

Windows+BSD+Linux Installation Guide

Subhasish Ghosh

subhasish_ghosh@linuxwaves.com Version: v1.1 Last updated: August 21st, 2003

Revision History:

- ◆ v1.0: Initial release, reviewed by Linux Documentation Project (LDP). A few structural changes and minor corrections made; Chapters "Troubleshooting" and "Installing OpenBSD 3.2-RELEASE" added.
- ◆ v1.1: Corrections made and info added. Information on FreeBSD -RELEASE updated to cover 4.8-RELEASE. Chapter "Installing NetBSD 1.6.1" added and a few minor modifications made to the guide.

Welcome! This document explains how three different operating systems can be installed and configured on the same hard disk of a computer. The reader may choose from: Microsoft Windows 95/98(Second Edition)/Millennium Edition(ME)/NT/2K/XP + FreeBSD 4.8-RELEASE/OpenBSD 3.2-RELEASE/NetBSD 1.6.1 + Linux. Only Intel architecture multi-booting is discussed and GNU GRUB is the boot loader of choice. If you have any questions or comments, feel free to contact me by e-mail at this address: subhasish_ghosh@linuxwaves.com. Thank you!

Note: In the Organization section, readers must choose to execute Chapter 3 or Chapter 4 or Chapter 5.

1. Introduction

- ◆ 1.1 Purpose of this Guide
- ◆ 1.2 What is Multiple-booting?
- ◆ 1.3 Multi-booting Pros and Cons
- ◆ 1.4 List of Assumptions
- ◆ 1.5 Acknowledgements
- ◆ 1.6 Legalese
- ◆ 1.7 About the Author

2. FAQ

3. Organization

- ◆ 3.1 Chapter 1: About the operating systems
 - ◇ 3.1.1 Microsoft Windows
 - ◇ 3.1.2 Linux
 - ◇ 3.1.3 FreeBSD
 - ◇ 3.1.4 OpenBSD
 - ◇ 3.1.5 NetBSD
- ◆ 3.2 Chapter 2: Installing Microsoft Windows
 - ◇ Section A: Installing MS Windows 95

- ◇ [Section B: Installing MS Windows 98 \(SE\)](#)
- ◇ [Section C: Installing MS Windows Millennium \(ME\)](#)
- ◇ [Section D: Installing MS Windows NT \(4.0\)](#)
- ◇ [Section E: Installing MS Windows 2000 \(Professional\)](#)
- ◇ [Section F: Installing MS Windows XP \(Home/Professional\)](#)
- ◇ [Summary of the actual steps](#)
- ◆ 3.3 Chapter 3: [Installing FreeBSD 4.8-RELEASE](#)
 - ◇ [1. Slicing in FreeBSD](#)
 - ◇ [2. Filesystems in FreeBSD](#)
 - ◇ [3. Summary of Installation Steps](#)
- ◆ 3.4 Chapter 4: [Installing OpenBSD 3.2-RELEASE](#)
 - ◇ [Summary of Installation Steps](#)
- ◆ 3.5 Chapter 5: [Installing NetBSD 1.6.1](#)
 - ◇ [Summary of Installation Steps](#)
- ◆ 3.6 Chapter 6: [Installing Red Hat Linux 7.3](#)
 - ◇ [1. Disk drive and partition naming in Linux](#)
 - ◇ [2. Filesystems in Linux](#)
 - ◇ [3. Summary of Installation Steps](#)
- ◆ 3.7 Chapter 7: [Mounting Filesystems](#)
 - ◇ [1. Mounting filesystems in FreeBSD](#)
 - ◇ [2. Mounting filesystems in Linux](#)

4. Troubleshooting

5. Conclusion

[Home](#)

[Next](#)

[Previous](#)[Home](#)[Next](#)

1. Introduction

1.1 Purpose of this Guide

Though many of us prefer using only one operating system on our personal computers for our daily tasks, there may be times when we would like to install two or more operating systems on the same computer. Let us suppose, you are required to use MATLAB 6.0 for your engineering applications; design, program and implement a web site using PHP and the back-end database using MySQL (as a part of an enterprise-application development project) and also learn how to configure a packet-filtering Firewall using IPFW. As it so happens, the MATLAB software runs on Microsoft Windows platform, the MySQL database you want to implement on the Linux operating system and last but not the least, your professor at the university is teaching "how to configure a packet-filtering Firewall using IPFW" using the FreeBSD operating system software. In such situations, you can safely fall back on the "multiple-booting" mechanism and of course refer to this guide from time to time.

This document represented in the form of a *Guide*, is intended to help those Windows, Linux and FreeBSD/OpenBSD/NetBSD users who believe in the method of "learning by doing". Multiple-booting systems is not an exact science. You come across a document, you read it, find it interesting, do it yourself, make mistakes and then finally you achieve the desired aim. That is how you are supposed to learn. That is how I learnt. That is how the Unix experts learnt a long time back. By the time a reader finishes reading this guide, he would have a clear understanding of the basic topics required for successfully installing and configuring three different operating systems on the same hard disk of a computer. I have tried my level best to describe each and every topic in a clear and easily understandable simple language.

Most installation HOWTOs and guides (for Linux, FreeBSD and OpenBSD operating systems) which are available on the Internet are incomplete because they assume too much leaving the reader to do most of the difficult stuff themselves. In this guide, I illustrate every step thereby making the entire process a simple walk through.

1.2 What is Multiple-booting?

The Webster's New World Computer Dictionary (9th edition) by Bryan Pfaffenberger, defines the term "dual-booting" as: 'A computer that enables the user to choose between two operating systems at boot time'. Rightly so. Considering this definition as our boilerplate, we may frame our own definition of a "multiple-booting" or (in short) a multi-boot system i.e., a computer that enables the user to choose between more than 2 operating systems at boot time.

This document explains how three varying operating systems can be successfully installed and configured on the same hard disk of a computer thus enabling it to become a "multi-boot" system. The operating systems chosen for this illustration include: either Microsoft Windows 95/98(Second Edition)/Millennium Edition(ME)/NT/2K/XP, FreeBSD 4.8-RELEASE/OpenBSD 3.2-RELEASE/NetBSD 1.6.1 and Red Hat Linux 7.3. I would install a Microsoft Windows operating system first, then either FreeBSD 4.8-RELEASE or

OpenBSD 3.2-RELEASE or NetBSD 1.6.1 and finally round-off by installing Red Hat Linux 7.3 (Valhalla). GNU GRUB is the boot loader used for booting these three operating systems.

1.3 Multi-booting Pros and Cons

An idea or thought as: "Hey! Dual and multi-booting computer systems only has advantages and absolutely no disadvantages" is wrong. Often an important question which readers do ask is: Where does it make sense to multi-boot a PC? The answer to this question is simple: Multi-booting systems only makes sense where you would like to experiment with a number of configurations (or operating systems in general) than you have computers for and more significantly where no data is at risk.

Let us consider a situation like this: Peter has a single PC at home which runs Slackware Linux. All his significant documents and downloaded files from the Internet are stored on it. He now decides to learn and practice hacking the FreeBSD Kernel. Thus, he decides to make his PC a dual-boot system. Converting one's only or in other words, the primary PC at home or at work into a dual or multi-booting system is a bad choice. If a primary home PC must be used as a dual or multi-booting system, it will be much safer to add a second hard disk and leave the first relatively untouched. A powerful boot loader like GRUB will allow booting from the other hard disk.

The Computer Science Department at my university has 15 laboratories for varying purposes that runs hardware ranging from i386s to Sun SPARC. Where data and security is of absolute concern, computers run only one operating system whether Windows 2000 Professional, Red Hat Linux, Slackware, FreeBSD or Sun Solaris. However, we do have "test labs" of about 10-12 PCs each, where we have dual and multi-booting systems running ASP Linux, Red Hat Linux, Slackware, Windows 2000 Professional, FreeBSD and others. This can be considered as an ideal situation.

1.4 List of Assumptions

Though each and every step required has been explained from the very ground-up, yet a few significant assumptions have been taken into consideration while writing this guide, a few of which are as follows:

1. The reader possesses some theoretical and practical experience of partitioning hard disks utilizing Microsoft fdisk, BSD style partition table editor, Linux fdisk and so on.
2. Understands hard disk geometries, concept of a primary partition, an extended partition, logical disks within an extended partition; Linux disk naming and partitioning schemes, BSD style disk labeling and partitioning schemes.
3. Possesses theoretical and practical experience of compiling and configuring custom Linux and BSD Kernels.
4. Knowledge of basic Unix commands common to both Linux and FreeBSD/OpenBSD/NetBSD operating systems. For example, mounting filesystems, editing configuration files like the /boot/grub/grub.conf on Linux and other such trivial tasks.
5. The reader is using an Intel x86 computer system. I would be installing Windows, FreeBSD/OpenBSD/NetBSD and Linux operating systems on IA32 platform.

6. The reader is using a hard disk whose BIOS supports the Logical Block Addressing (LBA) mode of representing data on the disk. By using LBA mode, the 1024 Cylinder Limit on old hard disks is dealt with.

Do not worry if you do not know some of these yet, simply read on because the most important steps would be explained in detail in the forthcoming sections as when required. Anyway, before proceeding any further, make sure you browse through the Frequently Asked Questions (FAQ) section of the guide.

1.5 Acknowledgements

I would like to thank all the following people and projects without the help and active participation of which, this document would never have been possible. Some of them include:

1. **Guyllhem Aznar** <[guyllhem at metalab dot unc dot edu](mailto:guyllhem@metalab.unc.edu)>, Chief coordinator, main contact of TLDP for making this guide possible.
2. **Tabatha Persad** <[tabatha at merlinmonroe dot com](mailto:tabatha@merlinmonroe.com)>, Linux Documentation Project Review Coordinator, for technical reviews and for answering my million queries.
3. **Michael K. Johnson** <[johnsonm at redhat dot com](mailto:johnsonm@redhat.com)>, for the excerpt from his "Linux Information Sheet" at the TLDP site, mentioned in Chapter 1 of this guide on Linux.
4. **Microsoft Corporation** for using important information on Microsoft tools and technologies mentioned in Chapter 2 of the guide.
5. To each and every Open-Source community contributor and to all my friends all around the world.

Additionally, while writing this guide, I consulted the following printed books, online journals, magazines and official papers:

1. Modern Operating Systems, by *Andrew S. Tanenbaum*
2. Understanding the Linux Kernel, by *Daniel P. Bovet, Marco Cesati*
3. Red Hat Linux 8 Bible, by *Christopher Negus*
4. Red Hat Linux Official x86 Installation Guides at <http://www.redhat.com>
5. Linux Gazette and Linux Focus online magazines at <http://www.linuxgazette.com> and <http://www.linuxfocus.org> respectively.
6. The FreeBSD Handbook at <http://www.freebsd.org>
7. The FreeBSD FAQ at <http://www.freebsd.org>
8. The OpenBSD FAQ at <http://www.openbsd.org>
9. The Official Microsoft Windows Installation Guides and FAQs at <http://www.microsoft.com>

1.6 Legalese

Trademarks are owned by their owners.

Although the information given in this document is believed to be correct, the author will accept no liability for the content of this document. Use the tips and examples given herein at your own risk.

Copyright (c) 2003, Subhasish Ghosh

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is located at www.gnu.org/copyleft/fdl.html, in the section entitled "GNU Free Documentation License".

1.7 About the Author

Subhasish "savvy" Ghosh has been working with GNU/Linux and FreeBSD (and more recently OpenBSD, NetBSD, Sun Solaris and other UNIX-variant) operating systems for the past 7 years or so since schooldays. During this time, Ghosh worked for a number of Linux and Linux-related projects, which included Linux Kernel source code hacking; analysis, design and implementation of Inter-Process Communication (IPC) mechanisms in Linux and FreeBSD, Red Hat Linux desktop, workstation and server installations, configurations, troubleshooting and others. Additionally during the past few years, Ghosh has written a number of articles on Linux Kernel compilation, GNOME programming and a host of others for Linux.com, Linuxgazette.com and Linuxfocus.org Online magazines.

Currently, aged 22, Ghosh is an engineering student at the Moscow Power Engineering Institute (Technical University) at Moscow, Russian Federation; specializing in the field of "Informatics and Computer-Science engineering". He is a Microsoft Certified Professional (MCP), MCSD, MCP Certified on NT 4.0; additionally has a host of other GNU/Linux and computer-industry related certifications. His web page can be accessed at <http://groups.msn.com/Linuxdump>.

[Previous](#)

[Home](#)

[Next](#)

2. Frequently Asked Questions (FAQ)

Readers have questions. Thus, this FAQ section has been included for answering some of the most obvious questions from the reader's point of view. All readers are requested to read through this section for finding answers to their personal questions.

After reading through this FAQ entirely, if the reader feels that he/she still has a "question" or two to ask, please feel free to drop a line at subhasish_ghosh@linuxwaves.com.

1. *What is the aim/purpose of this Guide?*
2. *Why have you used "3" operating systems for this guide?*
3. *Did you yourself try out the steps mentioned in this guide? What are the configurations of the computer system on which you tested all the material mentioned in this guide?*
4. *All your installations are "CD-ROM" based installation types. What about "Ethernet", "FTP", "HTTP", "NFS" installation types?*
5. *Why would anyone ever need to run 3 operating systems on the same hard disk of a computer? I want to run 3 operating systems on 2 separate hard disks. What is the big deal with this guide anyway?*
6. *Why Red Hat? Why NOT Debian GNU/Linux, Slackware, Caldera, SuSE, TurboLinux and others?*
7. *Why FreeBSD, OpenBSD and NetBSD operating systems? Why not BSD/OS, Darwin, Irix, Sun Solaris, Digital UNIX, HP-UX, IBM-AIX and others?*
8. *Which Kernel version have you used for GNU/Linux?*
9. *I installed 3 operating systems in my computer as you have illustrated in this guide. And then unfortunately, something went wrong somewhere.*
10. *You used the GNU GRUB for booting all the 3 operating systems. May I know the reason why? Why not use the popular Linux Loader (LILO) or FreeBSD or NT boot loader for this purpose?*
11. *I have older Red Hat, FreeBSD and OpenBSD distributions and/or releases. Will they work?*
12. *I would like to have access to good resources on Linux and FreeBSD. Could you suggest a few?*
13. *I have a few suggestions, new ideas and exciting comments. Can I send them to you?*
14. *I have "installation-specific" questions/queries or something "failed" or "this quite didn't work"? Can I send these reports to you?*
15. *Will you be updating this guide from time to time? Will "my" problems be listed here?*
16. *Does this guide have any newest version on the Internet?*
17. (For people living in and around Moscow, Russia) *I have a computer system and I would love to install 3 operating systems on them. Would you be kind enough to come to my place and do the installation and configuration?*
18. (For people of Russia) *We love Microsoft Windows, FreeBSD and OpenBSD operating systems. But "why" have you NOT used the common Russian GNU/Linux distributions? I mean "ASP Linux", "Black Cat Linux", "ALT*

Linux" and others?

1. *What is the aim/purpose of this Guide?*

This guide can be used by anyone for installing and configuring 3 operating systems (OSes) on the same hard disk of a computer. First, a Microsoft Windows operating system (according to the choice of the reader) is installed, then FreeBSD 4.8-RELEASE or OpenBSD 3.2-RELEASE or NetBSD 1.6.1 is installed and finally, we round-off the guide by installing Red Hat Linux 7.3. Though I always use this particular order, the reader can install in an order comfortable to him/her.

2. *Why have you used "3" operating systems for this guide?*

The current scenario is such that Microsoft Windows operating systems rule the desktops, with GNU/Linux distributions and *BSD releases (FreeBSD/OpenBSD/NetBSD) competing for "total world domination" in the server markets, thus, I thought it would be the absolute best to include all of them in this guide. The first OS is a Microsoft Windows OS, the second one belonging to the *BSD family and the last one a commonly used GNU/Linux distribution (Red Hat). Thus, the number of OSes ended up with "3".

