
psmintr: out of sync (xxxx != yyyy)
```

If this happens, disable the synchronization check code by setting the driver flags for the PS/2 mouse driver to `0x100`. This can be easiest achieved by adding `hint.psm.0.flags="0x100"` to `/boot/loader.conf` and rebooting.

Q: How do I reverse the mouse buttons?

A: Type `xmodmap -e "pointer = 3 2 1"`. Add this command to `~/.xinitrc` or `~/.xsession` to make it happen automatically.

Q: How do I install a splash screen and where do I find them?

A: The detailed answer for this question can be found in the Boot Time Splash Screens section of the FreeBSD Handbook.

Q: Can I use the Windows keys on my keyboard in X?

A: Yes. Use xmodmap(1) to define which functions the keys should perform.

Assuming all Windows keyboards are standard, 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 AltGr
- 117 — Menu, to the left of the right-hand Ctrl

To have the left Windows key print a comma, try this.

```
# xmodmap -e "keycode 115 = comma"
```

To have the Windows key-mappings enabled automatically every time X is started, either put the `xmodmap` commands in `~/.xinitrc` or, preferably, create a `~/.xmodmaprc` and include the `xmodmap` options, one per line, then add the following line to `~/.xinitrc`:

```
xmodmap $HOME/.xmodmaprc
```

For example, to 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 the window manager.

To do this, put the following in `~/.xmodmaprc`.

```
keycode 115 = F13
keycode 116 = F14
keycode 117 = F15
```

For the x11-wm/fvwm2 desktop manager, one 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 menu even if the cursor is not on the desktop, which is useful when no part of the desktop is visible.

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
```

Q: How can I get 3D hardware acceleration for OpenGL®?

A: The availability of 3D acceleration depends on the version of Xorg and the type of video chip. For an nVidia chip, 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.

nVidia provides detailed information on which card is supported by which driver on their web site: `http://www.nvidia.com/object/IO_32667.html` .

For Matrox G200/G400, check the x11-servers/mga_hal port.

For ATI Rage 128 and Radeon see ati(4), r128(4) and radeon(4).

Chapter 12. Networking

Q: Where can I get information on “diskless booting”?

A: “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, see the Handbook entry on diskless booting.

Q: Can a FreeBSD box be used as a dedicated network router?

A: Yes. Refer to the Handbook entry on advanced networking, specifically the section on routing and gateways.

Q: Can I connect my Windows® box to the Internet via FreeBSD?

A: 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.

Dialup users must use `-nat` and set `gateway_enable` to *YES* in `/etc/rc.conf` . For more information, refer to ppp(8) or the Handbook entry on user PPP.

If the connection to the Internet is over Ethernet, use natd(8). A tutorial can be found in the natd section of the Handbook.

Q: Does FreeBSD support PPP?

A: Yes. ppp(8) provides support for both incoming and outgoing connections.

For more information on how to use this, refer to the Handbook chapter on PPP.

Q: Does FreeBSD support NAT or Masquerading?

A: Yes. For instructions on how to use NAT over a PPP connection, see the Handbook entry on PPP. To use NAT over some other sort of network connection, look at the natd section of the Handbook.

Q: How can I set up Ethernet aliases?

A: If the alias is on the same subnet as an address already configured on the interface, add `netmask 0xffffffff` to this command:

```
# ifconfig ed0 alias 192.0.2.2 netmask 0xffffffff
```

Otherwise, specify the network address and netmask as usual:

```
# ifconfig ed0 alias 172.16.141.5 netmask 0xffffff00
```

More information can be found in the FreeBSD Handbook.

Q: Why can I not NFS-mount from a Linux® box?

A: 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
```

Q: Why does `mountd` keep telling me it can't change attributes and that I have a bad exports list on my FreeBSD NFS server?

A: The most frequent problem is not understanding the correct format of `/etc/exports`. Review exports(5) and the NFS entry in the Handbook, especially the section on configuring NFS.

Q: How do I enable IP multicast support?

A: Install the net/mrouted package or port and add `mrouted_enable="YES"` to `/etc/rc.conf` start this service at boot time.

Q: Why do I have to use the FQDN for hosts on my site?

A: See the answer in the FreeBSD Handbook.

Q: Why do I get an error, Permission denied, for all networking operations?

A: If the kernel is compiled with the `IPFIREWALL` option, be aware that the default policy is to deny all packets that are not explicitly allowed.

If the firewall is unintentionally misconfigured, restore network operability by typing the following as `root`:

```
# ipfw add 65534 allow all from any to any
```

Consider setting `firewall_type="open"` in `/etc/rc.conf` .

For further information on configuring this firewall, see the Handbook chapter.

Q: Why is my `ipfw` “fwd” rule to redirect a service to another machine not working?

A: Possibly because network address translation (NAT) is needed instead of just forwarding packets. A “fwd” rule only forwards packets, it does not actually change the data inside the packet. Consider this rule:

```
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 for a correct way to do this.