3. *Did you yourself try out the steps mentioned in this guide? What are the configurations of the computer system on which you tested all the material mentioned in this guide?*

I assure everyone that I tested each and every step mentioned in this guide on my personal computer systems. Later they were tested successfully on other computers with varying configurations as well. My computer has a 266.87 MHz Intel Pentium 2 processor, 64MB Physical DIMM RAM, a single 20.0 GB Maxtor IDE HDD and other usual accessories which runs Microsoft Windows 98(SE), FreeBSD 4.8-RELEASE and Red Hat Linux 7.3. The other one with pretty much the same configuration runs FreeBSD 4.8-RELEASE, OpenBSD 3.2-RELEASE and Red Hat Linux 7.3. The general rule of the thumb is: "Better the resources, better the performance of the computer".

4. *All your installations are "CD-ROM" based installation types. What about "Ethernet", "FTP", "HTTP", "NFS" installation types?*

I decided to keep the guide as simple as possible. And since most of us usually use CDs from local CD-ROMs for installing operating systems, I have assumed that the reader undertakes a CD-ROM based installation for installing the covered operating systems. Moreover, this guide is intended for all types of users, ranging from intermediate Linux users to the experienced Unix experts. Since, NFS, HTTP, FTP and other such installation types usually require networks with servers running, which is impossible for an ordinary home-user to access, I have only covered the CD-ROM based installation types. Sorry for the inconvenience caused to the "network" lovers.

5. *Why would anyone ever need to run 3 operating systems on the same hard disk of a computer? I want to run 3 operating systems on 2 separate hard disks. What is the big deal with this guide anyway?*

For lots of reasons. Fun, for learning new installing and booting methodologies, for gaining inside knowledge of "multi-booting", partitioning schemes, boot loaders and so on, or simply because you just cannot afford 2 separate 40.0 GB hard disk drives for the 3 different operating systems you want to tinker with. As for me, I guess the last reason "suits" the best.

Lastly, if you do want, you can always install and configure your computer system with 3 varying operating systems installed in 2 separate hard disks.

6. *Why Red Hat? Why NOT Debian GNU/Linux, Slackware, Caldera, SuSE, TurboLinux and others?*

Since most of my past Linux projects and work in general had been related to the Red Hat Linux distribution; continually installing, configuring and administering Red Hat Linux personal desktops and workstations. Currently, Red Hat Linux is "the" leading Linux solutions provider in the world, with big corporate real-time systems being implemented using the Red Hat operating system software. Moreover, Red Hat documentation and support are easy to find. Thus, Red Hat Linux has been chosen. The other distributions, namely Debian GNU/Linux, Slackware and Mandrake are good, and I will probably add more sections covering other Linux distributions later.

7. *Why FreeBSD, OpenBSD and NetBSD operating systems? Why not BSD/OS, Darwin, Irix, Sun Solaris, Digital UNIX, HP-UX, IBM-AIX and others?*

I personally have networking and source-code hacking experience on FreeBSD and OpenBSD boxes. And, more recently have access to NetBSD 1.6.1 and Sun Microsystems' SunOS operating system in the Solaris Operating Environment. I do "not" have any experience working with Irix, Ultrix, Minix, HP-UX, IBM-AIX, Darwin and others. Thus, FreeBSD, OpenBSD and NetBSD have been chosen.

8. *Which Kernel version have you used for GNU/Linux?*

Linux Kernel 2.4.18-3 has been used for the Red Hat Linux distribution (Valhalla). The latest Kernel at the time of writing, is Kernel 2.4.20.

9. *I installed 3 operating systems in my computer as you have illustrated in this guide. And then unfortunately, something went wrong somewhere. My computer crashed and I lost all the data. Would you be kind enough to take the responsibility for all this mess?*

Well, first my "deepest" condolences on your data-loss and for all this apparent mess. I would love to take all the responsibility, but as it is, I have enough problems of my own to handle. Thus, I am "not" ready to take the responsibility for anything that goes wrong anywhere. At the least, I can say, that all the steps and methods mentioned in this guide have been tested thoroughly on a number of systems with varying configurations, and no problems have been noted so far.

10. *You used the GNU GRUB for booting all the 3 operating systems. May I know the reason why? Why not use the popular Linux Loader (LILO) or FreeBSD or NT boot loader for this purpose?*

As known to everyone, boot loaders play a very vital role while configuring

multiple-booting systems where you can use any suitable boot loader which is capable of booting multiple operating systems. I chose GRUB in this guide for three specific reasons: 1) GNU GRUB is the "default" boot loader in Red Hat distribution releases currently. 2) As mentioned in the 'Release Notes' of Red Hat Linux 8.0, the use of LILO is deprecated and the package would be removed from one of its future releases. 3) GRUB is a very powerful (x86) PC-compatible boot loader which supports a number of operating systems "directly" and other proprietary ones too via "chain-loading" for booting them. Thus, GRUB is used instead of other boot loaders.

11. *I have older Red Hat, FreeBSD and OpenBSD distributions and/or releases. Will they work?*

I am sure they would with a few modifications at places. All steps mentioned should behave in a normal manner if you are using Kernel 2.2.x or higher (for Linux). For FreeBSD users, if you are using not earlier than 4.X-RELEASES, there should not be any problems whatsoever. Just take a deep breath, and start doing as given in this guide. If in case, at some place it does "not" work, make sure you are using the right step and commands etc. Most times, it is human error rather than computer-related error which lets big servers and networks down. Always remember this!

12. *I would like to have access to good resources on Linux and FreeBSD. Could you suggest a few?*

Yes, definitely I could. There is a lot of very good technical information on all the "open-source" operating systems out there on the Internet. For Linux, make sure you often visit <http://www.linuxgazette.com>, <http://www.linuxfocus.org>, <http://www.linux.com> and a host of others. Besides, "[The Linux Documentation Project](#)" (TLDP) provides excellent resources on all topics related to GNU/Linux. As for printed books, these are some of my absolute favorites: Red Hat Linux 8.0 Bible, Understanding the Linux Kernel, Modern Operating Systems, Linux Kernel Programming, The FreeBSD Handbook at <http://www.freebsd.org>, The Design of the UNIX Operating System, The Design and Implementation of the 4.4BSD Operating System. That would be enough for the time-being!

13. *I have a few suggestions, new ideas and exciting comments. Can I send them to you?*

Without asking, please do. Only by hearing from others, I can make this guide a better one in the near future. So, drop a line if you want to.

14. *I have "installation-specific" questions/queries or something "failed" or "this quite didn't work"? Can I send these reports to you?*

Of course do. But please be patient, cause I am busy with lots of things at my university; tests, exams, assignments, dating my "infinitely patient FreeBSD-compatible rock-stable girlfriend", maintaining this guide, other Linux assignments etc. So, I cannot promise to get back to you immediately. I need some time solving your problems.

15. ***Will you be updating this guide from time to time? Will "my" problems be listed here?***

Yes, I would definitely update this guide from time to time with changes, updates, fixes, additions and so on.

16. ***Does this guide have any newest version on the Internet?***

As of now, no, it does not. This is the only place where it will be updated from time to time.

17. (For people living in and around Moscow, Russia) ***I have a computer system and I would love to install 3 operating systems on them. Would you be kind enough to come to my place and do the installation and configuration?***

If you do live in Moscow, or on the outskirts of Moscow; drop me a line. I would come down to your place and do everything that is required for free. Cannot help others living far.

18. (For people of Russia) ***We love Microsoft Windows, FreeBSD and OpenBSD operating systems. But "why" have you NOT used the common Russian GNU/Linux distributions? I mean "ASP Linux", "Black Cat Linux", "ALT Linux" and others?***

All the above-mentioned GNU/Linux distributions are basically Red Hat-based Linux distributions. They heavily rely on Red Hat for package installation procedures, Kernel hacking, Kernel updates etc. I have used ASP Linux 7.2 myself, and though it does offer a few new and exciting features, but I personally prefer working with Red Hat Linux. I have "no" experience using Black Cat and ALT GNU/Linux distributions.

[Previous](#)

[Home](#)

[Next](#)

3. Organization

The content of this *guide* has been split into 7 distinct sections called "Chapters" as illustrated below for ease of searching and reading on behalf of the reader. The reader is expected to choose any one Microsoft Windows OS from the list mentioned in Chapter 2, thus the other sections regarding Microsoft Windows operating systems could be skipped "without" any risk at all. After the reader has chosen which Microsoft OS to install, he/she can proceed with the installation, then proceed with the FreeBSD/OpenBSD/NetBSD and Linux installations sequentially.

Chapter 1: The first Chapter, ***About the operating systems*** covers a brief discussion on Windows OS Family (*I chose to speak a few words on Microsoft Windows XP Professional*), Linux, FreeBSD, OpenBSD and NetBSD operating systems. General information on each operating system, legal stuff etc. is discussed. For more official and exhaustive information, refer to the official web-sites of the products and projects listed there.

Chapter 2: The second Chapter, ***Installing Microsoft Windows*** covers the installation of any one of the Microsoft Windows operating system that the reader chooses to install on a new hard disk. The OS that the reader chooses to install here is the one that would co-exist finally with the FreeBSD/OpenBSD/NetBSD and Red Hat Linux OSes. Disk partitioning and labeling schemes under Windows is discussed in detail herein.

Chapter 3: The third Chapter, ***Installing FreeBSD 4.8-RELEASE*** covers topics required for installing and configuring FreeBSD 4.8-RELEASE with the already existing Microsoft Windows operating system. If the reader chooses to install OpenBSD 3.2-RELEASE instead of FreeBSD 4.8-RELEASE, he/she may skip this chapter and proceed to the next chapter.

WARNING: *Since both FreeBSD and OpenBSD operating systems are POSIX compliant BSD-derivatives, both use the same BSD-style hard disk partitioning and labeling schemes, and both have the same set of commands when viewed from the user-level. Thus, partitioning and filesystems naming covered in FreeBSD section will not be repeated in the next chapter. If it so happens that you are a complete OpenBSD newbie or the term "OpenBSD" sounds like a Greek mythology character, it is best for you to go through all the information in this chapter and install and configure FreeBSD. OpenBSD installation, though simple and neat, is completely a text-based installation procedure and assumes a sound knowledge of hardware, BSD-style hard disk slicing and labeling schemes. Messing up at any one stage, messes up the entire installation and may even render the Microsoft Windows OS (installed initially) totally inaccessible. Experience has taught me to keep proper backup when doing such installations. Do not send problem reports later. You have been warned!*

Chapter 4: The fourth Chapter, ***Installing OpenBSD 3.2-RELEASE*** covers topics required for installing and configuring OpenBSD 3.2-RELEASE with the already existing Microsoft Windows operating system. If the reader chooses to install FreeBSD 4.8-RELEASE instead of OpenBSD 3.2-RELEASE, he/she may skip this chapter and take a peek into the earlier chapter. Partitioning and essential booting process details etc. is discussed herein.

Chapter 5: The fifth Chapter, ***Installing NetBSD 1.6.1*** covers topics needed for installing and configuring NetBSD 1.6.1 with the already existing Microsoft Windows operating system. If the reader chooses to install FreeBSD 4.8–RELEASE or OpenBSD 3.2–RELEASE, he/she may skip this chapter and take a peek into the earlier chapters. Partitioning and essential booting process details etc. is discussed herein.

Chapter 6: The sixth Chapter, ***Installing Red Hat Linux 7.3 (Valhalla)*** covers the topics required for installing and configuring Red Hat Linux 7.3 distribution release with the already pre-existing Microsoft Windows operating system and FreeBSD 4.8–RELEASE/OpenBSD 3.2–RELEASE/NetBSD 1.6.1.

Chapter 7: The seventh and the final Chapter, ***Mounting Filesystems*** contains a detailed discussion on topics required for mounting Linux filesystem (ext2fs) on FreeBSD systems and mounting UFS (or FFS) on Linux systems and other minute yet significant details. Readers interested in gathering working knowledge on Linux filesystems must browse through the contents of this Chapter.

[Previous](#)

[Home](#)

[Next](#)

[Previous](#)[Home](#)[Next](#)

3.1 About the operating systems

A general overview of the operating systems that have been covered in this *guide* has been provided here. Readers must note that these are my personal views (and comments) and have nothing to do with the views of the Linux/FreeBSD/OpenBSD/NetBSD or in general terms, the Open-Source community.

3.1.1 Microsoft Windows: Microsoft Corp. has a long (yet fatally weak and buggy) list of operating systems, targets ranging from the mild and feeble home-user to the truly multi-tasking, multi-processing, multi-programming, fault-tolerant, hack-resilient real-time systems; from MS Windows 95 to the most recently launched Microsoft Windows Server 2003. The reader is free to choose any OS according to his/her own personal taste and/or targeted environment. I personally do "not" consider Microsoft Windows 2000 family of Oses (which includes Windows 2000 Professional, Windows 2000 Server, Windows 2000 Advanced Server and Windows 2000 Datacenter server) a very good choice for "corporate" environments. If the **PASSME** theme of Software engineering is considered, where PASSME = Performance, Availability, Scalability, Security, Maintainability and Extensibility features of any software under scrutiny; Microsoft Windows 2000 servers (and operating systems, in general) are rated much lower than UNIX SVR4, *BSDs, Sun Solaris, Linux and any of the other commercial variants of UNIX like IBM-AIX, HP-UX, Digital UNIX and so on.

Anyway, I personally believe, "freedom of choice of OS is a birthright for all". So, it is the reader's choice, whether he/she uses Windows 98 or Windows 2000 Professional. Personally, I like tinkering with Microsoft Windows XP, which is the next version of Microsoft Windows beyond Windows 2000 and Windows Millennium. Windows XP brings the convergence of Windows operating systems by integrating the strengths of Windows 2000 – standards-based security, manageability and reliability with the best features of Windows 98 and Windows Me, Plug and Play, easy-to-use user interface, and innovative support services to create the best Windows yet. Windows XP is not a "free" operating system as like Linux, FreeBSD, NetBSD, OpenBSD and other such open-source software projects.

- ◆ For Microsoft Windows XP Professional "Overview", visit <http://www.microsoft.com/catalog/display.asp?site=10798&subid=22&pg=1>
- ◆ For "System requirements", visit <http://www.microsoft.com/catalog/display.asp?site=10798&subid=22&pg=3>
- ◆ Visit Microsoft Windows XP Professional homepage at <http://www.microsoft.com/windowsxp/pro/default.asp>

3.1.2 Linux: Linux is the kernel or core of an operating system called GNU/Linux system, where the GNU Project provides the software and applications that runs on the Linux kernel. The Linux kernel, originally written by Linus Benedict Torvalds, is a UNIX-clone, which is POSIX compliant and was initially targeted towards the Intel x86 architecture. As rightly mentioned in the "Linux Information Sheet" by Michael K. Johnson at <http://www.tldp.org>, "...Linux is a completely free reimplementation of the POSIX specification, with SYSV and BSD extensions (which means it looks like Unix, but does not come from the same source code base), which is available in both source code and binary form. Its copyright is owned by Linus Torvalds ; and other contributors, and is freely redistributable under the terms of the

GNU General Public License (GPL). A copy of the GPL is included with the Linux source; you can also get a copy from <ftp://prep.ai.mit.edu/pub/gnu/COPYING...>.

Readers must note the fact that though many Linux users worldwide refer to the GNU/Linux operating system simply as "Linux", it is more appropriate to refer to the operating system as a GNU/Linux system and when talking about the *kernel* in particular just to call it Linux. Moreover, a very popular misconception amongst new Linux users is: "Hey! Linux is UNIX!". No, absolutely "not". Once again, readers must note: The Linux kernel is a UNIX-clone, in other words a reimplementations of the UNIX-Kernel, but it is not UNIX itself. On the other hand, the BSDs, FreeBSD and OpenBSD for example, are not kernels merely. They are more closer to the real UNIX (AT&T's Official Release UNIX SVR4). They represent an entire operating system with a whole set of applications, tools, user utilities, shells, games, documentation, developer utilities, system and user-level binaries and full source code for the operating systems.