Q: How can I redirect service requests from one machine to another?

A: FTP and other service requests can be redirected with the sysutils/socket package or port. Replace the entry for the service in `/etc/inetd.conf` to call `socket`, as seen in this example for ftpd:

```
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.

Q: Where can I get a bandwidth management tool?

A: There are three bandwidth management tools available for FreeBSD. dummynet(4) is integrated into FreeBSD as part of ipfw(4). ALTQ has been integrated into FreeBSD as part of pf(4). Bandwidth Manager from Emerging Technologies is a commercial product.

Q: Why do I get /dev/bpf0: device not configured?

A: The running application requires the Berkeley Packet Filter (bpf(4)), but it was removed from a custom kernel. Add this to the kernel config file and build a new kernel:

```
device bpf        # Berkeley Packet Filter
```

Q: How do I mount a disk from a Windows® machine that is on my network, like smbmount in Linux®?

A: 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.

Q: What are these messages about: Limiting icmp/open port/closed port response in my log files?

A: This kernel message indicates that some activity is provoking it to send a large amount of ICMP or TCP reset (RST) responses. 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 indicates how many packets the kernel would have sent if the limit was not in place, and the second indicates the limit. This limit is controlled using `net.inet.icmp.icmplim` . This example sets the limit to `300` packets per second:

```
# sysctl net.inet.icmp.icmplim=300
```

To disable these messages without disabling response limiting, use `net.inet.icmp.icmplim_output` to disable the output:

```
# sysctl net.inet.icmp.icmplim_output=0
```

Finally, to disable response limiting completely, set `net.inet.icmp.icmplim` to `0`. Disabling response limiting is discouraged for the reasons listed above.

Q: What are these arp: unknown hardware address format error messages?

A: This means that some device on the 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. This is most commonly seen on cable modem networks. It is harmless, and should not affect the performance of the FreeBSD system.

Q: 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?

A: Because a packet is coming from outside the network unexpectedly. To disable them, set `net.link.ether.inet.log_arp_wrong_iface` to `0`.

Chapter 13. Security

Q: What is a sandbox?

A: "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 be able to run inside the walls. Since nothing the process does in regards to executing code is supposed to be able to breech the walls, a detailed audit of its code is not needed in order 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. 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 chrooted so that `/` 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.

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.

Q: What is securelevel?

A: `securelevel` is a security mechanism implemented in the kernel. When the securelevel is positive, the kernel restricts certain tasks; not even the superuser (`root`) is allowed to do them. The securelevel mechanism limits 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:

```
# sysctl -n kern.securelevel
```

The output contains the current value of the securelevel. If it is greater than `0`, at least some of the securelevel's protections are enabled.

The securelevel of a running system cannot be lowered as this would defeat its purpose. If a task requires that the securelevel be non-positive, change the `kern_securelevel` and `kern_securelevel_enable` variables in `/etc/rc.conf` and reboot.

For more information on securelevel and the specific things all the levels do, consult init(8).

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. Search the archives here for an extensive discussion. A more fine-grained mechanism is preferred.

Q: BIND (`named`) is listening on some high-numbered ports. What is going on?

A: 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. To get past that firewall, 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 sockstat(1) output and notice odd things!

Q: The Sendmail daemon is listening on port 587 as well as the standard port 25! What is going on?

A: 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.

Q: What is this UID 0 `toor` account? Have I been compromised?

A: Do not worry. `toor` is an "alternative" superuser account, where toor is root spelled backwards. It is intended to be used with a non-standard shell so the default shell for `root` does not need to change. This is important as shells which are not part of the base distribution, but are instead installed from ports or packages, are 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 the file system containing `/usr/local/bin`) is not mounted, `root` will not be able to log in to fix a problem and will have to reboot into single-user mode in order to enter 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, a user cannot log in using `toor` as it does not have a password, so log in as `root` and set a password for `toor` before using it to login.

Chapter 14. PPP

Q: I cannot make ppp(8) work. What am I doing wrong?

A: First, read ppp(8) and the PPP section of the Handbook. To assist in troubleshooting, 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 at the start of the `default` section in `/etc/ppp/ppp.conf` . Make sure that `/etc/syslog.conf` contains the lines below and the file `/var/log/ppp.log` exists:

```
!ppp
*.*        /var/log/ppp.log
```

A lot about what is going can be learned from the log file. Do not worry if it does not all make sense as it may make sense to someone else.

Q: Why does ppp(8) hang when I run it?

A: This is usually because the hostname will not resolve. The best way to fix this is to make sure that `/etc/hosts` is read first by the by ensuring that the `hosts` line is listed first in `/etc/host.conf` . Then, put an entry in `/etc/hosts` for the local machine. If there is no local network, change the `localhost` line:

```
127.0.0.1        foo.example.com foo localhost
```

Otherwise, add another entry for the host. Consult the relevant manual pages for more details.

When finished, verify that this command is successful: `ping -c1 `hostname`` .

Q: Why will ppp(8) not dial in `-auto` mode?

A: First, check that a default route exists. This command should display two entries:

```
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
```

If a default route is not listed, make sure that the `HISADDR` line has been added to `/etc/ppp/ppp.conf` .

Another reason for the default route line being missing is that a default route has been added to `/etc/rc.conf` and this line is missing from `/etc/ppp/ppp.conf` :

```
delete ALL
```

If this is the case, go back to the Final System Configuration section of the Handbook.

Q: What does No route to host mean?

A: This error is usually because the following section is missing in `/etc/ppp/ppp.linkup` :

```
MYADDR:
  delete ALL
  add 0 0 HISADDR
```

This is only necessary for a dynamic IP address or when the address of the default gateway is unknown. When using interactive mode, the following can be typed in 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 section of the Handbook for further details.

Q: Why does my connection drop after about 3 minutes?

A: 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 `ppp.conf` , 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.

Q: Why does my connection drop under heavy load?

A: If Link Quality Reporting (LQR) is configured, it is possible that too many LQR packets are lost between the FreeBSD system and the peer. ppp(8) deduces that the line must therefore be bad, and disconnects. LQR is disabled by default and can be enabled with the following line:

```
enable lqr
```

Q: Why does my connection drop after a random amount of time?

A: Sometimes, on a noisy phone line or even on a line with call waiting enabled, the modem may hang up because it incorrectly thinks that it lost carrier.

There is a setting on most modems for determining how tolerant it should be to temporary losses of carrier. Refer to the modem manual for details.

Q: Why does my connection hang after a random amount of time?

A: Many people experience hung connections with no apparent explanation. The first thing to establish is which side of the link is hung.

When using an external modem, try using ping(8) to see if the TD light is flashing when data is transmitted. If it flashes but 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, use the `set server` command in `ppp.conf`. When the hang occurs, connect to ppp(8) using pppctl(8). If the network connection suddenly revives due to the activity on the diagnostic socket, or if it will not connect but the `set socket` command succeeded at startup time, the problem is local. If it can connect but things are still hung, enable local 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 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, there are now two possibilities:

- If the problem is remote, read on entry Q:.
- If the problem is local, read on entry Q:.

Q: The remote end is not responding. What can I do?

A: There is very little that can be done about this. Many ISPs will refuse to help users not running a Microsoft® OS. Add `enable lqr` to `/etc/ppp/ppp.conf`, allowing ppp(8) to detect the remote failure and hang up. This detection is relatively slow and therefore not that useful.

First, try disabling all local compression by adding the following to the 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 is good information for the ISP, although it may make it apparent that it is not a Microsoft® system.

Before contacting the 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 the 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 why their side is having a problem.

Q: ppp(8) has hung. What can I do?

A: In this case, 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, type:

```
# cd /usr/src/usr.sbin/ppp
# env DEBUG_FLAGS='-g' make clean
# env DEBUG_FLAGS='-g' make install
```

Then, 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, use the `bt` or `where` commands to get a stack trace. Save the output of the gdb session, and "detach" from the running process by typing `quit`.

Q: I keep seeing errors about magic being the same. What does it mean?

A: Occasionally, just after connecting, there may be 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, there will be 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 `ppp.conf`:

```
set openmode passive
```

This tells ppp(8) to wait for the server to initiate LCP negotiations. Some servers however may never initiate negotiations. In this case, try 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.

Q: LCP negotiations continue until the connection is closed. What is wrong?

A: 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. This command can also be used 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.

Q: Why does ppp(8) lock up when I shell out to test it?

A: When using `shell` or `!`, ppp(8) executes a shell or the passed arguments. The ppp program will wait for the command to complete before continuing. Any attempt to use the PPP link while running the command will appear as a frozen link. This is because ppp(8) is waiting for the command to complete.

To execute commands like this, use `!bg` instead. This will execute the given command in the background, and ppp(8) can continue to service the link.

Q: Why does ppp(8) over a null-modem cable never exit?

A: 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.

Q: Why does ppp(8) dial for no reason in `-auto` mode?

A: If ppp(8) is dialing unexpectedly, determine the cause, and set up dial filters 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, the reason will be logged with a convenient timestamp next to it.

Next, 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 demand-dial capabilities. Most programs will need a DNS lookup before doing any other network related things.

In the DNS case, try to determine what is actually trying to resolve a host name. A lot of the time, Sendmail is the culprit. Make sure to configure Sendmail not to do any DNS lookups in its configuration file. See the section on using email with a dialup connection in the FreeBSD Handbook for details. You may also want to add the following line to `.mc`:

```
define(`confDELIVERY_MODE', `d')dnl
```

This will make Sendmail queue everything until the queue is run, usually, every 30 minutes, or until a `sendmail -q` is done, perhaps from `/etc/ppp/ppp.linkup` .

Q: What do these CCP errors mean?

A: 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, but the peer does not want to negotiate any compression at all. The messages are harmless, but can be silenced by disabling the compression:

```
disable pred1
```

Q: Why does ppp(8) not log my connection speed?

A: To log all lines of the modem conversation, enable the following:

```
set log +connect
```

This will make ppp(8) log everything up until the last requested “expect” string.

To see the connect speed when using PAP or CHAP, make sure to configure ppp(8) to expect the whole CONNECT line, using something like this:

```
set dial "ABORT BUSY ABORT NO\\sCARRIER TIMEOUT 4 \
  \"\" ATZ OK-ATZ-OK ATDT\\T TIMEOUT 60 CONNECT \\c \\n"
```

This gets the CONNECT, sends nothing, then expects a line-feed, forcing ppp(8) to read the whole CONNECT response.

Q: Why does ppp(8) ignore the \ character in my chat script?

A: The ppp utility parses each line in its configuration 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. To specify a " character, escape it using a backslash (\).

When the chat interpreter parses each argument, it re-interprets the argument to find any special escape sequences such as `\P` or `\T`. As a result of this double-parsing, remember to use the correct number of escapes.

To actually send a \ character, do 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
```

Q: What are FCS errors?

A: FCS stands for Frame Check Sequence. 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 the link is bad or if the serial driver is dropping packets, it will produce the occasional FCS error. This is not usually worth worrying about although it does slow down the compression protocols substantially.

If the link freezes as soon as it connects and produces a large number of FCS errors, make sure the modem is not using software flow control (XON/XOFF). If the link must use software flow control, use `set accmap 0x000a0000` to tell ppp(8) to escape the `^Q` and `^S` characters.

Another reason for too many FCS errors may be that the remote end has stopped talking PPP. In this case, enable `async` logging to determine if the incoming data is actually a login or shell prompt. If it is a shell prompt at the remote end, it is

possible to terminate ppp(8) without dropping the line by using `close lcp` followed by `term`) to reconnect to the shell on the remote machine.

If nothing in the log file indicates why the link was terminated, ask the remote administrator or ISP why the session was terminated.

Q: None of this helps — I am desperate! What can I do?

A: If all else fails, send the details of the error, the configuration files, how ppp(8) is being started, the relevant parts of the log file, and the output of `netstat -rn`, before and after connecting, to the FreeBSD general questions mailing list.

Chapter 15. Serial Communications

This section answers common questions about serial communications with FreeBSD. PPP is covered in the Networking section.

Q: Which multi-port serial cards are supported by FreeBSD?

A: There is a list of these in the Serial Communications chapter of the Handbook.

Most multi-port PCI cards that are based on 16550 or clones are supported with no extra effort.

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

Check uart(4) and sio(4) to get more information on configuring such cards.

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

A: See this section of the Handbook.

Q: How do I tell if FreeBSD found my serial ports or modem cards?