3.1.3 FreeBSD: FreeBSD which stands for the Free version of Berkeley Software Distribution operating system software (though all the BSDs out there are "open-source" or "free" in nature) is technically speaking, the most powerful, professional quality and advanced UNIX-compatible operating system software on earth today. FreeBSD is originally based on 4.4BSD-Lite (with a minor inclusion from the 4.4BSD-Lite2 code base too), developed at the University of California, Berkeley, under the authorization of the Computer Systems Research Group (CSRG), Computer Science Division, Department of Electrical Engineering and Computer Science at Berkeley. FreeBSD operating system (just like a Linux distribution) comes with a whole set of packages, including user applications, tools, user utilities, shells, games, documentation, developer utilities, system and user-level binaries and full source code for the operating systems. The FreeBSD Project web-site can be accessed at: <http://www.freebsd.org>.

Recently, a lot of media spotlight has put the GNU/Linux system (more specifically the Linux Kernel) in a stardom status. Some rate Linux as the best ever written UNIX-clone that would spell disaster for UNIX and other UNIX-clones and UNIX-compatibles out there in the market. But in reality it is not so. Readers must note: Just as the power of an automobile comes from the quality of the engine running under it's hood, similarly features like stability, reliability, extensibility and robustness of an operating system software comes from the internal structure of the Kernel or the core of that corresponding operating system. I personally feel that both the Linux and the FreeBSD Kernel are great works of creativity and tremendous effort and just a user-level or technical comparison between the two is meaningless. Situations where memory is a scarcity and moreover gets crunched and stressed upon real hard, the Linux Kernel breaks down. FreeBSD with it's highly optimized and advanced VM/Swap system comes to the rescue! On the other hand, under given situations, Linux may outperform FreeBSD by a mile. The significant point for readers to note and remember is that the Linux Kernel favors simplicity over highly optimized and dynamic yet more complex code whereas the FreeBSD Kernel optimizes performance by introducing more complex code and heavily optimizing the code base by rearranging and rewriting some of the more important internal algorithms and data structures from time to time.

A complete and exhaustive discussion on Linux and FreeBSD Kernels is beyond the scope of this guide. If readers find it interesting, they may search the Internet for more Linux and/or FreeBSD specific information. I would suggest the readers to read "Understanding the Linux Kernel" by Bovet and Cesati for getting more in-depth information on the inner-workings of the Linux Kernel, "The Design and Implementation of the 4.4BSD Operating System" for

FreeBSD/OpenBSD and so on, and "The Design of the UNIX Operating System" by M. Bach for UNIX SVR4 internals.

3.1.4 OpenBSD: The BSD family, other than the dominant big-brother FreeBSD, also consists of OpenBSD, NetBSD, BSD/OS and Darwin operating systems. OpenBSD is a fully functional, multi-platform UNIX-like Operating System based on Berkeley Networking Release 2 (Net/2) and 4.4BSD-Lite. The OpenBSD team strives to achieve what is called "a secure by default" status. This means that an OpenBSD user should feel safe that their newly installed machine will not be compromised. This "secure by default" goal is achieved by taking a proactive stance on security. FreeBSD –RELEASES target stability and reliability under the most adverse of conditions whereas OpenBSD –RELEASES target optimum security in corporate environments. The OpenBSD operating system software has been labeled "the proactively secure Unix-like operating system". The OpenBSD Kernel is heavily optimized with security features thereby providing customers with utmost secure and hack-resilient real-time systems. The OpenBSD Project web-site can be accessed at: <http://www.openbsd.org>.

The OpenBSD operating system software has integrated strong cryptography into the base system. A fully functional IPsec (Internet Protocol Security) implementation is provided as well as support for common protocols such as SSL (Secure Sockets Layer) and SSH (Secure Shell). Network filtering and monitoring tools such as packet filtering, NAT (Network Address Translation), and bridging are also included. For high performance demands, support for hardware cryptography has also been added to the base system. OpenBSD provides as many security options as possible to allow the user to enjoy secure computing without feeling burdened by it.

3.1.5 NetBSD: The NetBSD kernel focuses on clean design and well architected solutions. NetBSD supports a massive range of hardware platforms from a single source tree, including simultaneous release across all platforms, and continues to attract users and experienced developers despite lack of media exposure and commercial backing – all thanks to attention to code quality. The NetBSD operating system kernel targets "portability" and tries to run on as many platforms as possible. The NetBSD Project web-site can be accessed at: <http://www.netbsd.org>

- ◆ For browsing through the *Official FreeBSD Handbook*, visit: <http://www.freebsd.org>
- ◆ *The official sources for FreeBSD are available via anonymous FTP from:*
<ftp://ftp.FreeBSD.org/pub/FreeBSD/>
- ◆ *For a comparative study between FreeBSD, Linux and Microsoft Windows 2000, check out this document at http://people.freebsd.org/~murray/bsd_flier.html*

The following table exhibits the main "target" of every operating system covered in this guide:

Windows	Ask Mr. Gates himself... Maybe our purses...
Linux	Linux targets new, exciting, and cutting-edge features like HyperThreading, USB, Wireless NICs and so on.
FreeBSD	FreeBSD targets stability and robustness; optimizing the kernel code base for optimum performance under heavy network-loads.
OpenBSD	OpenBSD targets optimum security. It is the most proactively-secure UNIX-like OS.

NetBSD

NetBSD targets portability. To run on as many platforms as possible is its primary target.

[Previous](#)

[Home](#)

[Next](#)

[Previous](#)[Home](#)[Next](#)

3.2 Installing Microsoft Windows

I have used a Microsoft Windows operating system as the "first" OS on my computer. I have MS Windows 98 (SE) installed on my computer. The reader can choose any one of the Microsoft OSes mentioned below in the listed form. Suppose, you would too like to use Microsoft Windows 98 (SE) as your first OS. Then read the information given in Section B: Installing MS Windows 98 (SE) and after reading and doing likewise as mentioned in that section, proceed forward with installing FreeBSD 4.8-RELEASE or OpenBSD 3.2-RELEASE or NetBSD 1.6.1.

Thus, we have the following sections:

- ◆ [**Section A: Installing MS Windows 95**](#)
- ◆ [**Section B: Installing MS Windows 98 \(SE\)**](#)
- ◆ [**Section C: Installing MS Windows Millennium \(ME\)**](#)
- ◆ [**Section D: Installing MS Windows NT \(4.0\)**](#)
- ◆ [**Section E: Installing MS Windows 2000 \(Professional\)**](#)
- ◆ [**Section F: Installing MS Windows XP \(Home/Professional\)**](#)

One can use the Windows **fdisk** and **format** tools for partitioning a new unformatted hard disk. I have assumed that the reader has a new unformatted hard disk which he/she wishes to partition and then format for installing a Microsoft Windows OS. The material mentioned right below applies to Microsoft Windows 95, Microsoft Windows 98 (SE) and Microsoft Windows Millennium (ME). If you plan to install Windows NT, Windows 2000 or Windows XP, then skip this section and go to the part referring to the sections D, E and F.

Section A: Installing MS Windows 95

Section B: Installing MS Windows 98 (SE)

Section C: Installing MS Windows Millennium (ME)

The material mentioned below applies to Microsoft Windows 95, Microsoft Windows 98 (SE) and Microsoft Windows Millennium (ME). Thus, if you are installing any one of these, make sure you read through all the content mentioned herein which describes the Fdisk and Format tools and explains how to use them to partition or repartition a hard disk. The material describes the following topics:

- ◆ How to Use the Fdisk and Format Tools
- ◆ Important Considerations Before You Use the Fdisk and Format Tools
- ◆ How to Partition and Format a Master Hard Disk

Before you install your operating system, you must first create a primary partition on the hard disk (disk 1) on your computer, and then format a filesystem on that partition. The Fdisk tool is an MS-DOS-based tool that you can use to prepare (partition) a hard disk. You can use the Fdisk tool to create, change, delete, or display current partitions on the hard disk, and then

each allocated space on the hard disk (primary partition, extended partition, or logical drive) is assigned a drive letter. Disk 1 may contain one extended partition, and a second hard disk may contain a primary or extended partition. An extended partition may contain one or more logical MS-DOS drives.

After you have used the Fdisk tool to partition the hard disk, you can use the Format tool to format those partitions with a filesystem. The filesystem File Allocation Table (FAT) allows the hard disk to accept, store and retrieve data. Windows 95, Windows 98 (SE), Windows Millennium (ME) and Windows 2000 support the FAT16 and FAT32 filesystems. When you run the Fdisk tool on a hard disk that is larger than 512 megabytes (MB), you are prompted to choose one of the following file systems: FAT16 or FAT32. The FAT16 filesystem has a maximum of 2 gigabytes (GB) for each allocated space or drive letter. For example, if you use the FAT16 filesystem and have a 6-GB hard disk, you can have three drive letters (C, D, and E), each with 2 GB of allocated space. As for the FAT32 filesystem, it supports drives that are up to 2 terabytes in size and stores files on smaller sections of the hard disk than the FAT16 filesystem does. This results in more free space on the hard disk. Please note that the FAT32 filesystem does not support drives that are smaller than 512 MB. When you run the "fdisk" and "format" commands, the Master Boot Record (MBR) and file allocation tables are created. The MBR and file allocation tables store the necessary disk geometry that allows hard disk to accept, store, and retrieve data.

Please note that I have assumed that you own a computer system which can run without glitches and that you have access to a bootable Microsoft Windows CD-ROM. If you do run into problems booting from the CD-ROM or using Start-up disks, please consult your hardware manufacturer to obtain the CD-ROM device driver(s). For partitioning a master hard disk, you have to run the fdisk command. First insert the Startup disk in the floppy disk drive, restart your computer, and then use one of the following methods, depending on your operating system.

For a Windows 95/98/ME Startup disk: At a command prompt, type fdisk, and then press enter. If your hard disk is larger than 512 MB, you receive the following message: *Your computer has a disk larger than 512 MB. This version of Windows includes improved support for large disks, resulting in more efficient use of disk space on large drives, and allowing disks over 2 GB to be formatted as a single drive.*

If you enable large disk support and create any new drives on this disk, you will **NOT** be able to access the new drive(s) using other operating systems, including some versions of Windows 95 and Windows NT, as well as earlier versions of Windows and MS-DOS. In addition, disk utilities that were not designated explicitly for the FAT32 file system will not be able to work with this disk. If you need to access this disk with other operating systems or older disk utilities, do not enable large drive support.

Do you wish to enable large disk support?

If you want to use the FAT32 file system, press Y and then press enter. If you want to use the FAT16 file system, press N, and then press enter. After you press enter, the following Fdisk Options menu is displayed:

1. Create DOS partition or Logical DOS Drive
2. Set active partition
3. Delete partition or Logical DOS Drive

4. Display partition information
5. Change current fixed disk drive

Please note that option 5 is available only if you have two physical hard disks in the computer.

Press 1 to select the Create DOS partition or Logical DOS Drive menu option, and then press enter. Press 1 to select the Create Primary DOS Partition menu option, and then press enter. After you press enter, you receive the following message:

Do you wish to use the maximum available size for primary DOS partition?

After you receive this message, use one of the following methods, depending on the file system that you selected.

For a FAT32 File System: If you press Y for the FAT32 file system (in step 2) and you want all of the space on the hard disk to be assigned to drive C, press Y, and then press enter. Press ESC, and then press ESC to quit the Fdisk tool and return to a command prompt.

For a FAT16 File System: If you press N for the FAT16 file system (in step 2), you can accept the default 2 GB size for the partition size, or you can customize the size of the partition.

If you want to customize the size of the partitions (drive letters) on the hard disk, press N, and then press enter. A dialog box is displayed in which you can type the size that you want for the primary partition in MB or percent of disk space. Note that for computers that are running either Windows 98 (SE) or Windows ME, Microsoft recommends that you make the primary partition at least 500 MB in size. Type the size of the partition that you want to create, and then press enter. Press ESC to return to the Options menu. To assign drive letters to the additional space on the hard disk, press 1, and then press enter. Press 2 to select the Create Extended DOS Partition menu option, and then press enter. You receive a dialog box that displays the maximum space that is available for the extended partition. You can adjust the size of the partition or use the default size. Note that the default maximum space is recommended, but you can divide the space between multiple drive letters. Type the amount of space that you want, press enter, and then press ESC. The Create Logical DOS Drive(s) in the Extended DOS Partition menu is displayed. This is the menu that you can use to assign the remaining hard disk space to the additional drive letters. Type the amount of space that you want to assign to the next drive letter in the Enter logical drive size in Mbytes or percent of disk space (%) box, and then press enter. After this, you have to activate the partition from which you plan to boot. This, usually is the drive C. So, press 2 to select the Set active partition menu option, and then enter the number of the partition you want to make active.

After you create the partitions, you must format the partitions for accessing and using them. If you are using a Windows 95 Startup disk, a command prompt is displayed and you can skip to step 2. If you are using a Windows 98 Second Edition or a Windows ME Startup disk, select the Start computer without CD-ROM support menu option when the Windows 98 Startup menu is displayed. When a command prompt is displayed, type "**format c:**", and then press enter. This command formats drive C (or your "active" drive). For all other partitions, type format drive: (where drive is the letter of the partition that you want to format).

Readers must note that I speak about "partitioning a hard disk" using Microsoft fdisk utility in general here in this section in order to provide the absolute newbies with general information on how to partition and format a new hard disk drive.

Section D: Installing MS Windows NT (4.0)

Section E: Installing MS Windows 2000 (Professional)

Section F: Installing MS Windows XP (Home/Professional)

For installing any one of these 3 above-mentioned OSes, first use fdisk tool to partition your hard disk. Then comes to choosing the filesystem type.

During a new installation of Windows NT, 2000 or XP, you may have to choose which file system your computer should use. On my other PC, I personally use Microsoft Windows XP Professional which supports:

FAT32: An enhanced version of the file allocation table (FAT) system that is standard on all Windows operating systems starting with later (32-bit) versions of Windows 95. The FAT32 system can be used on large hard disks, from 512 megabytes (MB) to 32 gigabytes (GB).

NTFS: The NT file system (NTFS) is used with the Windows NT, Windows 2000, and Windows XP operating systems. NTFS provides enhanced reliability, stability, and security, and supports large hard disks of up to 2 terabytes (TB).

But very "important" information: The conversion to NTFS is one-way only; if you convert your FAT or FAT32 file system to NTFS you cannot convert your hard disk back to FAT later.

If you are not sure which file system to use, I would suggest using FAT32. If you want to change your file system, here are a few recommendations:

1. Use FAT32 if your hard disk is smaller than 32 GB.
2. Use FAT32 if you want to install more than one operating system on your computer.
3. Use NTFS if your hard drive is larger than 32 GB and you are running only one operating system on your computer.
4. Use NTFS if you want enhanced file security.
5. Use NTFS if you need better disk compression.

You can choose any filesystem type that you wish. After choosing the filesystem, proceed with the Windows NT/2000/XP installation normally. After installation, feel free to configure your new system. I personally chose the "FAT32" filesystem for formatting the partition in which I installed Microsoft Windows XP Professional (Final Release). It formed the first OS on my computer.

Summary of the actual steps:

Okay, enough said about the different Microsoft operating systems. Now let us look more closely how exactly we would partition the hard disk. As mentioned above, we use Microsoft fdisk, then create a Primary DOS Partition of about 2000MB or so. On my PC, I have a primary partition of 999MB which contains Microsoft Windows 98 (SE). I have used such a small space cause I hardly do any work on Windows thereby reserving space for FreeBSD and Linux! Readers can choose a bit more space if they feel like doing so. After creating this partition, do not forget to format it using the Format tool. On my system, I have a FAT32 Primary partition.

After creating this Primary Partition (suppose say, of 999MB), we do not (I repeat: do not) need to create any more partitions using Microsoft fdisk utility. For example, I have a 20.0 GB Maxtor HDD. I create a primary partition of about 1000MB using fdisk. The obvious question from the reader is: What about the rest 19GB? What happens to it? Well the answer is simple. I leave it untouched because I would be dividing this available space into 2 halves for installing FreeBSD or OpenBSD and Linux while using the FreeBSD Partition editor or OpenBSD Partition editor while installing FreeBSD 4.8-RELEASE or OpenBSD 3.2-RELEASE or NetBSD 1.6.1. So, no need to worry absolutely and just take the ride with me. I promise, by the time you are done reading this guide, not a single MB on your hard disk will be wasted.

Thus, before we finally move on to the next section where we install and configure FreeBSD 4.8-RELEASE or OpenBSD 3.2-RELEASE or NetBSD 1.6.1, I would like to round-off this chapter with a "Partition check". I have the following partition table on my computer after this step (as observed from Microsoft fdisk):

Display Partition Information

Partition	Status	Type	Volume_Label	Mbytes	System	Usage
C:1	A	PRI DOS	WIN	1000	FAT32	5%

Once this done, reboot your computer and using a bootable Microsoft Windows CD-ROM (depending on your operating system of choice), install the OS as usual. Do the usual system configuration and log into Windows to check whether everything works fine. That is all for this section, now take a break, have a coffee and proceed to the next chapter, [Chapter 3: Installing FreeBSD 4.8-RELEASE](#).

[Previous](#)

[Home](#)

[Next](#)

3.3 Installing FreeBSD 4.8-RELEASE

This Chapter focuses on installing FreeBSD 4.8-RELEASE and multi-booting the system so that it can coexist successfully with the already existing Windows operating system on the computer. Like the Windows installation, it too focuses on a CD-ROM based installation. Experts may proceed as usual without help, as for the newbies out there, make sure you check out the *Frequently Asked Questions for FreeBSD 2.X, 3.X and 4.X* at the "The FreeBSD Documentation Project" at <http://www.freebsd.org>. For a step-by-step FreeBSD Installation procedure with screenshots, please refer to the Chapter2 : Installing FreeBSD of the FreeBSD Handbook at http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/index.html. These and the other documentation available on the FreeBSD Project homepage provides exhaustive and extensive coverage of all aspects related to the FreeBSD software. Make sure you grab as much information as you can from this site before proceeding with this guide.

Readers must note that at the time of writing, 2 parallel branches of the FreeBSD operating system exist. The latest -STABLE release is the FreeBSD 4.8-RELEASE while the latest -CURRENT release is the FreeBSD 5.1-RELEASE. In this Chapter, we would discuss some theoretical aspects of installing FreeBSD 4.8-RELEASE, disk partitioning (better known as slicing in FreeBSD terminology) and then move on to the actual installation itself. Thus, the contents of this Chapter may be categorized in the following sections mentioned below:

1. **Slicing in FreeBSD**
 2. **Filesystems in FreeBSD**
 3. **Summary of Installation Steps**
-

1. Slicing in FreeBSD

Before installing FreeBSD, it would be a good idea to take a brief tour of what FreeBSD calls "slicing". If you already know this stuff, feel free to skip this part and read the section entitled "[Summary of Installation Steps](#)".

The FreeBSD operating system partitions (a hard disk) and labels partitions according to a particular chosen scheme which is similar to other BSD-based Unix systems such as NetBSD, OpenBSD, Ultrix, Digital Unix and SunOS. Let us see how the FreeBSD software "slices" your hard disk and then "creates partitions in it" for use. Each partition—that-contains—a-filesystem is stored in what FreeBSD calls a "slice". Slice is FreeBSD's term for what were earlier called partitions. Slices are numbered, starting at 1, through to 4 (In Windows terminology, they would be called as "primary partitions", starting at 1, through to 4). Thus, the reader must note that "a primary partition" in Windows refers to a "slice" in FreeBSD terminology, and "an extended partition containing logical disk drives in it" is simply called as "partitions" in FreeBSD terms. A logical disk drive within an extended partition may also be called as "a logical slice inside a physical slice" in FreeBSD. It may appear a bit awkward at times, but the sooner the reader gets used to these terms, the better!

Now, the important part. There exists a few operating systems which can boot even if their corresponding boot files are stored within a logical disk drive of an extended partition. For FreeBSD, this mechanism does "not" work. FreeBSD necessarily needs one of the slices (that is the 4 entries in the partition table on your computer's hard drive, which in Windows terminology would be called as a "primary partition"). It then uses a program called "disklabel" for making upto eight partitions in this slice. The reader must note: You cannot install FreeBSD in an extended partition made by Linux (or DOS).

FreeBSD labels hard drives and partitions as follows:

- ◆ First IDE hard disk drive (Primary Master) – /dev/ad0
- ◆ Second IDE hard disk drive (Primary Slave) – /dev/ad1
- ◆ Third IDE hard disk drive (Secondary Master) – /dev/ad2
- ◆ Fourth IDE hard disk drive (Secondary Slave) – /dev/ad3

where, /dev is the directory under the root ("/") directory in FreeBSD which contains all the device special files associated with devices. Readers must note that I assume readers are using a FreeBSD 4.X–RELEASE. IDE hard disk drives are named as wd before the FreeBSD 4.0–RELEASE.

What about SCSI drives? Well, FreeBSD labels that too!

- ◆ First SCSI hard disk drive (Primary Master) – /dev/da0
- ◆ Second SCSI hard disk drive (Primary Slave) – /dev/da1
- ◆ Third SCSI hard disk drive (Secondary Master) – /dev/da2
- ◆ Fourth SCSI hard disk drive (Secondary Slave) – /dev/da3

where, /dev is the directory under the root ("/") directory in FreeBSD which contains all the device files associated with devices.

Now, let us talk about the partitioning scheme under FreeBSD. It is just like in Linux, only the way the partitions are named is different. In FreeBSD, we first consider a particular hard disk drive, whether an IDE or a SCSI hard disk drive. Then, we consider the partitions on it. For example, the partitions on an IDE drive are named in the following way (/dev/ad0 is used as an example):

- ◆ First primary partition – /dev/ad0s1
- ◆ Second primary partition – /dev/ad0s2
- ◆ Third primary partition – /dev/ad0s3
- ◆ Fourth primary partition – /dev/ad0s4

These above mentioned drives may contain partitions as well. FreeBSD labels them too and each label has it's own corresponding mount point too. Let us consider an example. A computer has /dev/ad0 as its hard disk named, contains 2 slices; the first slice is a FAT32 partition and the second a BSD/i386 partition containing 4 logical slices [logical disk drives] in it. So, in this case, how would the partition table look like and mean? Let us see how FreeBSD labels this scheme. FreeBSD would represent this as:

- /dev/ad0
 - /dev/ad0s1
 - /dev/ad0s2
 - /dev/ad0s2a
 - /dev/ad0s2b
 - /dev/ad0s2c
 - /dev/ad0s2d
 - /dev/ad0s2e

- /dev/ad0s2f

where, ad0s2a refers to the "first partition (a) on the second slice (s2) on the first IDE disk (ad0)". In the above example, /dev/ad0s1 is the slice containing FAT32 filesystem and /dev/ad0s2 is the second slice on the hard disk drive under scrutiny. Readers must note that FreeBSD labels slices from /dev/ad0s1 onwards to /dev/ad0s4, with a maximum of 8 possible logical slices named as a, b, c, d, e, f, g and h. In this example, the "logical slices" a, b, e and f have their corresponding mount points in FreeBSD.

Here, the mount points are as follows:

- a = / (the root directory)
- b = swap filesystem
- e = /var
- f = /usr

For more exhaustive information, please refer to the "FreeBSD Handbook".

2. Filesystems in FreeBSD

Microsoft Windows operating systems use FAT16, FAT32, NTFS 4.0 and NTFS 5.0 filesystems for storing and retrieving data. The recently released NTFS 5.0 is implemented in all of the Windows 2000 operating systems. The Linux Kernel uses the "Third Extended native filesystem", denoted as ext3fs. Linux also supported the ext (now obsolete and no longer supported) and ext2 filesystems. The "Second Extended filesystem" (ext2fs) is an advanced hierarchical filesystem developed for the Linux operating system, which included advanced features like a maximum file size of 2.0 GB, a maximum file name length of 255 characters and support for three time/date stamps. They are the date of creation, date of last modification and date of last access. The "Third Extended filesystem" has all these features, and additionally it implements the "journaled file system architecture" (JFS). The JFS architecture ensures the integrity of data stored on a hard disk in the event of an unplanned shutdown, such as one caused by a power outage. A journaled filesystem maintains a log of all read and write events; this log enables the disk to be restored to its last stable state and additionally allows incompletely written data to be restored to the maximum possible extent.

The FreeBSD operating system basically uses the Unix File System (UFS) architecture for its underlying filesystem. Berkeley made more advancements to it referred to as the "Berkeley Improvements" or "Berkeley Extensions". This filesystem, currently used by FreeBSD as its native filesystem is often called the "Fast Filesystem" or FFS, because access reads and writes to data stored in FFS is very fast. The architecture is a little more complex than Linux's ext2fs. It offers a better way to insure filesystem data integrity, mainly with the "**softupdates**" option. This option decreases synchronous I/O and increases asynchronous I/O because writes to a UFS filesystem are not synced on a sector basis but according to the filesystem structure. This ensures that the filesystem is always coherent between two updates.

The FreeBSD filesystem also supports file flags, which can stop a would-be intruder dead in his or her tracks. There are several flags that you can add to a file such as the immutable flag. The immutable (schg) flag would not allow any alteration to the file or directory unless you remove it. Other very handy flags are append only (sappnd), cannot delete (sunlnk), and archive (arch). When you combine these with the kernel security level option, you have a very impenetrable system. In short, the FreeBSD Kernel uses UFS = UFS+FFS+Softupdates+dirpref+dirhash functionality for storing and retrieving data on

disks.

3. Summary of Installation Steps

The partitioning scheme and filesystems covered in FreeBSD, its time for us to move on to the actual FreeBSD installation. For a complete step-by-step coverage of FreeBSD installation, please consult the installation guide available on the FreeBSD Project web site. As in the previous Chapter, here also, I would be mentioning only the significant steps for the installation. Use these steps to install FreeBSD on your computer and configure it for "multiple-booting":

1. Right now, we have a computer running Microsoft Windows OS. On my PC, it is Windows 98 (SE). I put a bootable FreeBSD 4.8-RELEASE CD-ROM in the CD-ROM drive and reboot the system.
2. The computer boots, reads the CD-ROM, displays the usual hardware probing messages on the screen, and presents you with the Kernel configuration menu. Exit it using a 'Q' and proceed to the FreeBSD System Installation and Configuration Utility. In FreeBSD it is called the /stand/sysinstall utility.
3. Once the /stand/sysinstall screen appears, select a Novice, Custom or Express install. I choose the Custom installation because it provides the optimum flexibility. For novice users, I would suggest using the Recommended installation. Proceed as usual, till you reach the FreeBSD Fdisk editor.
4. Here select the unused space and create 2 approximate equal partitions. Thus, 2 slices would be created (in addition to the one that already exists). One would be used for installing FreeBSD and the other for Red Hat Linux later. Readers with intermediate and/or expert FreeBSD knowledge should find this easy to do. After creating these 2 partitions, there should be no free space of your hard disk. Exit saving all the changes and proceed as usual to the FreeBSD Disk label editor. Readers must note: While creating the Linux slice, the filesystem ID should be 131. This creates a Linux native filesystem (ext2fs filesystem). Otherwise, the default FreeBSD UFS filesystem would be created.
5. The next screen allows you to install a boot manager. Since Microsoft Windows 98 (SE) overwrites the MBR rendering any other OS unbootable (so mean man!), you should choose to install the FreeBSD boot manager in the Master Boot Record (MBR) of your hard disk. I have installed FreeBSD alongside another operating system on the same hard disk, and I want FreeBSD to boot the other operating system when I start the computer. That is the reason, I installed the FreeBSD boot manager into the MBR. Once done, we enter the Sysinstall Disklabel Editor.
6. The Sysinstall Disklabel Editor represents the already existing Windows FAT32 partition. Readers must note that FreeBSD names this partition as a FreeBSD slice but without an appropriate "mount point". Here, I would create the logical partitions for the FreeBSD slice (which is /dev/ad0s2).
7. On my PC, the Windows FAT32 partition is named /dev/ad0s1 (C: in DOS). In other words, it is the first slice on the IDE hard disk of my computer. Disklabel can automatically create partitions for you and assign them default sizes if you press the key A. Depending on the size of the disk you are using the defaults may or may not be appropriate. For me, it works most of the time! To quote from the FreeBSD Handbook, "...Beginning with FreeBSD 4.5, the default partitioning assigns the /tmp directory its own partition instead of being part of the / partition. This helps avoid filling the / partition with temporary files...". This is an important consideration if you

have a lot of temporary files to deal with. I create a /, swap, /usr, /var and other usual partitions in my FreeBSD slice (named as /dev/ad0s2).

8. Once the slices created and partitions within the FreeBSD slice set and done, we proceed to choose what to install on the system. Depending on the intended use and resources at your disposal (or available), you must make the correct choice of what to install and what NOT to install. You are the best judge of that. For example, Mr. A just wants to check out FreeBSD 4.8-RELEASE for fun! Well, a "Minimal Install" is the best choice for him. It saves him both time and hard disk space. His brother, Mr. B is a hot-shot Kernel Developer. He wants access to documentation, full system binaries and the Kernel source code. He would go in for a "Kern-Developer" canned distribution set. And if you have enough resources at your disposal or if it so happens you do not want to use the `pkg_add` tool to install packages later or from the ports collection manually, well, as it so happens, you can always go in for the "All" install option. This would install everything on the system. Thus, we see that the predefined options range from installing the smallest possible configuration (Minimal) to everything (All). Those who are new to Unix and/or FreeBSD should almost certainly select one of these canned options. Readers must note: If a graphical user interface is desired then a distribution set that is preceded by an X should be chosen. The configuration of XFree86 and selection of a default desktop is part of the post-installation steps.
9. Next, choose the installation media from the list of available choices. I guess you are doing a Local CD/DVD-ROM installation just like me! Using the arrow keys highlight "Install from a FreeBSD CD/DVD", and press Enter to proceed with the installation. (Concerned, [OK] was highlighted). If it so happens that you want to undertake a different kind of installation, choose the appropriate option and follow the steps.
10. Next appears the "Committing to the Installation" screen. You would be asked whether you are sure of installing FreeBSD on this system or not. Select [Yes] and press enter to proceed. The installation can now proceed if desired. Readers must note: This is absolutely the last chance for aborting the installation to prevent changes to the hard drive.
11. The installation time will vary according to the distribution chosen, installation media used, and the speed of the computer. There will be a series of messages displayed indicating the status. The installation is complete when the following message is displayed: "Congratulations! You now have FreeBSD installed on your system....".
12. Next comes the post-installation steps. It usually involves chores as "Network Device Configuration", "Configure Gateway", "Configure Internet Services", "Anonymous FTP", "NFS Server", "NFS Client", setting up a "Security Profile", "System Console Settings", "Setting The Time Zone", "Adding Users and Groups" and so on. For an exhaustive covering of each and every step along with helpful screenshots, please refer to the FreeBSD Handbook at <http://www.freebsd.org>.
13. Once you have completed the post-installation steps, exit the `/stand/sysinstall` system installer utility, take out the CD-ROM, and reboot the system. When your computer reboots, since you have installed the FreeBSD boot loader into the MBR (Master Boot Record) of your hard disk, you would normally get a following kind of prompt on your screen. It display something like: **F1 DOS, F2 FreeBSD, F3 Linux**; each one in a separate line. The Default: F1 is also shown. FreeBSD uses a 3-stage bootstrap by default, and this is actually the preceding bootblock, named `/boot/boot0`, which lives on the Master Boot Record, the special part of the disk that the system bootstrap looks for and runs, and it simply shows a list of possible slices to boot from.