A: As the FreeBSD kernel boots, it will probe for the serial ports for which the kernel is configured. Either watch the boot messages closely or run this command after the system is up and running:

```
% dmesg | grep -E "^sio[0-9]"
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 example shows two serial ports. The first is on IRQ4, port address `0x3f8`, and has a 16550A-type UART chip. The second uses the same kind of chip but is on IRQ3 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 the system, or if there are more modem cards or serial ports than the kernel is configured for, reconfigure using the instructions in building a kernel for more details.

Q: How do I access the serial ports on FreeBSD?

A: The third serial port, `sio2`, or `COM3`, 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?

When opening `/dev/ttydX` in blocking mode, a process will wait for the corresponding `cuadX` device to become inactive, and then wait for the carrier detect line to go active. When the `cuadX` device is opened, it makes sure the serial port is not already in use by the `ttydX` device. If the port is available, it steals it from the `ttydX` device. Also, the `cuadX` device does not care about carrier detect. With this scheme and an auto-answer modem, remote users can log in and local users can still dial out with the same modem and the system will take care of all the conflicts.

Q: How do I enable support for a multi-port serial card?

A: The section on kernel configuration provides information about configuring the kernel. For a multi-port 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`).

Q: Can I set the default serial parameters for a port?

A: See the Serial Communications section in the FreeBSD Handbook.

Q: How can I enable dialup logins on my modem?

A: Refer to the section about Dial-in Services in the FreeBSD Handbook.

Q: How can I connect a dumb terminal to my FreeBSD box?

A: This information is in the Terminals section of the FreeBSD Handbook.

Q: Why can I not run `tip` or `cu`?

A: The built-in tip(1) and cu(1) utilities can only access the `/var/spool/lock` directory via user `uucp` and group `dialer`. Use the `dialer` group to control who has access to the modem or remote systems by adding user accounts to `dialer`.

Alternatively, everyone can be configured to run tip(1) and cu(1) by typing:

```
# chmod 4511 /usr/bin/cu
# chmod 4511 /usr/bin/tip
```

Chapter 16. Miscellaneous Questions

Q: FreeBSD uses a lot of swap space even when the computer has free memory left. Why?

A: FreeBSD will proactively move entirely idle, unused pages of main memory into swap in order to make more main memory available for active use. This heavy use of swap is balanced by using the extra free memory for cacheing.

Note that while FreeBSD is proactive in this regard, it does not arbitrarily decide to swap pages when the system is truly idle. Thus, the system will not be all paged out after leaving it idle overnight.

Q: Why does `top` show very little free memory even when I have very few programs running?

A: The simple answer is that free memory is wasted memory. Any memory that 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.

Q: Why will `chmod` not change the permissions on symlinks?

A: Symlinks do not have permissions, and by default, chmod(1) will follow symlinks to change the permissions on the source file, if possible. For the file, `foo` with a symlink named `bar`, 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, use either `-H` or `-L` together with `-R` to make this work. See chmod(1) and symlink(7) for more information.

Warning

`-R` does a *recursive* chmod(1). Be careful about specifying directories or symlinks to directories to chmod(1). 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`, to change the permissions of `foo` (actually `bar`), 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`.

Q: Can I run DOS binaries under FreeBSD?

A: Yes. A DOS emulation program, emulators/doscmd, is available in the FreeBSD Ports Collection.

If doscmd will not suffice, emulators/pcemu emulates an 8088 and enough BIOS services to run many DOS text-mode applications. It requires the X Window System.

The Ports Collection also has emulators/dosbox. The main focus of this application is emulating old DOS games using the local file system for files.

Q: What do I need to do to translate a FreeBSD document into my native language?

A: See the Translation FAQ in the FreeBSD Documentation Project Primer.

Q: Why does my email to any address at `FreeBSD.org` bounce?

A: The `FreeBSD.org` mail system implements some Postfix checks on incoming mail and rejects mail that is either from misconfigured relays or otherwise appears likely to be spam. Some of the specific requirements are:

- The IP address of the SMTP client must "reverse-resolve" to a forward confirmed hostname.
- The fully-qualified hostname given in the SMTP conversation (either HELO or EHLO) must resolve to the IP address of the client.

Other advice to help mail reach its destination include:

- Mail should be sent in plain text, and messages sent to mailing lists should generally be no more than 200KB in length.
- Avoid excessive cross posting. Choose *one* mailing list which seems most relevant and send it there.

If you still have trouble with email infrastructure at `FreeBSD.org` , send a note with the details to <`postmaster@freebsd.org` >; Include a date/time interval so that

logs may be reviewed — and note that we only keep one week's worth of mail logs. (Be sure to specify the time zone or offset from UTC.)

Q: Where can I find a free FreeBSD account?

A: 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, 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.

Q: What is the cute little red guy's name?

A: 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”.

More about the BSD daemon is available on his home page.

Q: Can I use the BSD daemon image?

A: Perhaps. The BSD daemon is copyrighted by Marshall Kirk McKusick. Check his Statement on the Use of the BSD Daemon Figure for detailed usage terms.

In summary, the image can be used in a tasteful manner, for personal use, so long as appropriate credit is given. Before using the logo commercially, contact Kirk McKusick <mckusick@FreeBSD.org > for permission. More details are available on the BSD Daemon's home page.

Q: Do you have any BSD daemon images I could use?

A: Xfig and eps drawings are available under `/usr/share/examples/BSD_daemon/ .`

Q: I have seen an acronym or other term on the mailing lists and I do not understand what it means. Where should I look?

A: Refer to the FreeBSD Glossary.

Chapter 17. Advanced Topics

Q: How can I learn more about FreeBSD's internals?

A: See the FreeBSD Architecture Handbook.

Additionally, much general UNIX® knowledge is directly applicable to FreeBSD.

Q: How can I contribute to FreeBSD?

A: See the article on Contributing to FreeBSD for specific advice on how to do this. Assistance is more than welcome!

Q: What are snapshots and releases?

A: There are currently 4 active/semi-active branches in the FreeBSD Subversion Repository. (Earlier branches are only changed very rarely, which is why there are only 4 active branches of development):

- stable/8/ AKA *8-STABLE*
- stable/9/ AKA *9-STABLE*
- stable/10/ AKA *10-STABLE*
- head/ AKA *-CURRENT* AKA *11-CURRENT*

`HEAD` is not an actual branch tag. It is a symbolic constant for the current, non-branched development stream known as *-CURRENT*.

Right now, *-CURRENT* is the 11.*X* development stream; the *10-STABLE* branch, stable/10/, forked off from *-CURRENT* in January 2014 and the *9-STABLE* branch, stable/9/, forked off from *-CURRENT* in September 2011.

Q: Can I follow *-CURRENT* with limited Internet access?

A: Yes, this can be done *without* downloading the whole source tree by using the CTM facility.

Q: I have written a kernel extension, who do I send it to?

A: Take a look at the article on Contributing to FreeBSD to learn how to submit code.

And thanks for the thought!

Q: How can I make the most of the data I see when my kernel panics?

A: 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
```

This message is not enough. While the instruction pointer value is important, it is also configuration dependent as it varies depending on the kernel image. If it is a `GENERIC` kernel image from one of the snapshots, it is possible for somebody else to track down the offending function, but for a custom kernel, only you can tell us where the fault occurred.

To proceed:

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:

   ```
   % nm -n kernel.that.caused.the.panic | grep f0xxxxx
   ```

   If that does not yield any results, chop off another digit. Repeat until there is 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 the kernel configuration file:

   ```
   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 `KERNCONF` is not included, the `GENERIC` kernel will instead 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 capture a crash dump, edit `/etc/rc.conf` and set `dumpdev` to point to either the swap partition or `AUTO`. This will cause the rc(8) scripts to use the dumpon(8) command to enable crash dumps. This command can also be run 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 physical RAM. Therefore, make sure there is enough space in `/var/crash` to hold the dump. Alternatively, run savecore(8) manually and have it recover the crash dump to another directory with 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, for 1 GB of RAM,

limit the kernel's memory usage to 128 MB, so that the crash dump size will be 128 MB instead of 1 GB.

Once the crash dump has been recovered , get a stack trace 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, 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. The stack trace is usually read from the bottom up to trace the exact sequence of events that lead to the crash. kgdb(1) can also be used to print out the contents of various variables or structures to examine the system state at the time of the crash.

Tip

If a second computer is available, kgdb(1) can be configured to do remote debugging, including setting breakpoints and single-stepping through the kernel code.

Note

If `DDB` is enabled and the kernel drops into the debugger, a panic and a crash dump can be forced 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.

Q: Why has `dlsym()` stopped working for ELF executables?

A: 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.

To search, using `dlsym()`, for symbols present in the main executable of a process, link the executable using the `--export-dynamic` option to the ELF linker (ld(1)).

Q: How can I increase or reduce the kernel address space on i386?

A: By default, the kernel address space is 1 GB (2 GB for PAE) for i386. When running a network-intensive server or using ZFS, this will probably not be enough.

Add the following line to the kernel configuration file to increase available space and rebuild the 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 18. 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 join them in making this FAQ even better.

Bibliography

[1] *FreeBSD Unleashed*. Michael Urban and Brian Tiemann. Sams. 1st edition. 992 pages. October 2001. ISBN 0-67232-206-4.

[2] *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.

[3] *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] *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.

[5] *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.

[6] *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.

[7] *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 MA . 1996. ISBN 0-201-54979-4.

[8] *The Design and Implementation of the FreeBSD Operating System*. M. K. McKusick and George V. Neville-Neil. Addison-Wesley. Boston MA . 2004. ISBN 0-201-70245-2.

[9] *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.

[10] *The Complete FreeBSD*. Greg Lehey. Walnut Creek. 3rd edition. June 1999. 773 pages. ISBN 1-57176-246-9.

[11] *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.

[12] *FreeBSD for PC 98'ers (in Japanese)*. SHUWA System Co, LTD.. ISBN 4-87966-468-5 C3055 P2900E.

[13] *FreeBSD (in Japanese)*. CUTT. ISBN 4-906391-22-2.

[14] *Complete Introduction to FreeBSD (in Japanese)*. Shoeisha Co., Ltd. ISBN 4-88135-473-6 P3600E.

[15] *Personal UNIX Starter Kit FreeBSD (in Japanese)*. ASCII. ISBN 4-7561-1733-3 P3000E.

[16] *FreeBSD Handbook (Japanese translation)*. ASCII. ISBN 4-7561-1580-2 P3800E.

[17] *FreeBSD mit Methode (in German)*. Computer und Literature Verlag/Vertrieb Hanser. 1998. ISBN 3-932311-31-0.

[18] *FreeBSD install and Utilization Manual (in Japanese)*. Mainichi Communications Inc..

[19] *Building Internet Server with FreeBSD (in Indonesia Language)*. Elex Media Komputindo. Onno W Purbo, Dodi Maryanto, Syahrial Hubbany, and Widjil Widodo.

[20] *The FreeBSD Corporate Networker's Guide*. Addison-Wesley.

[21] *UNIX in a Nutshell*. O'Reilly & Associates, Inc.. 1990. ISBN 093717520X.

[22] *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.

[23] *FreeBSD User's Reference Manual (Japanese translation)*. Mainichi Communications Inc.. Jpman Project, Japan FreeBSD Users Group. 1998. ISBN 4-8399-0088-4 P3800E.