14. Press F1 to boot into Microsoft Windows OS. On my computer, I successfully booted into Windows 98 (SE). So far so good! I rebooted and pressed F2 this time, and I successfully booted into FreeBSD as well. Bingo! You already have a dual-boot system which uses the FreeBSD boot loader to boot 2 operating systems.
 15. Depending on the configuration of your computer, you will receive bootup messages on your screen when you boot into FreeBSD for the first time. Though the messages scroll off the screen rather fast for you to read anything productive, you can always read it later by using the `"/sbin/dmesg | more"` command. Finally, you will arrive at the login prompt. Login using the username/password you set during installation. Readers must note: The Linux slice is not ready yet, cause we have not defined the mount points of the root (/) directory and swap filesystem as well. So, for doing this and more, we proceed to the chapter, [Chapter 6: Installing Red Hat Linux 7.3 \(Valhalla\)](#). Readers must note that if you have already installed FreeBSD, you can skip the next 2 chapters and proceed with the Linux installation directly.
-

[Previous](#)

[Home](#)

[Next](#)

[Previous](#)[Home](#)[Next](#)

3.4 Installing OpenBSD 3.2-RELEASE

This Chapter focuses on installing OpenBSD 3.2-RELEASE and multi-booting the system so that it can coexist successfully with the already existing Windows operating system on the computer. Like the Windows and FreeBSD installation, it too focuses on a CD-ROM based installation. Experts may proceed as usual without help, as for the newbies out there, make sure you check out the *Frequently Asked Questions for OpenBSD* at the "The OpenBSD Project homepage" at <http://www.openbsd.org>. For a step-by-step OpenBSD Installation procedure with screenshots, please refer to the "Installing OpenBSD" Online manual at the OpenBSD Project homepage. These and other documentation available on the OpenBSD 3.2-RELEASE CD-ROM provides exhaustive and extensive coverage of all aspects related to OpenBSD. Make sure you grab as much information as you can from this site before proceeding with this guide.

Since both FreeBSD and OpenBSD operating systems are BSD-derivatives, UNIX-related information present in the /usr/share/doc directory on both the systems provide very high-quality technical information on how to install, use and customize a UNIX-system. If you are not yet a FreeBSD/OpenBSD wizard, my humble and honest advice would be to read the "Unix User's Supplementary Documents (USD)", "Unix System Manager's Manual (SMM)" and the "Unix Programmer's Supplementary Documents (PSD)" on these systems.

Note: I hereby assume the reader executing this Chapter possesses a sound understanding and knowledge of BSD-style hard disk partitioning and labeling schemes, device naming conventions and so on before proceeding any further. If in case you have no idea at all of what I am talking about or in general of OpenBSD, I strongly suggest you to go back to the earlier section and read through the entire contents of the Chapter. Though the material is FreeBSD specific, but most of it apply to OpenBSD as well since OpenBSD like any other BSD out there follows the "slicing" scheme.

A few significant points to remember: In Linux, an entry like /dev/hda1 refers to a partition which in FreeBSD is likely to be labeled /dev/ad0s1. The OpenBSD counterpart would be /dev/wd0a for the first hard disk drive, /dev/wd1a for the second hard disk drive and so on. OpenBSD disklabel allows up to 16 partitions. It can detect and represent all primary partitions that exist on the hard disk. OpenBSD disklabel though sees all partitions within an extended partition but it does not list the extended partition container. Readers must note that OpenBSD reserves the c: partition which represents the entire hard disk, i: through p: are reserved for partitions belonging to other operating systems other than OpenBSD (for example Windows or Linux partitions), with a:, b: and d: - h: available for OpenBSD specific partitions. Thus, an important fact to note is that the word "partition", without qualification, usually refers to a UNIX disk area on which a file system will be mounted.

Summary of Installation Steps

For a complete step-by-step coverage of OpenBSD installation, please consult the "Installing OpenBSD Manual" available on the OpenBSD Project web site. As in the previous Chapter, here also, I would be mentioning only the significant steps for the installation. Use these steps to install OpenBSD on your computer and configure it for "multiple-booting":

1. Right now, we have a computer running Microsoft Windows OS. On my PC, it is Windows 98 (SE). I

- put a bootable OpenBSD 3.2-RELEASE CD-ROM in the CD-ROM drive and reboot the system.
2. The computer boots, reads the CD-ROM, displays the usual hardware probing messages on the screen and presents you with an interactive text-based installation program. Readers must note: There are several ways to install OpenBSD onto a disk. The easiest way, should your computer support it, is to boot off the CD-ROM. Otherwise, you can boot from a 3.5" 1.44MB floppy disk. You should now be ready to install OpenBSD.
 3. Read the questions that appear on the screen and answer them accordingly. Make sure that is the best choice before committing to the installation. If any question has a default answer, it will be displayed in brackets ("[]") after the question. If you wish to stop the installation, you may hit Control-C at any time, but if you do, you will have to begin the installation process again from scratch.
 4. You will next be asked for your terminal type. You should just hit return to select the default (vt220). vt220 refers to "Console" rather than serial/dial-up or network/pseudo terminals.
 5. After entering the terminal type you will be asked whether you wish to do an "(I)nstall" or an "(U)pgrade". Enter 'I' for a fresh install or 'U' to upgrade an existing installation.
 6. You will be presented with a welcome message and asked if you really wish to install (or upgrade). Assuming you answered yes, the install program will then tell you which disks of that type it can install on, and ask you which it should use. Check whether OpenBSD correctly assigns the corresponding device file to your hard disk or not. OpenBSD assigned my first IDE hard disk a label /dev/wd0.
 7. Then, select a [no] and press enter, when OpenBSD asks you whether you want to assign the whole hard disk for installing OpenBSD or not. I select a no here, cause I am preparing a multi-boot system that would run 3 operating systems (one at a time of course!).
 8. Proceed with the installation as usual till you reach the OpenBSD fdisk. If fdisk is being invoked on your behalf, it will start by displaying the current partitions defined and then allow you to modify this information, add new partitions and change which partition to boot from by default. Here, I have to create a slice with "Partition ID#": A6 that would be used later for installing OpenBSD 3.2. Readers must note: FreeBSD partition ID is A5 and is different from OpenBSD partition ID A6. So while creating the slice, the ID should be A6 and not A5.
 9. I had an output like this on my screen after I successfully created my OpenBSD slice.

```

Disk:wd0                geometry: 1274 / 255 / 63          [ 20466810 Sectors ]
Offset:0

```

		Starting			Ending			LBA Info:		
#	Id	C	H	S	C	H	S	Size		
0:	0B	0	1	1	249	254	63	4016187	Win95 FAT32	
*1:	A6	250	1	1	1273	254	63	16450497	OpenBSD	

10. Try understanding the geometry. It represents the actual physical geometry or arrangement of my hard disk. The number of sectors available are given. The 2 sections containing the "Starting" and "Ending" C (cylinder), H (heads) and S (sectors) are also given. I started out with the first slice with ID: 0B only available. The information under LBA Info shows me that it is a Microsoft Windows 95 FAT32 filesystem partition. Hey! that is the one I created when I installed Microsoft Windows 98 (SE) in it.
11. I then create a new slice, with filesystem ID#: A6 to represent an OpenBSD slice, mention the "Starting" and "Ending" cylinders, heads and sectors and finally (very important) set the slice bootable by setting the appropriate flag from the command line. This is shown by the asterisk (*) before the starting of the second slice (#1:) is represented as (*#1:). It tells the MBR code to execute the OpenBSD boot loader cause it will in turn boot my other operating system.
12. We then enter a "**quit**" and not an "exit" on the command line and exit the OpenBSD fdisk partition

editor. Readers must note: Typing "quit" means to exit the editor saving the changes and "exit" means to exit the editor without saving any changes. So, we do a "quit", not an "exit".

13. Next the disk label which defines the layout of the OpenBSD file systems must be set up. The installation script will invoke an interactive editor allowing you to do this. I create a swap of about a few thousand bytes, and provide the rest to the "a" partition i.e. to the "root" directory on the filesystem. The install program will now label your disk and ask which file systems should be created on which partitions. It will automatically select the 'a' partition to be the root file system. Next it will ask for which disk and partition you want a file system created on.
 14. After all your file systems have been created, the install program will give you an opportunity to configure the network. Next, you are prompted to configure your network interfaces, default route and IP address of the primary name server to and the host table.
 15. Once all these done, you will be able to extract the distribution sets onto your system. When installing from a CD-ROM, you will be asked which device holds the distribution sets. This will typically be "cd0". Next you will be asked which partition on the CD-ROM the distribution is to be loaded from. This is normally partition "a".
 16. Next you will have to identify the file system type that has been used to create the distribution on the CD-ROM, this can be either FFS or ISO CD9660. The OpenBSD CD distribution uses the CD9660 format.
 17. You will also have to provide the relative path to the directory on the CD which holds the distribution, for the i386 this is "3.2/i386".
 18. When all the selected distribution sets has been extracted, you will be allowed to select which time zone your system will be using, all the device nodes needed by the installed system will be created for you and the file systems will be unmounted. For this to work properly, it is expected that you have installed at least the "base32", "etc32", and "bsd" distribution sets.
 19. A message will appear on your screen telling you that OpenBSD 3.2 operating system has been successfully installed on your computer. Type in halt to halt the system and reboot.
 20. When I reboot for the first time, I get the following prompt on the screen: **F1 DOS, F2 BSD** each in one line. I press the F1 functional key to boot into Windows 98 (SE) and check whether everything is working perfectly or not. As it so happens, it does! Then I reboot and this time, I press F2, which logs me into OpenBSD 3.2.
 21. Depending on the configuration of your computer, you will receive bootup messages on your screen when you boot into OpenBSD for the first time. Though the messages scroll off the screen rather fast for you to read anything productive, you can always read it later by using the "/sbin/dmesg | more" command. Finally, you will arrive at the login prompt. Login using the username/password you set during installation. Check if everything is in place or not and try exploring the new system.
 22. Readers must note: So that is all with OpenBSD this time. We still have Linux to install and configure. So, for doing this and more, we proceed to the next chapter, [Chapter 6: Installing Red Hat Linux 7.3 \(Valhalla\)](#).
-

[Previous](#)

[Home](#)

[Next](#)

3.5 Installing NetBSD 1.6.1

This Chapter focuses on installing NetBSD 1.6.1 and multi-booting the system so that it can coexist successfully with the already existing Windows operating system on the computer. Like the Windows, FreeBSD and OpenBSD installations, it too focuses on a CD-ROM based installation. Experts may proceed as usual without help, as for the newbies out there, make sure you check out the documentation available on the NetBSD Homepage at <http://www.netbsd.org>. These and other documentation available on the NetBSD 1.6.1 CD-ROM provides exhaustive and extensive coverage of all aspects related to NetBSD. Make sure you grab as much information as you can from this site before proceeding with this guide.

Note: I hereby assume the reader executing this Chapter possesses a sound understanding and knowledge of BSD-style hard disk partitioning and labeling schemes, device naming conventions and so on before proceeding any further. If in case you have no idea at all of what I am talking about or in general of NetBSD, I strongly suggest you to go back to the Chapter covering FreeBSD.

A few significant points to remember: NetBSD labels hard disks similar to OpenBSD with /dev/wd0 for the first hard disk, /dev/wd1 for the second hard disk and so on. NetBSD disklabel allows up to 16 partitions. It can detect and represent all primary partitions that exist on the hard disk. NetBSD disklabel though sees all partitions within an extended partition but it does not list the extended partition container. Readers must note that NetBSD reserves the c: partition which represents the entire hard disk, i: through p: are reserved for partitions belonging to other operating systems other than NetBSD (for example Windows or Linux partitions), with a:, b: and d: – h: available for NetBSD specific partitions.

Summary of Installation Steps

For a complete step-by-step coverage of NetBSD installation, please consult the documentation available on the NetBSD Project web-site. As in the previous Chapter, here also, I would be mentioning only the significant steps for the installation. Use these steps to install NetBSD on your computer and configure it for "multiple-booting":

1. Right now, we have a computer running Microsoft Windows OS. On my PC, it is Windows 98 (SE). I put a bootable NetBSD 1.6.1 CD-ROM in the CD-ROM drive and reboot the system.
2. The computer boots, reads the CD-ROM, displays the usual hardware probing messages on the screen and presents you with an installation program. Readers must note: There are several ways to install NetBSD onto a disk. The easiest way, should your computer support it, is to boot off the CD-ROM. Otherwise, you can boot from a 3.5" 1.44MB floppy disk. You should now be ready to install NetBSD.
3. The blue-colored NetBSD 1.6.1 Installation Tool is called sysinst. At the very beginning, you receive a "Welcome to sysinst" screen. Select the option "Install NetBSD to hard disk" and press enter.

4. This presents you a confirmation screen, select "Yes" and press enter.
5. Your hard disk is detected and NetBSD labels it correspondingly. On my PC, my first IDE HDD gets labeled as wd0. Hit enter to continue.
6. Next appears the "Disk Geometry" screen where your hard disk drive geometry is matched with the BIOS geometry. If sysinst was unable to uniquely match the disk you chose with a disk known to the BIOS, then you have to set the disk geometry by hand. In my case, the disk matches the following BIOS disk:

BIOS# cylinders heads sectors

0 1023 255 63

7. Thus I select, "This is the correct geometry" and press enter. Next comes the screen where you have to make the choice that you would like to multi-boot your system and not dangerously-dedicate your system. Select "Use only part of the disk" instead of "Use the entire disk" and press enter.
8. Proceed to choose the "size unit to use". I chose "Megabytes" as my size specifier.
9. Proceed with the installation as usual till you reach the NetBSD fdisk where you can view the DOS Partition table on your computer. I have a computer system which was running Windows 98 (SE), NetBSD 1.6 and FreeBSD 4.8-RELEASE. The DOS partition table was as follows:

Total disksize 19541 MB.

	Start(MB)	Size(MB)	End(MB)	Kind
0:	0	9774	9774	Windows FAT32, LBA
1:	9773	3001	12775	NetBSD
2:	12774	6766	19540	old NetBSD/FreeBSD/386BSD
3:				unused

10. Edit partitions 1 and 2 by making "Kind" as "unused", then selecting "Partition OK" and pressing enter.
11. Then select unused partition 1, then "Kind", then "NetBSD" and finally "Start and size". Mention the "Start:" value as "-0" and "size:" to the value you would like to set. I chose about 4000 MB. Make sure you select "Set active" and press enter and "Partition OK" finally.
12. The new DOS Partition table on my PC looks like this:

Total disksize 19541 MB.

	Start(MB)	Size(MB)	End(MB)	Kind
0:	0	9774	9774	Windows FAT32, LBA
1:	9773	4001	13774	NetBSD
2:				unused
3:				unused

13. Select "Exit" and press enter.
14. The next screen tells you that you have more than one operating system installed on your computer. When asked to install a bootselector, answer "Yes".