[24] "Online Guide for newcomers to the UNIX environment". Edinburgh University.

[25] *DNS and BIND*. O'Reilly & Associates, Inc. ISBN 1-56592-512-2. Paul Albitz Albitz and Cricket Liu. 1998. 3rd edition.

[26] *Sendmail*. O'Reilly & Associates, Inc. 1997. 2nd edition. Brian Costales. ISBN 1-56592-222-0.

[27] *Essential System Administration*. Æleen Frisch. 2nd edition. O'Reilly & Associates. 1995. ISBN 1-56592-127-5.

[28] *TCP/IP Network Administration*. Craig Hunt. 2nd edition. O'Reilly & Associates, Inc. 1997. ISBN 1-56592-322-7.

[29] *Managing NFS and NIS*. Hal Stern. O'Reilly & Associates, Inc. 1991. ISBN 0-937175-75-7.

[30] *FreeBSD System Administration's Manual*. Jpman Project, Japan FreeBSD Users Group. Mainichi Communications Inc.. 1998. ISBN 4-8399-0109-0 P3300E.

[31] *X Window System Toolkit*. Digital Press. Paul Asente. ISBN 1-55558-051-3.

[32] *C: A Reference Manual*. Prentice Hall. 1995. 4th edition. Samuel P. Harbison and Guy L. Jr. Steele. ISBN 0-13-326224-3.

[33] *The C Programming Language*. Prentice Hall. 1998. Brian Kernighan and Dennis Ritchie. ISBN 0-13-110362-9.

[34] *Porting UNIX Software*. Greg Lehey. O'Reilly & Associates, Inc.. 1995. ISBN 1-56592-126-7.

[35] *The Standard C Library*. Prentice Hall. 1992. P. J. Plauger. ISBN 0-13-131509-9.

[36] *Advanced Programming in the UNIX Environment*. Addison-Wesley. 1992. W. Richard Stevens. ISBN 0-201-56317-7.

[37] *UNIX Network Programming*. W. Richard Stevens. Prentice Hall. 1998. 2nd edition. ISBN 0-13-490012-X.

[38] *Writing Serial Drivers for UNIX*. Bill Wells. December 1994. Dr. Dobb's Journal. pp68-71, pp97-99.

[39] *UNIX System Architecture*. Prentice-Hall, Inc. 1990. Prabhat K. Andleigh. ISBN 0-13-949843-5.

[40] *Porting UNIX to the 386*. William Jolitz. Dr. Dobb's Journal. January 1991-July 1992.

[41] *TCP/IP Illustrated, Volume 1: The Protocols*. W. Richard Stevens. Addison-Wesley. 1996. ISBN 0-201-63346-9.

[42] *Unix Systems for Modern Architectures*. Addison-Wesley. Curt Schimmel. 1994. ISBN 0-201-63338-8.

[43] *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.

[44] *UNIX Internals -- The New Frontiers*. Uresh Vahalia. Prentice Hall. 1996. ISBN 0-13-101908-2.

[45] *TCP/IP Illustrated, Volume 2: The Implementation*. Gary R. Wright and W. Richard Stevens. 1995. Addison-Wesley. ISBN 0-201-63354-X.

[46] *Firewalls and Internet Security: Repelling the Wily Hacker*. William R. CHeswick and Steven M. Bellovin. Addison-Wesley. 1995. ISBN 0-201-63357-4.

[47] *Practical UNIX Security*. Simson Garfinkel and Gene Spafford. 1996. 2nd edition. O'Reilly & Associates, Inc. ISBN 1-56592-148-8.

[48] *PGP Pretty Good Privacy*. Simson Garfinkel. O'Reilly & Associates, Inc. 1995. ISBN 1-56592-098-8.

[49] *Pentium Processor System Architecture*. Don Anderson and Tom Shanley. Addison-Wesley. 1995. 2nd edition. ISBN 0-201-40992-5.

[50] *Programmer's Guide to the EGA, VGA, and Super VGA Cards*. Richard F. Ferraro. 3rd edition. Addison-Wesley. 1995. ISBN 0-201-62490-7.

[51] *80486 System Architecture*. Tom Shanley. Addison-Wesley. 1995. 3rd edition. ISBN 0-201-40994-1.

[52] *ISA System Architecture*. Tom Shanley. Addison-Wesley. 3rd edition. 1995. ISBN 0-201-40996-8.

[53] *PCI System Architecture*. Tom Shanley. Addison-Wesley. 1995. 3rd edition. ISBN 0-201-40993-3.

[54] *The Undocumented PC*. Frank Van Gilluwe. Addison-Wesley. 1994. ISBN 0-201-62277-7.

[55] *Bell System Technical Journal, Unix Time-Sharing System*. American Telephone & Telegraph Company. July-August 1978. Vol 57, No 6, Part 2. ISSN0005-8580.

[56] *Lion's Commentary on UNIX*. John Lion. ITP Media Group. 1996. 6th edition. ISBN 1573980137.

[57] *The New Hacker's Dictionary*. Eric S. Raymond. MIT Press. 1996. 3rd edition. ISBN 0-262-68092-0.

[58] *A quarter century of UNIX*. Peter H. Salus. Addison-Wesley. 1994. ISBN 0-201-54777-5.

[59] *The UNIX-HATERS Handbook*. Steven Strassman, Daniel Weise, and Simon Garfinkel. IDG Books Worldwide, Inc. 1994. ISBN 1-56884-203-1.

[60] *Life with UNIX — special edition*. Don Libes and Sandy Ressler. Prentice-Hall. 1989. ISBN 0-13-536657-7.

[61] *The BSD Family Tree*. 1997.

[62] *Absolute BSD*. Michael Lucas. No Starch Press. June 2002. ISBN 1-886411-74-3.

[63] *The C/C++ Users Journal*. R&D Publications Inc.. ISSN 1075-2838.

[64] *Sys Admin — The Journal for UNIX System Administrators*. Miller Freeman, Inc. ISSN 1061-2688.