15. This brings you to the "Bootselection menu items" screen. Here, you can change the simple menu entries for matching partition entries that are displayed when the system boots. Also, you can specify the timeout (in seconds) and the default action to be taken (if no selection is made in the bootmenu). By default, the Boot menu timeout is set to 10 seconds, which I change to 100 seconds. The default boot menu action is set to "Boot off partition 0".
16. Then edit menu entries 0 and 1, to label "Windows" and "NetBSD" respectively. Then exit.
17. Next appears the NetBSD Disklabel screen. Here you are expected to set up your NetBSD disklabel. Four options are mentioned: Standard, Standard with X, Custom and Use existing.
18. I choose the installation type: "Standard with X" which requires about 450.00 MB HDD space.
19. Next screen exhibits the BSD-disklabel partitions. Since for most people in most situations this suits the best, keep the changes as it is and select "Partitions are OK" and press enter
20. A name is prompted for your NetBSD disk, I entered "ghosh" and continue.
21. /sbin/newfs runs to create the new filesystems.
22. In the next screen you are prompted to install either normal set of bootblocks or serial bootblocks. Since we are using the BIOS console device as the console, you must select "Use normal bootblocks" and press enter.
23. This completes the first part of the NetBSD installation procedure. Sysinst has written a disklabel to the target disk, and newfs'ed and fsck'ed the new partition on the disk. The next step is to fetch and unpack the distribution filesystems.
24. I chose "Custom Installation" of NetBSD distribution sets.
25. In the next screen, choose all that you would like to install and press "Exit".
26. Enter the type of installing media. Select "cdrom" and device:cd0 and directory:/i386/binary/sets and continue.
27. The following .tgz files gets untarred and installed onto your hard disk: base.tgz, etc.tgz, comp.tgz, games.tgz, man.tgz, misc.tgz, text.tgz, xbase.tgz, xfont.tgz, xserver.tgz, xcomp.tgz and xcontrib.tgz.
28. A message appears as "All selected distribution sets unpacked successfully". Next, device files are made automatically, and you are prompted for a few necessary post-installation configuration steps.
29. You need to set timezone, I choose "Europe/Moscow".
30. Next, to choose the password cipher. The options available are: DES or MD5. I prefer MD5 to DES.
31. I set the root password and then finally a message appears on your screen telling you that NetBSD 1.6.1 operating system has been successfully installed on your computer. Reboot your computer to boot from hard disk.
32. When I reboot for the first time, I get the following prompt on the screen: **F1 Windows, F2 NetBSD** each in one line. I press the F1 functional key to boot into Windows 98 (SE) and check whether everything is working perfectly or not. As it so happens, it does! Then I reboot and this time, I press F2, which logs me into NetBSD 1.6.1.
33. Depending on the configuration of your computer, you will receive bootup messages on your screen when you boot into NetBSD for the first time. Though the messages scroll off the screen rather fast for you to read anything productive, you can always read it later by using the "/sbin/dmesg | more" command. Finally, you will arrive at the login prompt. Login using the username/password you set during installation. Check if everything is in place or not and try exploring the new system.

34. Readers must note: We still have Linux to install and configure. So, for doing this and more, we proceed to the next chapter, [Chapter 6: Installing Red Hat Linux 7.3 \(Valhalla\)](#).
-

[Previous](#)

[Home](#)

[Next](#)

3.6 Installing Red Hat Linux 7.3 (Valhalla)

This Chapter focuses on installing Red Hat Linux 7.3 and configuring it for booting with the already existing operating systems on the computer. It focuses on a CD-ROM based installation, so it is ideal for users (both new and old) who want a quick and simple installation solution. If you are an experienced Linux user, just shoot ahead of this Chapter and install Red Hat Linux on your system. You may skip all the contents of this Chapter if you have enough practical experience. But in case you are a complete Linux newbie, or you just decided to taste Linux for fun, then you may want to gather more information on how to install Linux with another already existing OS on the computer. Red Hat offers excellent Installation tools while installing but while multi-booting your computer with 3 operating systems, it is always advisable to use the "Custom" installation which makes the story for the Linux newbies a little difficult. Using Linux fdisk or Disk Druid is easy and if you are using the Official boxed-set version from Red Hat, make sure you use the Official Red Hat Linux x86 Installation Guide enclosed with the package. Otherwise, you can always refer to the Distribution-specific Installation Manuals at the Official Red Hat web-site. Red Hat's official web-site is located at <http://www.redhat.com>. Make sure you refer to the Red Hat Frequently Asked Questions for answers to questions and problems that may occur before, during or after the installation. You will find the FAQ online at: http://www.redhat.com/support/resources/faqs/rhl_general_faq/s1-contact.html.

Red Hat Linux has some exciting and important installation features. Large improvements were made in the install process as version 6 arrived. Now in the latest versions 7.3 (Valhalla) and 8.0 (Psyche), Red Hat Linux's installation is very smooth with Anaconda doing most of the dirty jobs for you. Some of the important installation features include "Kudzu" and the "Kernel 2.4". The contents of this Chapter may be categorized in the following sections mentioned below:

1. **Disk drive and partition naming in Linux**
 2. **Filesystems in Linux**
 3. **Summary of Installation Steps**
-

1. Disk drive and partition naming in Linux

Before shooting ahead with the installation of Red Hat Linux, let us take a brief tour of the disk drive and partition naming conventions in Linux. Worthwhile mentioning, I discussed in detail the drive and partition naming convention in FreeBSD earlier when I talked about installing FreeBSD.

Linux uses a special way of naming hard drives and the partitions that you make within them. In UNIX (always remember, the Linux kernel is the reimplement of UNIX only!) terminology, UNIX systems (and more specifically the Kernel) cannot read and understand "devices". To the UNIX kernel, everything is abstracted as a "file containing ASCII characters". That means a sentence like "I have a 30.0 GB C: drive" in Linux does not make

any sense at all. So, let us see how our hard drives and partitions are named under Linux.

- First IDE hard disk drive (Primary Master) – /dev/hda
- Second IDE hard disk drive (Primary Slave) – /dev/hdb
- Third IDE hard disk drive (Secondary Master) – /dev/hdc
- Fourth IDE hard disk drive (Secondary Slave) – /dev/hdd

where, /dev is the directory under the root ("/") directory in Linux which contains all the device files associated with devices.

What about SCSI drives? Well, Linux labels that too!

- First SCSI hard disk drive (Primary Master) – /dev/sda
- Second SCSI hard disk drive (Primary Slave) – /dev/sdb
- Third SCSI hard disk drive (Secondary Master) – /dev/sdc
- Fourth SCSI hard disk drive (Secondary Slave) – /dev/sdd

where, /dev is the directory under the root ("/") directory in Linux which contains all the device files associated with devices.

Now, let us talk about the partitioning scheme under Linux. In Linux, we first consider a particular hard disk, whether an IDE or a SCSI hard disk. Then, we consider partitions on it. For example, the partitions on an IDE drive are named in the following way (/dev/hda is used as an example):

- First primary partition – /dev/hda1
- Second primary partition – /dev/hda2
- Third primary partition – /dev/hda3
- Fourth primary partition – /dev/hda4

What about if we have an "extended partition" under Linux which contains a few logical disk drives in it? Well, the Linux kernel maps those partitions too to the corresponding "block" device files stored in the /dev directory. Let us consider that the /dev/hda in our above example, contains 1 primary partition and 1 extended partition, which in turn has 5 logical disk drives in it. So, in this case, how would the partition table look like and mean? Let us see how Linux labels this scheme. Linux would represent this as:

- /dev/hda
 - /dev/hda1
 - /dev/hda2
 - /dev/hda5
 - /dev/hda6
 - /dev/hda7
 - /dev/hda8
 - /dev/hda9

where, /dev/hda1 is the only primary partition and /dev/hda2 is the only extended partition on the hard disk under scrutiny. Note that Linux labels primary partitions from /dev/hda1 onwards to /dev/hda4, with logical disk drives extending from 5 till 16. Thus, in the above example, hda5 to hda9 are the 5 logical disk drives existing on the extended partition. In fact recently on one of my friend's computer, I partitioned the hard disk drive and installed Microsoft Windows Millennium (ME), Microsoft Windows XP Professional and Red Hat Linux 8.0 (Psyche). Linux fdisk when executed, shows exactly the same partition table as given above. He uses one IDE hard disk drive and has 1 primary partition labeled as C: drive under Windows which contains WinXP (FAT32 filesystem), with hda5 and hda6 labeled as

D: and E: drives under Windows (where hda5 actually contains Windows ME edition (FAT32 filesystem)) with hda7, hda8 and hda9 used for Linux. hda7 is the /boot partition, hda8 is the "root" (/) filesystem and hda9 is the "swap" filesystem. This was a simple example of how the partition table looks like in Linux. If you have quite a few partitions and plan to install a few operating systems, be ready to see a more complex partition table when you execute a command like `"/sbin/fdisk /dev/hda"` (with /dev/hda being the Primary Master on your computer).

For example, when I executed the Linux **fdisk** on my friend's computer, it gave an output similar to this:

Command (m for help): **p**

Disk /dev/hda: 255 heads, 63 sectors, 5005 cylinders
Units = cylinders of 16065 * 512 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/hdc1	*	1	1275	10241406	c	Win95 FAT32 (LBA)
/dev/hdc2		1276	5005	29961225	f	Win95 Ext'd (LBA)
/dev/hdc5		1276	2550	10241406	b	Win95 FAT32
/dev/hdc6		2551	3825	10241406	b	Win95 FAT32
/dev/hdc7		3826	3831	48163+	83	Linux
/dev/hdc8		3832	4213	3068383+	83	Linux
/dev/hdc9		4214	4246	265041	82	Linux swap

2. Filesystems in Linux

After our brief discussion on the drive and partition labeling schemes in Linux, let us talk a bit about the Linux filesystem. Microsoft Windows OSes use either FAT16, FAT32, NTFS 4.0 or the most recently released NTFS 5.0 filesystems. The GNU/Linux Kernel originally used the "First Extended filesystem", commonly denoted as ext or extfs. It has now been superseded by ext2 and ext3 filesystems. Kernel 2.4, currently uses Linux's native "*Third Extended filesystem*", commonly denoted as "ext3" or "ext3fs". Though Kernels 2.2.x and 2.4.x support ext2fs and ext3fs, but the latest stable release i.e. Kernel 2.4.20 uses the "ext3fs" as the default filesystem. ext3fs is an extension to ext2fs functionality as is NTFS 5.0 to NTFS 4.0 in Windows. The significant difference between the "ext2fs" and "ext3fs" filesystems is that ext3fs is a "Journaling filesystem" while ext2fs is not. We will discuss these filesystems in detail later when we compare them with the Unix Filesystem (UFS) which is the native filesystem under FreeBSD.

The "*Linux Programmer's Manual*" has a section called "FILESYSTEMS(5)". It explains which filesystem types Linux supports. One can access all the details by simple typing in `"man fs"` at the command prompt on a Linux workstation. It includes the following filesystems: minix, ext, ext2, ext3, xia, msdos, umsdos, vfat, proc, nfs, iso9660, hpfs, sysv, smb, ncpfs. When, as is customary, the filesystem is mounted on /proc, and you can find in the file /proc/filesystems which filesystems your kernel currently supports. If you need a currently unsupported one, either you can insert the corresponding module or just recompile the kernel to create a new one. Then, just install the new custom kernel to access the

filesystem of your choice. In order to use a filesystem, you have to mount it.

Below is a short description of a few of the available filesystems. **minix** is the filesystem used in the Minix operating system, the first to run under Linux. It has a number of shortcomings: a 64MB partition size limit, short filenames, a single time stamp, etc. It remains useful for floppies and RAM disks. The **ext** is an elaborate extension of the minix filesystem. It has been completely superseded by the second version of the extended filesystem (ext2) and has been removed from the kernel (in 2.1.21). The **ext2fs** is the high performance disk filesystem used by Linux for fixed disks as well as removable media. The second extended filesystem was designed as an extension of the extended file system (ext). ext2 offers the best performance (in terms of speed and CPU usage) of the filesystems supported under Linux. The **ext3** is a journaling version of the ext2 filesystem. It is easy to switch back and forth between ext2 and ext3. The **xia** was designed and implemented to be a stable, safe filesystem by extending the Minix filesystem code. It provides the basic most requested features without undue complexity. The xia filesystem is no longer actively developed or maintained. It was removed from the kernel in 2.1.21. **msdos** is the filesystem used by DOS, Windows and some OS/2 computers. msdos filenames can be no longer than 8 characters, followed by an optional period and 3 character extension. The **umsdos** is an extended DOS filesystem used by Linux. It adds capability for long filenames, UID/GID, POSIX permissions, and special files (devices, named pipes, etc.) under the DOS filesystem, without sacrificing compatibility with DOS.

The **vfat** is an extended DOS filesystem used by Microsoft Windows 95 and Windows NT. VFAT adds the capability to use long filenames under the MSDOS filesystem. The **proc** filesystem is not a general filesystem. It is a "pseudo-filesystem" which is used as an interface to kernel data structures rather than reading and interpreting /dev/kmem. In particular, its files do not take disk space. The **iso9660** is a CD-ROM filesystem type conforming to the ISO 9660 standard. Linux supports **High Sierra**, the precursor to the ISO 9660 standard for CD-ROM filesystems. It is automatically recognized within the iso9660 filesystem support under Linux. Linux also supports the System Use Sharing Protocol records specified by the Rock Ridge Interchange Protocol. They are used to further describe the files in the iso9660 filesystem to a UNIX host, and provide information such as long filenames, UID/GID, POSIX permissions, and devices. It is automatically recognized within the iso9660 filesystem support under Linux. The **hpf** is the High Performance Filesystem, used in OS/2. This filesystem is read-only under Linux due to the lack of available documentation. **sysv** is an implementation of the SystemV/Coherent filesystem for Linux. It implements all of Xenix FS, SystemV/386 FS, and Coherent FS. The **nfs** is the network filesystem used to access disks located on remote computers. **smb** is a network filesystem that supports the SMB protocol, used by Windows for Workgroups, Windows NT, and Lan Manager. To use smbfs, you need a special mount program, which can be found in the ksmbfs package. The **ncpfs** is a Network filesystem that supports the NCP protocol, used by Novell NetWare.

3. Summary of Installation Steps

Now that we have covered quite a lot on partitioning schemes and filesystems under Linux, it is time to move on to the actual Linux installation. For a complete step-by-step coverage of Red Hat Linux installation, please consult the distribution-specific installation guides on Red Hat's official web-site. They cover all types of installation with appropriate screenshots and tips and tricks for all the necessary steps of the Linux installation. Here, I would only mention the significant steps for the installation. Use these steps to install Red Hat Linux on your

computer:

1. Place your Red Hat Linux 7.3 bootable CD-ROM on your CD-ROM drive and then restart the computer. I have assumed here that the reader has a Red Hat Linux 7.3 distribution release CD. If you have older distributions, that is no problem at all. You can always use them. The basic installation steps are the same for all the distribution releases. Make sure that your computer is bootable from the CD-ROM in the BIOS setup. If you cannot boot from the CD-ROM drive, alternative boot methods are also available. Refer to the Red Hat Installation Guides for more information on making "local", "network" and "PCMCIA" boot disks.
2. After a short delay, the Red Hat Linux 7.3 screen containing the boot: prompt should appear. The screen contains information on a variety of boot options. Either press "Enter" for entering a GUI-based installation setup or type in "**linux text**" at the **boot:** prompt for entering a ncurses-based text-based installation type. I always prefer text-based installations. Watch the boot messages to see whether the Linux kernel detects your hardware. If it does not properly detect your hardware, you may need to restart the installation in expert mode. Enter the "expert" mode using the following boot command: boot: **linux expert** and press enter.
3. Proceed as usual, configuring language, keyboard and the mouse. Then appears the "Welcome to Red Hat Linux" screen. Proceed forward, till you reach the "Install" or "Upgrade" screen. Since we are installing Linux for the very first time, we select "Install". Now the question that arises here is: *Which installation type to consider and why?* Red Hat Linux 7.3 provides 4 options: Desktop, Workstation, Server and Custom. We do "not" consider the Desktop, Workstation and Server setups owing to obvious reasons. Because they do not provide the user with custom features and functionality. Thus, we proceed with the "Custom" installation.
4. Proceed as usual and at the partitioning screen, you would be asked to choose between "fdisk" and "disk druid". For Linux experts, fdisk is the obvious tool of choice. Disk druid is Red Hat's graphical tool for viewing, editing, adding and deleting partitions on hard disk drives. Choose as you wish. I chose "disk druid" for this guide. At the next screen, you would be shown the entire hard disk drive geometry. Make an effort to understand what is being shown on the screen and try to analyze it with the information of this chapter above where I have had discussed about partitioning in detail. Try mapping the partitions being shown to Windows terminology.
5. On my system, Disk Druid represents a /dev/hda as /dev/hda1, /dev/hda2 and a /dev/hda3 partitions initially. /dev/hda1 is the FAT32 partition which (on my computer) contains the Microsoft Windows 98 (SE), /dev/hda2 is the FreeBSD (or OpenBSD) slice and /dev/hda3 is the Linux partition. Readers must note: The /dev/hda3 partition is the slice I created in FreeBSD disk label editor by assigning 131 to the underlying filesystem.
6. I keep /dev/hda1 and /dev/hda2 as it is, and delete /dev/hda3. As a result, I get some free space. Then I add a **/boot** of about 100MB as ext2fs, then a **swap** of about 128MB and finally a root ("/") by assigning to it the remaining space on the drive. That is all what needs to be done. Thus, we have something like this:

Command (m for help): **p**

Disk /dev/hda: 255 heads, 63 sectors, 2491 cylinders
Units = cylinders of 16065 * 512 bytes

Windows+BSD+Linux Installation Guide, v1.1

Device	Boot	Start	End	Blocks	Id	System
/dev/hda1		1	125	1004031	b	Win95 FAT32
/dev/hda2 *		126	1399	10233405	a5	FreeBSD
/dev/hda3		1400	1412	104422+	83	Linux
/dev/hda4		1413	2491	8667067+	f	Win95 Ext'd (LBA)
/dev/hda5		1413	1428	128488+	82	Linux swap
/dev/hda6		1429	2491	8538516	83	Linux

As shown above, /dev/hda3 is mounted as /boot, /dev/hda5 is the Linux swap filesystem and /dev/hda6 is mounted as /.

7. Proceed with the installation as usual, till you reach the screen prompting you to choose the boot loader. This is a very important step in this installation, in fact, the most important. A **boot loader** is literally the first software program that runs when you switch on your computer. The Master Boot Record (MBR) is a special area on your hard drive that is automatically loaded by your computer's BIOS. It is the earliest point at which the boot loader (whether LILO or GRUB) can take control of the entire boot process. GRUB, when runs, reads instructions passed to it from its own configuration file (for example, the configuration file for the Linux Loader is the /etc/lilo.conf and for GRUB is the /boot/grub/grub.conf), passes significant instructions to the BIOS, and then loads the appropriate operating system boot files into the computer's memory. This, in turn enables the correct filesystems to be mounted at the correct mount points, which in turn then prepares the computer for later use. Thus, the boot loader is vital for proper booting of the operating system.
8. You can use any one of the 2 available boot loaders in Linux. Linux Loader (LILO) or the GNU Grand Unified Boot loader (GRUB). The reader is free to choose whichever he/she likes. I prefer using GNU GRUB for the reasons outlined in the FAQ. Readers must note that I am omitting the design, implementation and configuration details of GRUB in this guide as they exist beyond the scope of this guide. A lot of information is available out there on the Internet on these and other "free" as well as "proprietary" boot loaders. Check them out if it is required.

GRUB is a very powerful x86-based boot loader which can be used to boot a number of operating systems. GRUB supports directly all GNU/Linux distributions, FreeBSD, NetBSD, OpenBSD, Sun Solaris and others. It supports booting other proprietary operating systems as well using the "chain loading" technology. Whether you have Microsoft Windows 95/98 (SE)/Millennium Edition/NT/2K/XP as the first initially installed OS on your computer and then FreeBSD 4.8-RELEASE or OpenBSD 3.2-RELEASE; simple lay back and relax because GRUB is there with all its power and dynamic functionality to take care of things. If you have Microsoft Windows 95/98 (SE)/Millennium Edition which do not have boot loaders by default (unlike Microsoft Windows NT/2K/XP), then "Add GRUB to the MBR".

9. After startup, you will receive the GRUB screen prompting you to select the OS. If in case you have Microsoft Windows NT/2K/XP, which uses by default the "NT Boot loader" for booting the operating system software, then also go ahead and "Add GRUB to the MBR". Do not panic! It would cause no problems at all. Only in this case, after successful startup, you will receive the GRUB screen prompting you to select the OS. You will have 2 entries in the list, for example, I have "Windows" and "Linux". Selecting "Linux" and pressing enter loads the Red Hat Linux boot files to the computer's memory and the computer bootstraps itself into Linux. Whereas, on

the other hand, selecting "Windows" and pressing enter utilizes the "chain loading" mechanism, and invokes the "FreeBSD boot loader". The FreeBSD boot loader then loads Microsoft Windows 98 (SE) into the computer's memory.

10. That explains pretty much everything. After adding GRUB to your Master Boot Record (MBR), proceed with the installation as usual. The reader must note that he/she may install GRUB in one of two places, either the MBR or in the boot sector of the active partition. If you plan to use another boot loader on your system (for example, OS/2's Boot Manager, or Windows NT's NT Boot loader, or any commercial boot loaders like System Commander and Partition Magic), add GRUB to the "first sector of your root partition" and then configure the other boot loader to start GRUB (which will then boot Red Hat Linux).
11. Once you have created the partitions and installed GRUB in the MBR, all that is left is to proceed with the installation as usual, selecting packages for installation as you wish (*you can later install them as well using the RPM command from the command line*) and so on. Finally, finish with the Red Hat Linux installation. When prompted to create a "bootdisk", make sure you create one. It comes handy during system crashes. You can always create a "setup bootdisk" later using the `/sbin/mkbootdisk` command from the command line interface (CLI) mode.
12. Depending on the number of packages that you had chosen earlier as well as your computer's resources, it will take some time for the installation to complete. Sit back and relax while the packages are being installed. Or run to your nearby McDonalds to grab a burger or 2. For the experts out there, after reading through all this, if you do feel that something was wrongly stated/explained, make sure you drop me a line or two. For the absolute newbies, I have a simple advice: Practice, practice and practice till you get perfect. There is absolutely no shortcut to success. It is not possible at all.
13. After all packages have been successfully installed and configured, and your Red Hat Linux Installation is complete, the Linux CD-ROM would be ejected from the CD-ROM drive, and the computer would automatically reboot. As mentioned above, after successful startup, you will receive the GRUB screen prompting you to select the OS for booting. Depending on your configuration during the installation, you will have 2 entries in the list, for example, I have "Windows" and "Linux". Check whether both OSes are working properly or not by booting into each of them one after another. I am 100% sure that everything would work just fine.
14. If everything works fine, well, it is time for rejoicing. If in case, something goes wrong somewhere, then depending on what exactly went wrong where, many things could be done. I sincerely request the reader to note everything if in case something goes wrong someplace. Write down on paper a short report if required. This helps a lot in later diagnosis. Now, it is time for configuring FreeBSD within the Red Hat Linux environment which would enable GRUB to boot FreeBSD too via "chain loading" mechanism.
15. Log into Red Hat Linux, and open the `/boot/grub/grub.conf` file using the vi editor. Be very cautious while editing it. Since, GRUB reads the `/boot/grub/grub.conf` file while bootstrapping the computer, typos inside `/boot/grub/grub.conf` file can result in total disasters. Readers must note: GRUB does not have any utility like LILO uses the `/sbin/lilo -v -v` command from the command prompt for checking the `/etc/lilo.conf`. So be extra careful while using GRUB to boot OSes. Now add an entry for FreeBSD before the "Windows" entry. So, your `/boot/grub/grub.conf` file after editing should resemble this:

```
default=0
timeout=150
```

```
splashimage=(hd0,2)/grub/splash.xpm.gz
```

```
title Linux (2.4.18-3)
    root (hd0,2)
    kernel /vmlinuz-2.4.18-3 ro root=/dev/hda6
```

```
title FreeBSD
    root (hd0,a)
    kernel /boot/loader ro root=/dev/hda2
```

```
title Windows
    rootnoverify (hd0,0)
    chainloader +1
```

16. As shown above, the first block is for booting Linux, the second for FreeBSD and the last one is for booting Windows 98 (SE). Readers must note the entry following the word "root" in the 2nd line of the first and second blocks. It defines the exact location on the computer's hard disk where the boot files required for booting the operating systems are located. On the Linux system, we can see that (hd0,2) refers to the 3rd partition which is /dev/hda3. And as we already know, /dev/hda3 refers to the /boot partition which contains the boot files. Similarly, on the FreeBSD system, we can see that (hd0,a) refers to the /dev/ad0s2a which refers to the root(/) filesystem and which on the Linux system is mapped into /dev/hda2. The rootnoverify mentioned for the Windows block tells the Linux Kernel not to mount the /dev/hda1 partition since it is not a Linux filesystem.
17. Now when you reboot, you have a multi-boot system enabling you to boot into any one of three operating systems: Windows 98 (SE) (on my computer), FreeBSD 4.8-RELEASE or OpenBSD 3.2-RELEASE or NetBSD 1.6.1 and Red Hat Linux 7.3. That is all what really needs to be done for preparing a fully featured multi-boot system. If you would like to know how to mount different filesystems on varying operating systems, then proceed to the next chapter, [Chapter 7: Mounting Filesystems](#).

[Previous](#)

[Home](#)

[Next](#)

3.7 Mounting Filesystems

Your computer now has a Microsoft Windows operating system, either FreeBSD 4.8-RELEASE or OpenBSD 3.2-RELEASE or NetBSD 1.6.1 and Red Hat Linux 7.3 installed and coexisting peacefully together. Is that all? Is that enough? I am sure, no! One of the most important reasons that I have Microsoft Windows 98(SE) + FreeBSD 4.8-RELEASE + Red Hat Linux 7.3 all running on my computer (on which I prepared a part of this guide you are reading right now) is because I spend most of my time doing things on FreeBSD and on Linux, but I need access to Windows owing to my projects at the university. So, I need to access data stored on my Windows partition from time to time. But every time when I am working on FreeBSD and I need to access some data on my Windows partition, I just cannot afford to reboot my computer, log into Windows, copy the files on a floppy and then logout, reboot once again, log into FreeBSD, and then copy the contents of the floppy on to my /home/ghosh directory on the FreeBSD system. A process like this is not cute, neither logical nor feasible. Thanks to the "mount" command on FreeBSD and Linux systems, I do not have to go through this "not so cute, neither logical nor feasible" process every time I need access to some data on my Windows partition occasionally.

The mount command is definitely one of the most important commands available to the superuser (or root) account on a *nix system. It allows the root user of a *nix system to mount filesystems (for regular users to access and use) which may or may not be physically located on the same computer. That is, in other words, the mount command allows you to mount both a local filesystem (one that is present physically on the same computer) or a remote filesystem using the appropriate path for accessing the remote node. The mount command first appeared in Version 1 ATTUNIX. See the manual page for the mount command on your FreeBSD or OpenBSD or Linux system for an in-depth analysis of all possible options that can be passed to the mount command.

Though it is a very powerful command on a Unix or Unix-like system, the mount command itself is very simple and easy to use. The syntax for the mount command is: **mount -t filesystem device_name mount_point**. It means you are instructing the mount command to consider the device file named device_name which is of the filesystem type specified by the switch -t filesystem, and then graft it onto the existing FreeBSD/OpenBSD/Linux system at the directory named as the mount point. As cute as that!

This chapter is divided into the following sections:

1. **Mounting filesystems in FreeBSD**
2. **Mounting filesystems in Linux**

1. Mounting filesystems in FreeBSD

For mounting Linux Second Extended filesystem (ext2fs) on a FreeBSD system, I have to build and install a custom FreeBSD Kernel. Building and installing a custom kernel in

FreeBSD is easy. I log in as "root" and execute these following steps to build one:

1. I execute this from the command line: **cd /usr/src/sys/i386/conf**. If there is not a /usr/src/sys directory on your system, then the kernel source has not been installed. The easiest way to do this is by running /stand/sysinstall as root, choosing Configure, then Distributions, then src, and then sys.
2. I create a copy of the original Kernel configuration file provided there as a sample by executing: **cp GENERIC GHOSH**. Here, I have named the would-be kernel GHOSH. Feel free to name it otherwise.
3. I open the GHOSH text file using ee or vi editor. Though the general format of the configuration file is quite simple, full explanation of all the possible kernel options is beyond the scope of this guide. Please refer to the FreeBSD Handbook for more details. Add the following line: '**options EXT2FS**' which provides support for identifying ext2fs Linux native filesystem. Tinker with the other options if you feel like or simply leave them alone.
4. Save it and exit. When I am done, I type the following to compile and install the new kernel: **/usr/sbin/config GHOSH**.
5. Then execute: **cd ../../compile/GHOSH**
6. Execute: "**make depend**", followed by "**make**" and finally finish-off by executing "**make install**".
7. The new kernel will be copied to the root directory as /kernel and the old kernel will be moved to /kernel.old.. Now, reboot your system to use the new kernel.

Once this done, I try mounting a Linux partition using the command: **mount -t ext2fs /dev/ad0s3 /mnt/linux**; assuming /dev/ad0s3 is the FreeBSD slice representing a Linux filesystem and /mnt/linux directory exists. The filesystem should mount at the specified directory acting as the mount point.

For mounting MSDOS partitions as well as Windows 95/98(SE)/ME formatted FAT16 and/or FAT32 filesystems, I execute the following command: **mount -t msdos /dev/ad0s1 /mnt/windows** where, /dev/ad0s1 is the FreeBSD slice that represents the Windows 98(SE) partition on my computer and /mnt/windows directory exists. For unmounting, I execute: **umount /mnt/windows** simply. For mounting MSDOS-based floppies, I execute: **mount -t msdos /dev/fd0 /mnt/floppy**. And for unmounting, I execute: **umount /mnt/floppy** simply. For people who have NTFS filesystem, they can try mounting using the command: **mount -t ntfs /dev/fd0 /mnt/floppy** and unmount using: **umount -t ntfs /dev/fd0 /mnt/floppy**. Readers must note: There is limited writing support for NTFS filesystems.

2. Mounting filesystems in Linux

Mounting a FreeBSD filesystem i.e. UFS (Unix FileSystem) under Linux requires building and installing a custom Linux kernel. Building a Linux kernel is an easy job, and you may refer to any documentation you have access to. Here, I am just specifying the main steps. For building a custom Linux kernel, I log in as root and execute the following steps:

1. I execute: **cd /usr/src/linux-2.4**. If the /usr/src/linux-2.4 directory does not exist on your system, then the kernel-source package has not been installed. For kernel compilation, you need to have kernel-source, make, dev86, gcc and a few dependencies installed.
2. I make a backup (copy) of the original Makefile by executing: **cp Makefile Makefile.bak**. You can always refer to the original Makefile by accessing it as

Makefile.bak.

3. Then, I open the Makefile using vi or any text editor. You may change the **EXTRAVERSION** value to anything of your choice. For example, on my system, the initial Linux Kernel was 2.4.18-3. Since then I built numerous others as 2.4.18-3ghosh, 2.4.18-3smp, 2.4.18-3_dummy, 2.4.18-3bsd and so on. Thus, the EXTRAVERSION code helps to distinguish between a large number of available Linux Kernels on a system.
4. Once done, I save and exit the Makefile. Then, execute: **make mrproper**. It cleans the compilation folders of any junk that was created while compiling Linux kernels on the same system earlier. Readers must note that all the commands executed (including this one) should be executed from within the /usr/src/linux-2.4 directory.
5. I then execute: **make menuconfig**. Readers may also use **xconfig** concerned they have access to X11 or the X Window System.
6. Once inside menuconfig, tinker with the options available. Make sure you enter the "File systems" section, then select the "UFS file system support (read-only)" and "UFS file system write support (DANGEROUS)" options. Thus, you should have something like this: <*> UFS file system support (read-only) and [*] UFS file system write support (DANGEROUS). The config Kernel-level options are as: **CONFIG_UFS_FS** and **CONFIG_UFS_FS_WRITE** for the read-only and write support respectively. Also, enter the "Partition types" section and select the [*] BSD disklabel (FreeBSD partition tables) option i.e. the **CONFIG_BSD_DISKLABEL** option. This allows you to access and read the FreeBSD disk labels in the FreeBSD slices.
7. With these three options enabled, I save the configuration file and exit. Now execute: **make dep && clean**.
8. Then execute: **make bzImage && modules** for creating the zipped (compressed) Linux kernel image file and the corresponding modules.
9. Then execute: **make modules_install** for installing the newly created modules into their own corresponding directories.
10. After this, I copy the bzImage file to the /boot directory. If required, you can also create an initial RAM-disk file image using the mkinitrd command.
11. Once this done, I open the /boot/grub/grub.conf file using the vi editor and make a corresponding entry into it. Save and exit. Reboot to check whether the new Kernel is properly working or not. I am sure it would!

Once the newly built custom Linux kernel is installed, you can start using it to mount UFS filesystems. Use this command: **mount -t ufs /dev/hda2 /mnt/bsd**; concerned /dev/hda2 is the FreeBSD slice and /mnt/bsd exists. Unmount using the command: **umount /mnt/bsd** simply.

For mounting Windows 95/98/ME FAT32 filesystems on Linux, use the command: **mount -t vfat /dev/hda1 /mnt/windows**. The /mnt/windows directory must exist on the system. Unmount by executing: **umount /mnt/windows** simply. For readers who are using the NTFS filesystem, there's bad news! They have to include the NTFS filesystem support in the configuration file and build a custom Linux kernel. Readers must note: NTFS writing support is DANGEROUS and limited.

[Previous](#)

[Home](#)

[Next](#)

[Previous](#)[Home](#)[Next](#)

4. Troubleshooting

Though FreeBSD, OpenBSD and GNU/Linux operating systems are very fine pieces of software, yet when dual or multi-booting PCs with all (or combinations) of them, one can often run into problems. It may range from easy ones to detect and troubleshoot to really weird ones. Troubleshooting, as known (and appreciated) by all UNIX-veterans is not an exact art. In this Chapter, I have listed some problems that I have had (and continue facing) while dual and multi-booting systems with varying operating systems and hardware.

Readers must note that depending on the particular hardware and distributions/releases used, they may experience slightly different versions of the problems listed below. If you encounter a new problem altogether or can contribute to this *Guide* by providing a better solution to one of the problems listed below, make sure you drop me a line at:

subhasish_ghosh@linuxwaves.com.

My personal experience with multi-booting systems saw some weird kinds of failure and often erratic behavior, but mostly from Linux's fdisk and installation procedure. I often made notes of them for diagnosing and troubleshooting later. As I can conclude, amongst Linux distributions (which includes Red Hat, Slackware, SuSE and Mandrake), OpenBSD and FreeBSD releases, FreeBSD's fdisk, disklabel and overall installation procedure (/stand/sysinstall) is the best of them all with absolutely no errors and abrupt crashes. Initially, I always thought that there must be some exact way of installing all these operating systems or an exact order that needs to be followed, but with passing years, I have learnt the truth: Concerned the hardware support is fine, installation CD-ROMs okay, the erratic behavior of installation programs is due to unerased data from previous installs. So before committing yourself to creating a multi-booting system, make sure you start installation on a freshly formatted hard disk. Read on to follow the details of the procedure! Some of the common questions from readers include:

Q1. I created an extended partition with logical disk drives or sub partitions within it. But when I installed FreeBSD, it completely ignored the inner partitions within that partition. Is that normal?

Ans1: Yes, it is absolutely normal. FreeBSD operating system software can only detect and represent primary hard disk partitions which it calls "slices". An extended partition, though would be represented in FreeBSD fdisk will not represent the constituents of the container.

Q2. I installed FreeBSD and OpenBSD operating systems on my PC successfully. When I tried installing Red Hat Linux, Linux's fdisk displayed a "too many partitions" error message. What does it mean? What can I do now? or,

Q3. I deleted earlier installs of FreeBSD, OpenBSD and/or NetBSD operating systems on my PC. When I tried installing Red Hat Linux, Linux's fdisk displayed a "too many partitions" error message. What am I supposed to do now?

Ans2 and Ans3: As the readers can see, this error message flagged by Red Hat Linux fdisk tool can occur under a variety of conditions. I myself encountered both of them while multi-booting PCs. When this message gets displayed Linux's fdisk cannot make any changes to the disk. Personally speaking, I strongly feel that this message: "too many partitions (16, maximum is 8)" usually appears whenever you have unerased data from previous installs of FreeBSD and/or OpenBSD and/or NetBSD operating systems. This usually happens whenever those data partitions were deleted, but the Master Partition Table (MPT) contained in the Master Boot Record (MBR) was not overwritten. Since this problem does not seem to occur on clean hard disks, I conclude that it only appears if you have old BSD disk label data on your hard disk.

Once this error gets flagged, there is nothing much you can do about it because whatever you do or whatever you type in, nothing gets written to disk. If it so happens that you are performing a clean install and you get this error, do a "fdisk /mbr" from the MS-DOS prompt, and try installing Red Hat Linux once again. If it so happens that you have FreeBSD/OpenBSD/NetBSD installed on your system already, then (sorry dude!) erase all earlier installs, wipe the disk clean using "dd" utility or whatever you have at your disposal and start installing everything all over again.

Q4. I was performing a new install of Red Hat Linux on my PC which earlier had multiple installs of FreeBSD and/or OpenBSD and/or NetBSD. I got a "too many partitions" error from the Red Hat Linux fdisk. Can't I perform a full install of Red Hat Linux on my hard disk to remove the BSD disklabel data on my hard disk?

Ans4: No, you cannot. Once this error gets flagged, even fdisk's 'o' option which normally clears all partition information would not function. Even if you are successful in doing a complete Linux only "full disk" install using the Disk Druid disk-partitioning tool, this would not erase the old and already existing BSD disklabel information. Sooner or later, you would land in trouble.

Q5. I deleted all existing partitions containing data on my PC. But I want to be sure and wipe my disk clean with some "professional" disk-wiping utility. Which ones can I use? How would I use them?

Ans5: Third-party partition managers and security software which are specifically designed to erase disks can be used for this purpose. Personally, I do not have any experience using any third-party tools for wiping hard disks because I prefer using the UNIX-based dd utility.

UNIX's dd can also be used to wipe the disk clean absolutely. It writes zeroes to the disk surface which is accessible from the FreeBSD, OpenBSD and Linux install CDs. This is easily accessed from the OpenBSD install CD by selecting the shell option rather than install or using [Ctrl+C] to exit the OpenBSD install at any time. The OpenBSD command to clear the first IDE hard disk is "#dd if=/dev/zero of=/dev/wd0c". Red Hat Linux's install CD's rescue mode is a single user prompt. You do not have to mount the system as you are going to erase it, not "rescue" it. Once you get the "#" prompt "dd if=/dev/zero of=/dev/hda" will clear the first disk. FreeBSD's "Fixit" option from the main install menu provides access to a single user prompt; the second of the four install CDs is needed. The command "dd if=/dev/zero of=/dev/ad0" appears to clear the first disk. Once this done, you can be rest assured that your hard disk is wiped clean absolutely!

Q6. I wiped my hard disk clean using UNIX dd utility. I then performed a fresh install of Red Hat Linux on my PC followed by a FreeBSD install. I installed the FreeBSD boot manager into the MBR. Now my system does not boot! What happened? Where did I go wrong? or,

Q7. I wiped my hard disk clean using UNIX dd utility. I then performed a fresh install of Red Hat Linux on my PC followed by an OpenBSD install. I installed the OpenBSD boot manager into the MBR. Now my system does not boot! What happened? Where did I go wrong?

Ans6 and Ans7: Nothing went wrong anywhere. But you made a small yet fatal mistake at the very beginning. Always remember that when you write zeroes to the hard disk surface using UNIX dd utility, it includes the Master Boot Record (MBR) area also. The MBR area which has address **(0,0,1)**: Cylinder address 0, Head address 0 and Sector address 1, that is, in other words, the first 512 bytes of the hard disk which contains the Master Partition Table (MPT) and the Initial Program Load (IPL) code. When this area gets overwritten with zeroes, different operating systems react and behave in different ways. They react strangely because they expect the standard IPL code, but instead they find none.

For example, Red Hat Linux fdisk reports the zeroed partition table as invalid and can recreate an empty one; the only IPL code it will install is LILO or GRUB and not standard IPL code. OpenBSD simply shows an empty partition table and it causes a hell lot of problems booting on such hard disks. The only operating system that which is reported to boot flawlessly from a completely zeroed disk without error is FreeBSD. But, I personally did encounter a situation in which (owing to strange reasons) FreeBSD booting got locked and it simply would not boot.

To prevent all this from happening, just boot from a Windows 9x or DOS v6 boot disk with FDISK.EXE on it. Once the system is booted to an MS-DOS prompt, enter the following: A:\>fdisk /mbr. It reinitializes the MBR to its normal state and rewrites the MPT and creates the standard IPL. Continue installing the operating systems as usual.

Q8. I installed Red Hat Linux on my PC, but while installing LILO to the Master Boot Record (MBR), I got an error that said the installation program could not write this info to the MBR. What has happened?

Ans8: You have either locked the MBR of your hard disk on your PC or a virus-scan software is enabled which prevents writing to the MBR. You have to access your computer system's BIOS and verify that the MBR is not write-protected. Depending on the system, you may already have another boot loader in the MBR that conflicts with LILO. Try performing a "fdisk /mbr" from the MS-DOS command prompt and installing LILO.

Q9. I was installing OpenBSD 3.2 on my PC the other day. After creating partitions in OpenBSD fdisk, when I saved the changes and exited from fdisk using the option "quit", it flagged some warning as: "wd0: no disk label" in bright white on blue. What does that mean? Did I do something wrong?

Ans9: Absolutely no. You did nothing wrong. When you exit the OpenBSD fdisk (saving the changes using the option "quit"), and the OpenBSD slice is not at the same offset as a previously installed OpenBSD system, then fdisk displays "wd0: no disk label" message. Though it looks like a warning message or an apparent error, rather it is an important message

flagged by OpenBSD installation procedure which assures you that though the data and OpenBSD specific partitions have been created, yet the OpenBSD disklabel has to be setup. The disklabel defines the layout of the OpenBSD file systems on the OpenBSD partitions on the hard disk.

In dual and multi-booting systems, this message is almost a prerequisite for a correct OpenBSD installation. If it so happens that after saving changes to OpenBSD fdisk, you were not flagged this message, then it means that OpenBSD disklabel is reading and using information from a previous install. This is a dangerous situation because if it so happens that the disklabel is using old disklabel data and any partition information has changed since its creation initially, then disklabel's behavior becomes erratic and you may encounter strange problems. For preventing this from happening, make sure you wipe the disk clean before performing a clean install of OpenBSD.

Q10. I have FreeBSD 4.8-RELEASE and OpenBSD 3.2-RELEASE operating systems installed on my PC. I tried installing Red Hat Linux on the free hard disk space that I have. But each time I tried using fdisk or Disk Druid disk-partitioning tool, it flagged strange error messages. Why is this happening? What can I do to install Red Hat Linux on my system?

Ans10: An honest answer to this question is: "Nobody exactly knows why!". Over the years while performing dual and multi-booting installations, I have noticed that Red Hat Linux's fdisk and Disk Druid flags the most number of error messages whenever OpenBSD operating system is in the vicinity. OpenBSD fdisk handling of the partition table is different than FreeBSD and Linux and does not conform with the standards. This maybe one of the many possible explanations.

For example, let me describe a real situation. This happened a couple of weeks back (at the time of writing this *Guide*). I installed OpenBSD 3.2 followed by FreeBSD 4.8 on a test PC. Then I tried installing Red Hat Linux 8.0 (Psyche edition) on to the remaining hard disk space on the computer. Readers must note: I do not have any faulty hardware on my system, I meet all hardware compatibility requirements, I wiped my disk clean before starting the installs using the UNIX dd utility, the memory (physical RAM) available on my system is absolutely 100% correct and the Red Hat Linux 8.0 installation CD-ROMs are okay. Yet, each time when I tried to install Red Hat onto my system (each time using different boot-time command line options), Linux's fdisk started with several dialog boxes. The first said "Invalid partition on /tmp/hda". When I ignored the error, a new dialog displayed "Unable to align partition properly. This probably means that another partitioning tool generated an incorrect partition table, because it didn't have the correct BIOS geometry. It is safe to ignore, but ignoring may cause (fixable) problems with some boot loaders". Fdisk then displayed the "too many partitions" error. Error, error and error was all that I got! So, I switched to Red Hat Disk Druid to fancy my luck!

Red Hat's Disk Druid displayed an "Invalid partition on /tmp/hda" message and when ignored, showed the OpenBSD partition as unused and /dev/hda as a BSD/386 partition. I forced the "auto partitioning" process but it crashed miserably after some time with an "unhandled exception". I did not save the crash dump to a floppy because I had no plans to send a bug report.

This happened with Red Hat Linux 8.0 (Psyche). I met with same problems with Red Hat Linux distributions 7.3 and 7.1. But when I tried installing Red Hat Linux 7.0, it installed

painlessly without any shouts and cries. Could someone out there (or more specifically at Red Hat Inc.) please explain me what exactly happened or what is so special about Red Hat Linux 7.0 that the others do not have? I am still trying to figure out this problem!

Q11. I read the PR (problem report) above. I have FreeBSD and OpenBSD operating systems installed on my PC as well. But I would like to install a Linux distribution other than Red Hat on my system. Which one do you suggest?

Ans11: If you have already read and understood the PR above, then it makes no sense at all which Linux distribution you try installing on your system. I met with similar problems when I tried installing Mandrake 9.0 as well as SuSE 7.0 on my test computer. Mandrake Linux managed creating partitions, formatting them, but it too miserably crashed while installing packages. SuSE 7.0 crashed at the very beginning only! In the near future, I would like to test Red Hat Linux 9's integrity on such systems!

Q12. While installing FreeBSD 4.X-RELEASE on my computer the FreeBSD fdisk (or disklabel) used an 'X' partition name instead of a /dev/ad[0-3]s[1-4]n labeling scheme. Hey! what exactly happened? What does that 'X' represent?

Ans12: FreeBSD operating system software allows 7 partitions per slice (c: is reserved for the whole hard disk space). FreeBSD fdisk uses an 'X' partition instead of a valid partition name such as "/dev/ad0s1a" if you have created too many of them on your PC. If you have a partition labeled as 'X', you must delete it immediately without proceeding any further. If you do not delete this partition, FreeBSD disklabel will let you proceed with the entire installation, and then when you are finally done with the full installation, when your system comes up, you will encounter an error and would be dropped into single-user mode for maintenance. Thus, in other words, if you do not delete a partition which is labeled as 'X', FreeBSD will not complete the boot sequence. Instead it displays error messages and drops into single user mode.

[Previous](#)

[Home](#)

[Next](#)

[Previous](#)

[Home](#)

5. Conclusion

So folks, this brings us to the end of our journey of creating a multi-boot system configured with three modern operating systems. Assuming that you followed every step of this guide and everything went well, allow me to congratulate you on your success. If in case, something went wrong, make sure you read documentation and manuals of the relative operating system. Otherwise you can always refer to the Online manuals and resources.

Make sure you browse these links:

- ◆ [Distribution specific Official Red Hat Linux Installation and Upgrade manuals](#)
- ◆ [Linux Kernel resources](#)
- ◆ [Linux Loader \(LILO\) and other boot loader questions](#)
- ◆ [Red Hat Linux Hardware questions](#)

For referring to Online manuals dealing with "multi-booting" specifically, I consider these as some of the "very" best:

- ◆ [Installing and Using FreeBSD With Other Operating Systems](#)
- ◆ [An excellent page explaining booting process and methodologies.](#)

Ultimately, though, the responsibility for any errors or omissions you might find is mine and mine alone, so please address your ire to me at subhasish_ghosh@linuxwaves.com. For those who would like to send PRs (Problem Reports), please make sure you send enough valid and related information for me to work with. I cannot help you if you just send mails with messages like "This screwed up!", "That doesn't work!" and so on. Try sending detailed PRs. You may very well find your comments, suggestions and corrections reflected in the next version of this guide. Thank you!

[Previous](#)

[Home](#)